Latent Space Sparse Subspace Clustering

Vishal M. Patel, Hien Van Nguyen
Center for Automation Research, UMIACS
UMD, College Park, MD 20742
{pvishalm,hien}@umiacs.umd.edu

René Vidal
Center for Imaging Science
JHU, Baltimore, MD 21218
rvidal@cis.jhu.edu

Abstract

We propose a novel algorithm called Latent Space Sparse Subspace Clustering for simultaneous dimensionality reduction and clustering of data lying in a union of subspaces. Specifically, we describe a method that learns the projection of data and finds the sparse coefficients in the low-dimensional latent space. Cluster labels are then assigned by applying spectral clustering to a similarity matrix built from these sparse coefficients. An efficient optimization method is proposed and its non-linear extensions based on the kernel methods are presented. One of the main advantages of our method is that it is computationally efficient as the sparse coefficients are found in the low-dimensional latent space. Various experiments show that the proposed method performs better than the competitive state-of-the-art subspace clustering methods.

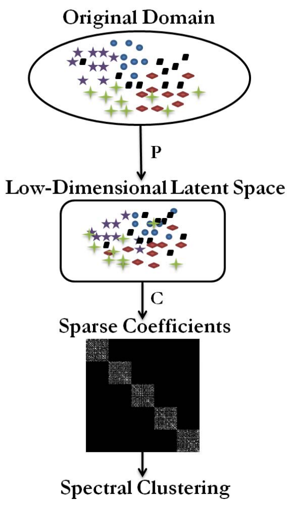

Figure 1. Overview of the proposed latent space sparse subspace clustering method.

1. Introduction

Many practical computer vision and image processing applications require processing and representation of high-dimensional data. Often these high-dimensional data can be represented by a low-dimensional subspace. For instance, it is well known that the set of face images under all possible illumination conditions can be well approximated by a 9-dimensional linear subspace [2]. Similarly, trajectories of a rigidly moving object in a video [4] and hand-written digits with different variations [10] also lie in low-dimensional subspaces. Therefore, one can view the collection of data from different classes as samples from a union of low-dimensional subspaces. In subspace clustering, given the data from a union of subspaces, the objective is to find the number of subspaces, their dimensions, the segmentation of the data and a basis for each subspace [25].

Various algorithms have been proposed in the literature for subspace clustering. Some of these algorithms are iterative in nature [11], [31] while the others are based on spectral clustering [3], [9], [29], [5]. Statistical [8] and algebraic [27], [12] approaches have also been proposed in the literature. In particular, sparse representation and low-rank approximation-based methods for subspace clustering [15], [7], [5], [6], [26], [20], [16] have gained a lot of traction in recent years. These methods find a sparse and low-rank representation of the data and build a similarity graph on the sparse coefficient matrix for segmenting the data. One of the advantages of these methods is that they are robust to noise and occlusion. Furthermore, some of these approaches do not require the knowledge of the dimensions and the number of subspaces. In particular, the Sparse Subspace Clustering (SSC) algorithm [5], [6] is well supported by theoretical analysis [23] and provides state-of-the-art results on many publicly available datasets such as the Hopkins155 benchmark motion segmentation dataset [24].

Computation of sparse and low-rank representations is very computationally demanding especially when the dimension of the features is high [6]. This is one of the drawbacks of the sparse and low-rank methods. To deal with this problem, dimensionally reduction is generally applied on the data prior to applying these algorithms. Dimensionally reduction methods such as Principle Component Analysis (PCA) and Random Projections (RP) can reduce the dimension of data. However, a well learned projection matrix can lead to a higher clustering accuracy at a lower dimensionality. Several works have been proposed in the literature that find a sparse representation on a low-dimensional latent space [17], [30], [19]. However, these methods are specifically designed for classification tasks and not for clustering.

Motivated by some of the sparsity promoting dimensionality reduction methods, in this paper, we propose a method for simultaneous dimensionality reduction and subspace clustering under the framework of SSC. We learn the transformation of data from the original space onto a low-dimensional space such that its manifold structure is maintained. An efficient algorithm is proposed that simultaneously learns the projection and finds the sparse coefficients in the low-dimensional latent space. Finally, the segmentation of the data is obtained by applying spectral clustering to a similarity matrix built from the sparse coefficients. Using kernel methods, the proposed algorithm is also extended to non-linear manifolds. Figure 1 presents an overview of our subspace clustering method.

Key contributions of our work are as follows:

- Simultaneous dimensionality reduction and sparse coding for SSC is proposed.
- A simple iterative procedure is introduced for solving the proposed optimization problem.
- Nonlinear extensions of the proposed algorithm are made through the use of Mercer kernels.

1.1. Organization of the paper

This paper is organized as follows. In Section 2, we provide a brief overview of the SSC method. Sections 3 and 4 give the details of our linear and non-linear simultaneous dimensionality reduction and subspace clustering approaches, respectively. Experimental results are presented in Section 5 and Section 6 concludes the paper with a brief summary and discussion.

2. Background

In this section, we give an overview of the SSC algorithm. Let $\mathbf{Y} = [\mathbf{y}_1, \cdots, \mathbf{y}_N] \in \mathbb{R}^{D\times N}$ be a collection of N signals $\{\mathbf{y}_i \in \mathbb{R}^D\}_{i=1}^N$ drawn from a union of n linear subspaces

$$\mathcal{S}_1 \cup \mathcal{S}_2 \cup \cdots \cup \mathcal{S}_n$$

of dimensions $\{d_\ell\}_{\ell=1}^n$ in $\mathbb{R}^D$. Let $\mathbf{Y}_\ell \in \mathbb{R}^{D\times N_\ell}$ be a submatrix of $\mathbf{Y}$ of rank d_ℓ with $N_\ell > d_\ell$ points that lie in $\mathcal{S}_\ell$ with $N_1 + N_2 + \cdots + N_n = N$. It is easy to see that each data point in $\mathbf{Y}$ can be efficiently represented by a linear combination of at most d_ℓ other points in $\mathbf{Y}$. That is, one can represent $\mathbf{y}_i$ as follows

$$\mathbf{y}_i = \mathbf{Y}\mathbf{c}_i, \ \ c_{ii} = 0, \ \|\mathbf{c}_i\|_0 \le d_\ell,$$

where $\mathbf{c}_i = [c_{i1}, c_{i2}, \cdots, c_{iN}]^T \in \mathbb{R}^N$ are the coefficients. Often $N_\ell > d_\ell$. As a result the following ℓ_1-minimization problem is solved to obtain the coefficients

$$\min \|\mathbf{c}\|_1 \text{ such that } \mathbf{y}_i = \mathbf{Y}\mathbf{c}_i, \ c_{ii} = 0, \tag{1}$$

where $\|\mathbf{x}\|_1 = \sum_{i=1}^N |x_i|$ is the ℓ_1-norm of $\mathbf{x} \in \mathbb{R}^N$. Considering all the data points $i = 1, \cdots, N$, in matrix form, the above optimization problem can be rewritten as

$$\min \|\mathbf{C}\|_1 \ \text{ subject to } \ \mathbf{Y} = \mathbf{Y}\mathbf{C}, \ \ \mathrm{diag}(\mathbf{C}) = \mathbf{0}, \tag{2}$$

where $\mathbf{C} = [\mathbf{c}_1, \cdots, \mathbf{c}_N] \in \mathbb{R}^{N\times N}$ is the coefficient matrix whose column $\mathbf{c}_i$ is the sparse representation vector corresponding to $\mathbf{y}_i$. Once $\mathbf{C}$ is found, spectral clustering methods [18] are applied on the affinity matrix $\mathbf{W} = |\mathbf{C}| + |\mathbf{C}|^T$ to obtain the segmentation of the data $\mathbf{Y}$ into $\mathbf{Y}_1, \mathbf{Y}_2, \cdots, \mathbf{Y}_n$.

In the case where the data is contaminated by some arbitrary noise $\mathbf{Z}$, i.e. $\mathbf{Y} = \mathbf{Y}\mathbf{C} + \mathbf{Z}$, the following problem can be solved to obtain $\mathbf{C}$

$$\min \|\mathbf{C}\|_1 + \lambda_1 \|\mathbf{Y} - \mathbf{Y}\mathbf{C}\|_F^2 \quad \text{s. t.} \quad \mathrm{diag}(\mathbf{C}) = \mathbf{0}, \tag{3}$$

where $\|.\|_F$ denotes the Frobenius norm. In practice, the data may lie in a union of affine subspaces. In this case, the following problem can be solved to obtain the sparse coefficients

$$\min \|\mathbf{C}\|_1 \ \text{ s. t.} \mathbf{Y} = \mathbf{Y}\mathbf{C}, \mathrm{diag}(\mathbf{C}) = \mathbf{0}, \ \mathbf{C}^T\mathbf{1} = \mathbf{1}. \tag{4}$$

The above problems can be efficiently solved by using the classical alternating direction method of multipliers (ADMM) [6].

3. Latent Space SSC (LS3C)

Different from the traditional SSC, we develop an algorithm that embeds signals into a low-dimensional space and simultaneously finds the sparse codes in that space. Let $\mathbf{P} \in \mathbb{R}^{t\times D}$ be a matrix representing a linear transformation that maps signals from the original space $\mathbb{R}^D$ to a latent output space of dimension t. We can learn the mapping and find the sparse codes simultaneously by minimizing the following cost function

$$\begin{aligned}[\mathbf{P}^*, \mathbf{C}^*] &= \min_{\mathbf{P},\mathbf{C}} \mathcal{J}(\mathbf{P}, \mathbf{C}, \mathbf{Y}) \\ \text{subject to} \quad & \mathbf{P}\mathbf{P}^T = \mathbf{I}, \mathrm{diag}(\mathbf{C}) = \mathbf{0},\end{aligned} \tag{5}$$

where

$$\begin{aligned}\mathcal{J}(\mathbf{P}, \mathbf{C}, \mathbf{Y}) = \|\mathbf{C}\|_1 &+ \lambda_1 \|\mathbf{P}\mathbf{Y} - \mathbf{P}\mathbf{Y}\mathbf{C}\|_F^2 \\ &+ \lambda_2 \|\mathbf{Y} - \mathbf{P}^T\mathbf{P}\mathbf{Y}\|_F^2.\end{aligned} \tag{6}$$

The first two terms of $\mathcal{J}$ essentially promote sparsity of data in the reduced space. The second term, which is a PCA-like regularization term, ensures that the projection does not loose too much information available in the original domain. λ_1 and λ_2 are non-negative constants that control sparsity and regularization, respectively. Furthermore, we require the rows of $\mathbf{P}$ to be orthogonal and normalized to unit norm. This prevents the solution from becoming degenerate and the leads to a computationally efficient scheme for optimization.

Note that the above formulation can be extended so that it can deal with data that lie in a union of affine subspaces. This can be simply done by adding a constraint in the optimization problem (5) as follows

$$\begin{aligned}&[\mathbf{P}^*, \mathbf{C}^*] = \min_{\mathbf{P},\mathbf{C}} \mathcal{J}(\mathbf{P}, \mathbf{C}, \mathbf{Y})\\ &\text{subject to} \quad \mathbf{P}\mathbf{P}^T = \mathbf{I}, \mathbf{C}^T\mathbf{1} = \mathbf{1}, \text{diag}(\mathbf{C}) = \mathbf{0}.\end{aligned} \tag{7}$$

3.1. Optimization

With the above definitions, one can prove the following proposition.

Proposition 1 *There exists an optimal solution* $\mathbf{P}^*$ *to (5) that has the following form*

$$\mathbf{P}^* = \mathbf{\Psi}^T\mathbf{Y}^T$$

for some $\mathbf{\Psi} \in \mathbb{R}^{N\times t}$.

Intuitively, the above proposition says that the projection can be written as a linear combination of the data samples. This formulation has been used under the framework of dictionary learning in [21].

With this proposition, the cost function (8) can be written as

$$\begin{aligned}\mathcal{J}(\mathbf{\Psi}, \mathbf{C}, \mathbf{Y}) = \|\mathbf{C}\|_1 + \lambda_1\|\mathbf{\Psi}^T\mathbf{K}(\mathbf{I}-\mathbf{C})\|_F^2\\ + \lambda_2\|\mathbf{Y}(\mathbf{I}-\mathbf{\Psi}\mathbf{\Psi}^T\mathbf{K})\|_F^2,\end{aligned} \tag{8}$$

where $\mathbf{K} = \mathbf{Y}^T\mathbf{Y}$. The equality constrain now becomes

$$\mathbf{P}\mathbf{P}^T = \mathbf{\Psi}^T\mathbf{K}\mathbf{\Psi} = \mathbf{I}. \tag{9}$$

As a result, the optimization problem (5) can be re-written as

$$\begin{aligned}&[\mathbf{\Psi}^*, \mathbf{C}^*] = \min_{\mathbf{\Psi},\mathbf{C}} \mathcal{J}(\mathbf{\Psi}, \mathbf{C}, \mathbf{Y})\\ &\text{subject to} \quad \mathbf{\Psi}^T\mathbf{K}\mathbf{\Psi} = \mathbf{I}, \text{diag}(\mathbf{C}) = \mathbf{0}.\end{aligned} \tag{10}$$

This formulation allows the update of $\mathbf{P}$ via $\mathbf{\Psi}$. Furthermore, as will become apparent later, this form of the cost function makes it easier to extend the algorithm to non-linear manifolds using kernel methods. We can solve the above optimization problem by optimizing over $\mathbf{\Psi}$ and $\mathbf{C}$ iteratively.

3.2. Update step for $\mathbf{\Psi}$

In this step, we assume that $\mathbf{C}$ is fixed. So the sparsity constraint can be removed and the following problem needs to be solved

$$\begin{aligned}\lambda_1\|\mathbf{\Psi}^T\mathbf{K}(\mathbf{I}-\mathbf{C})\|_F^2 + \lambda_2\|\mathbf{Y}(\mathbf{I}-\mathbf{\Psi}\mathbf{\Psi}^T\mathbf{K})\|_F^2\\ \text{subject to} \quad \mathbf{\Psi}^T\mathbf{K}\mathbf{\Psi} = \mathbf{I}.\end{aligned} \tag{11}$$

The cost function can be expanded as follows

$$\begin{aligned}&\text{trace}\left(\lambda_1(\mathbf{I}-\mathbf{C})(\mathbf{I}-\mathbf{C})^T\mathbf{K}^T\mathbf{Q}^T\mathbf{K}\right)\\ &+ \text{trace}\left(\lambda_2(\mathbf{K} - 2\mathbf{K}^T\mathbf{Q}^T\mathbf{K} + \mathbf{K}^T\mathbf{Q}^T\mathbf{K}\mathbf{Q}\mathbf{K})\right),\end{aligned} \tag{12}$$

where $\mathbf{Q} = \mathbf{\Psi}\mathbf{\Psi}^T \in \mathbb{R}^{N\times N}$. The constraint $\mathbf{\Psi}^T\mathbf{K}\mathbf{\Psi} = \mathbf{I}$ leads to the new constraint $\mathbf{\Psi}\mathbf{\Psi}^T\mathbf{K}\mathbf{\Psi}\mathbf{\Psi}^T = \mathbf{\Psi}\mathbf{\Psi}^T$ or $\mathbf{Q}\mathbf{K}\mathbf{Q}^T = \mathbf{Q}$. The objective function (12) can be further simplified as

$$\text{trace}\left(\left(\lambda_1(\mathbf{I}-\mathbf{C})(\mathbf{I}-\mathbf{C})^T - \lambda_2\mathbf{I}\right)\mathbf{K}^T\mathbf{Q}^T\mathbf{K}\right), \tag{13}$$

where we have made use of the equality constraint and used the fact that trace$(\mathbf{K})$ is constant. Using the eigen decomposition of $\mathbf{K} = \mathbf{V}\mathbf{S}\mathbf{V}^T$, we get

$$\mathbf{K}^T\mathbf{Q}^T\mathbf{K} = \mathbf{V}\mathbf{S}^{\frac{1}{2}}\mathbf{M}\mathbf{M}^T\mathbf{S}^{\frac{1}{2}}\mathbf{V}^T,$$

where $\mathbf{M} = \mathbf{S}^{\frac{1}{2}}\mathbf{V}^T\mathbf{\Psi}$. As a result, (13) can be rewritten as

$$\text{trace}\left(\mathbf{M}^T\mathbf{S}^{\frac{1}{2}}\mathbf{V}^T\left(\lambda_1(\mathbf{I}-\mathbf{C})(\mathbf{I}-\mathbf{C})^T - \lambda_2\mathbf{I}\right)\mathbf{V}\mathbf{S}^{\frac{1}{2}}\mathbf{M}\right).$$

Using the fact that, $\mathbf{\Psi}^T\mathbf{K}\mathbf{\Psi} = \mathbf{M}^T\mathbf{M}$ and with the following change of variable

$$\mathbf{\Delta} = \mathbf{S}^{\frac{1}{2}}\mathbf{V}^T\left(\lambda_1(\mathbf{I}-\mathbf{C})(\mathbf{I}-\mathbf{C})^T - \lambda_2\mathbf{I}\right)\mathbf{V}\mathbf{S}^{\frac{1}{2}},$$

we arrive at the following optimization problem, which is equivalent to (11)

$$\mathbf{M}^* = \min_{\mathbf{M}} \ \text{trace}\left(\mathbf{M}^T\mathbf{\Delta}\mathbf{M}\right) \ \text{ s.t. } \ \mathbf{M}^T\mathbf{M} = \mathbf{I}. \tag{14}$$

Problem (14) is the classical minimum eigenvalue problem whose solution is given by the ℓ eigenvectors associated with the first ℓ smallest eigenvalues of $\mathbf{\Delta}$ [22]. Once the optimal $\mathbf{M}^*$ is found, the optimal $\mathbf{\Psi}^*$ can be recovered as

$$\mathbf{\Psi}^* = \mathbf{V}\mathbf{S}^{-\frac{1}{2}}\mathbf{M}^*.$$

Hence, we have proved the following proposition:

Proposition 2 *The optimal solution of (10) when* $\mathbf{C}$ *is fixed is*

$$\mathbf{\Psi}^* = \mathbf{V}\mathbf{S}^{-\frac{1}{2}}\mathbf{M}^*, \tag{15}$$

where $\mathbf{V}$ *and* $\mathbf{S}$ *come from the eigen decomposition of* $\mathbf{K} = \mathbf{V}\mathbf{S}\mathbf{V}^T$*, and* $\mathbf{M}^* \in \mathbb{R}^{N\times t}$ *is the optimal solution of the following minimum eigenvalues problem*

$$\mathbf{M}^* = \min_{\mathbf{M}} \ \textit{trace}\left(\mathbf{M}^T\mathbf{\Delta}\mathbf{M}\right) \ \textit{ s.t. } \ \mathbf{M}^T\mathbf{M} = \mathbf{I}. \tag{16}$$

where

$$\mathbf{\Delta} = \mathbf{S}^{\frac{1}{2}}\mathbf{V}^T\left(\lambda_1(\mathbf{I}-\mathbf{C})(\mathbf{I}-\mathbf{C})^T - \lambda_2\mathbf{I}\right)\mathbf{V}\mathbf{S}^{\frac{1}{2}}.$$

3.3. Update step for $\mathbf{C}$

For a fixed $\boldsymbol{\Psi}$, we have to solve the following problem to obtain $\mathbf{C}$:

$$\min_{\mathbf{C}} \|\mathbf{C}\|_1 + \lambda_1 \|\mathbf{B} - \mathbf{B}\mathbf{C}\|_F^2 \text{ s.t. } \operatorname{diag}(\mathbf{C}) = \mathbf{0}, \quad (17)$$

where $\mathbf{B} = \boldsymbol{\Psi}^T\mathbf{K}$. This problem is the same as the SSC problem, except that the data matrix $\mathbf{Y}$ is replaced by the $\mathbf{B}$ matrix. Therefore, it can be solved by the ADMM method in [6].

4. Non-Linear Latent Space SSC (NLS3C)

In [6], it was shown that the performance of the SSC method depends on the principle angles between subspaces and the distribution of the data in each subspace. The smallest principle angle, θ_{ij}, between two subspaces $\mathcal{S}_i$ and $\mathcal{S}_j$ is defined as

$$\cos(\theta_{i,j}) = \max_{\mathbf{a}_i \in \mathcal{S}_i \mathbf{a}_j \in \mathcal{S}_j} \frac{\mathbf{a}_i^T \mathbf{a}_j}{\|\mathbf{a}_i\|_2 \|\mathbf{a}_j\|_2}.$$

In particular, when the smallest principle angle between subspaces and the number of data points in each subspace is small, the SSC clustering error increases. By mapping the data onto a high-dimensional feature space and then projecting back onto a low-dimensional space, one may be able deal with this small-principle-angles problem.

Let $\Phi : \mathbb{R}^D \to \mathcal{H}$ be a mapping from the input space to the reproducing kernel Hilbert space $\mathcal{H}$. The non-linear mapping $\mathcal{P}$ can be characterized by a compact linear operator $\boldsymbol{\mathcal{P}} : \mathcal{H} \to \mathbb{R}^t$. Let $\boldsymbol{\mathcal{K}} \in \mathbb{R}^{N\times N}$ be a positive semidefinite kernel Gram matrix whose elements are computed as

$$\begin{aligned} [\boldsymbol{\mathcal{K}}(\mathbf{Y},\mathbf{Y})]_{i,j} &= [\langle \Phi(\mathbf{Y}), \Phi(\mathbf{Y}) \rangle_{\mathcal{H}}]_{i,j} \\ &= \Phi(\mathbf{y}_i)^T \Phi(\mathbf{y}_j) \\ &= \kappa(\mathbf{y}_i, \mathbf{y}_j), \end{aligned} \quad (18)$$

where $\kappa : \mathbb{R}^D \times \mathbb{R}^D \to \mathbb{R}$ is the kernel function and $\Phi(\mathbf{Y}) = [\Phi(\mathbf{y}_1), \Phi(\mathbf{y}_2), \cdots, \Phi(\mathbf{y}_N)]$. Some commonly used kernels include polynomial kernels $\kappa(\mathbf{x},\mathbf{y}) = \left(\langle \mathbf{x},\mathbf{y}\rangle + a\right)^b$, Gaussian kernels $\kappa(\mathbf{x},\mathbf{y}) = \exp\left(-\sigma\|\mathbf{x}-\mathbf{y}\|^2\right)$, and hyperbolic tangent kernels $\kappa(\mathbf{x},\mathbf{y}) = \tanh(s\mathbf{x}^T\mathbf{y} + o)$, where a, b, σ, s and o are the parameters of the kernel functions.

With the above definitions, the cost function for the non-linear LS3C can be written as follows

$$\begin{aligned} \mathcal{J}(\boldsymbol{\mathcal{P}}, \mathbf{C}, \mathbf{Y}) = \|\mathbf{C}\|_1 &+ \lambda_1 \|\boldsymbol{\mathcal{P}}\Phi(\mathbf{Y}) - \boldsymbol{\mathcal{P}}\Phi(\mathbf{Y})\mathbf{C}\|_F^2 \\ &+ \lambda_2 \|\Phi(\mathbf{Y}) - \boldsymbol{\mathcal{P}}^T\boldsymbol{\mathcal{P}}\Phi(\mathbf{Y})\|_F^2. \end{aligned}$$

This formulation is the same as that in (8) except that $\mathbf{Y}$ is now replaced by $\Phi(\mathbf{Y})$. Furthermore, similar to Proposition 1, it can be shown that the optimal projection takes the following form

$$\boldsymbol{\mathcal{P}}^* = \boldsymbol{\Psi}^T \Phi(\mathbf{Y})^T. \quad (19)$$

As a result, we get the following cost function

$$\begin{aligned} \mathcal{J}(\boldsymbol{\Psi}, \mathbf{C}, \boldsymbol{\mathcal{K}}) &= \|\mathbf{C}\|_1 + \lambda_1 \|\boldsymbol{\Psi}^T\boldsymbol{\mathcal{K}}(\mathbf{I}-\mathbf{C})\|_F^2 \\ &+ \lambda_2 \operatorname{trace}\Big((\mathbf{I} - \boldsymbol{\Psi}\boldsymbol{\Psi}^T\boldsymbol{\mathcal{K}})^T \boldsymbol{\mathcal{K}} (\mathbf{I} - \boldsymbol{\Psi}\boldsymbol{\Psi}^T\boldsymbol{\mathcal{K}})\Big) \end{aligned} \quad (20)$$

and the constraint $\boldsymbol{\mathcal{P}}\boldsymbol{\mathcal{P}}^T = \mathbf{I}$ becomes $\boldsymbol{\Psi}^T\boldsymbol{\mathcal{K}}\boldsymbol{\Psi} = \mathbf{I}$. This optimization problem can be solved in the same way as the linear case. Note that the dimension of the output space is upper bounded by the number of training samples. Both the linear and non-linear methods for finding the sparse coefficient matrix in the latent space along with the projection matrix are summarized in Algorithm 1. Note that the optimization problem is non-convex. However, numerical simulations have shown that the algorithm usually converges to a local minimum in a few iterations.

Algorithm 1: Simultaneous dimension reduction and sparse coding for both linear and non-linear case.

```
Input: Kernel matrix $\boldsymbol{\mathcal{K}} \in \mathbb{R}^{N\times N}, \lambda_1, \lambda_2$.
Initialization:
- Set iteration $J = 1$. Perform eigen decomposition $\boldsymbol{\mathcal{K}} = \mathbf{V}\mathbf{S}\mathbf{V}^T$.
- Set $\boldsymbol{\Psi} = \mathbf{V}(:, \mathcal{I})$, where $\mathcal{I}$ is the index set of the $d$ largest eigenvalues of $\boldsymbol{\mathcal{K}}$.
Stage 1: Fix $\boldsymbol{\Psi}$ and update $\mathbf{C}$
- Compute $\mathbf{B} = \boldsymbol{\Psi}^T\boldsymbol{\mathcal{K}}$.
- Solve the sparse optimization problem (17) to obtain $\mathbf{C}$.
Stage 2: Fix $\mathbf{C}$ and update $\boldsymbol{\Psi}$
- Compute $\boldsymbol{\Delta} = \mathbf{S}^{\frac{1}{2}}\mathbf{V}^T\left(\lambda_1(\mathbf{I}-\mathbf{C})(\mathbf{I}-\mathbf{C})^T - \lambda_2\mathbf{I}\right)\mathbf{V}\mathbf{S}^{\frac{1}{2}}$.
- Perform eigen decomposition of $\boldsymbol{\Delta} = \mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$.
- Set $\mathbf{M} = \mathbf{U}(:, \mathcal{I}_J)$, where $\mathcal{I}_J$ is the index set of the $d$ smallest eigenvalues of $\boldsymbol{\Delta}$.
- Update $\boldsymbol{\Psi} = \mathbf{V}\mathbf{S}^{-\frac{1}{2}}\mathbf{M}$.
    - Increment $J = J + 1$. Repeat from stage 1 until stopping conditions reached.
Output: $\mathbf{C}$ and $\boldsymbol{\Psi}$.
```

Similar to the SSC method, once the sparse coefficient matrix $\mathbf{C}$ is found, spectral clustering is applied on the affinity matrix $\mathbf{W} = |\mathbf{C}| + |\mathbf{C}|^T$ to obtain the segmentation of the data in the low-dimensional latent space. The resulting latent space SSC (LS3C) and non-linear latent space SSC (NLS3C) methods are summarized in Algorithm 2.

Algorithm 2: Latent Space Sparse Subspace Clustering for both linear and non-linear cases.

```
Input: Kernel matrix $\boldsymbol{\mathcal{K}} \in \mathbb{R}^{N\times N}, \lambda_1, \lambda_2$.
Algorithm:
- Apply Algorithm 1 to find the sparse coefficient matrix $\mathbf{C}$.
- Normalize the columns of $\mathbf{C}$ as $\mathbf{c}_i \leftarrow \frac{\mathbf{c}_i}{\|\mathbf{c}_i\|_\infty}$.
- Form a similarity graph with $N$ nodes and set the weights on the edges between the nodes by $\mathbf{W} = |\mathbf{C}| + |\mathbf{C}|^T$.
- Apply spectral clustering to the similarity graph.
Output: Cluster labels for all signals.
```

5. Experimental Results

In this section, we evaluate our proposed method on both synthetic and real datasets. We compare our method with several state-of-the-art subspace clustering algorithms such as SSC [6], Low-Rank Representation (LRR) [15], Low-Rank Subspace Clustering (LRSC) [7], Local Subspace Affinity (LSA) [29] and Spectral Curvature Clustering (SCC) [3]. For all the experiments, we set the maximum number of iteration in our Algorithm 1 to $J = 3$. We set $\lambda_1 = \lambda_2 = 50$. All the experiments are done on an OS X system with 2.6 GHz Intel Core i7 processor using Matlab. Subspace clustering error is used to measure the performance of different algorithms. It is defined as

$$\text{subspace clustering error} = \frac{\#\text{of misclassified points}}{\text{total}\#\text{of points}} \times 100.$$

5.1. Synthetic Data

In this section, we generate a synthetic data to study the performance of LNS3C and NLS3C when the data in each subspace and the smallest principle angle between subspaces are small. We follow the same experimental setting as in [6]. We consider $n = 3$ subspaces of dimension $d = 3$ embedded in $D = 50$ dimensional space. We generate the bases $\{\mathbf{T}_i \in \mathbb{R}^{D\times d}\}_{i=1}^3$ such that $\operatorname{rank}([\mathbf{T}_1, \mathbf{T}_2, \mathbf{T}_3]) = 2d$. Also, the subspaces are generated such that $\theta_{12} = \theta_{23} = \theta$. Furthermore, we generate the same number of points, N_g, in each subspace at random and change the value of N_g.

For a fixed value of d, we change the minimum angle between subspaces, θ, as well as the number of points in each subspace N_g. For each pair of (θ, N_g), we compute the subspace clustering error. Since the performance of SSC and NSSC methods are based on how well the sparse coefficients are found, we also calculate the subspace sparse recovery error. For the data points $\{\mathbf{y}_i\}_{i=1}^{3N_g}$, the sparse recovery error E_{SR} is given by

$$E_{SR} = \frac{1}{3N_g}\sum_{i=1}^{3N_g}\left(1 - \frac{\|\mathbf{c}_{iq_i}\|_1}{\|\mathbf{c}_i\|_1}\right),$$

where $\mathbf{c}_i^T = [\mathbf{c}_{i1}{}^T, \mathbf{c}_{i2}{}^T, \mathbf{c}_{i3}{}^T]$ represents the sparse coefficients of$\mathbf{y}_i \in \mathcal{S}_{q_i}$ and $\mathbf{c}_{ij}$ corresponds to the points in $\mathcal{S}_j$.

We vary the smallest principle angle between subspaces and the number of points in each subspace as $\theta \in [6, 60]$ and $N_g \in [d+1, 20d]$, respectively. For each pair (θ, N_g), we calculate the average subspace clustering error as well as the average E_{SR} over 20 trials. In each trial we randomly generate data points and subspaces. Results of this experiment are shown in Figure 2. When θ and N_g decrease both the sparse recovery and clustering errors of all the methods increase. Also, the clustering error is highly dependent on the sparse recovery error and both errors follow the same pattern. In other words, clustering results are highly dependent on how well the sparse coefficients are recovered. By comparing the decay of errors, one can see that in the case where both θ and N_g are small, our methods perform better than the SSC method. The error decays faster in the case of LS3C and NLS3C than SSC. This can be explained by the fact that our method finds the projection directly from data and preserves the sparse structure of data in the latent space. In this experiment, for NLS3C we used a polynomial kernel with parameters $b = 1$ and $a = 0.015$.

5.2. Motion Segmentation

In motion segmentation, the idea is to segment a video sequence into multiple spatiotemporal regions corresponding to different rigid body motions. Suppose that we have tracked N feature points over F frames in a video sequence, $\{\mathbf{x}_{ij}\}$, where $i = 1, \cdots, N$ and $j = 1 \cdots, F$. Each feature trajectory $\mathbf{y}_i \in \mathbb{R}^{2F}$ is obtained by stacking the feature points in the video, i.e

$$\mathbf{y}_i^T = [\mathbf{x}_{1i}^T, \mathbf{x}_{2i}^T, \cdots, \mathbf{x}_{Fi}^T].$$

Then, the objective is to separate these feature trajectories according to their motions. It has been shown that trajectories of a general rigid motion under affine projection span a $4n$-dimensional linear subspace [4]. In other words, feature trajectories of n rigid motions lie in a union of n-dimensional subspaces of $\mathbb{R}^{2F}$. Hence, the problem of clustering the trajectories according to the different motion is equivalent to the problem of clustering affine subspaces. In our framework, this can be achieved by solving (7) that enforces the constraint that $\mathbf{C}^T\mathbf{1} = \mathbf{1}$.

We apply our joint dimensionality reduction and subspace clustering framework to the Hopkins155 motion segmentation database [24]. The dataset contains 155 video sequences where 120 video sequences contain 2 motions and 35 video sequences have 3 motions. For each sequence, a tracker is used to extract the point trajectories and the outliers are extracted manually [24]. On average, each sequence of 2 motions has 266 feature trajectories and 30 frames and each sequence of 3 motions has 398 feature trajectories and 29 frames. For the NLS3C, we used a polynomial kernel with parameters $a = 0.8$ and $b = 1$.

Table 1 compares the performance of different methods. For the methods other than LS3C and NLS3C, the data is first projected onto the 4-dimensional subspace using PCA [6]. As can be seen from this table, on average our method LS3C performs the best on both 2 motion and 3 motion sequences. It is able to learn the projection directly from the data better than PCA. Non-linear LS3C also obtains small clustering errors compared to the other competitive subspace clustering methods.

In the second set of experiments with the Hopkins155 dataset, we study the performance of different methods as

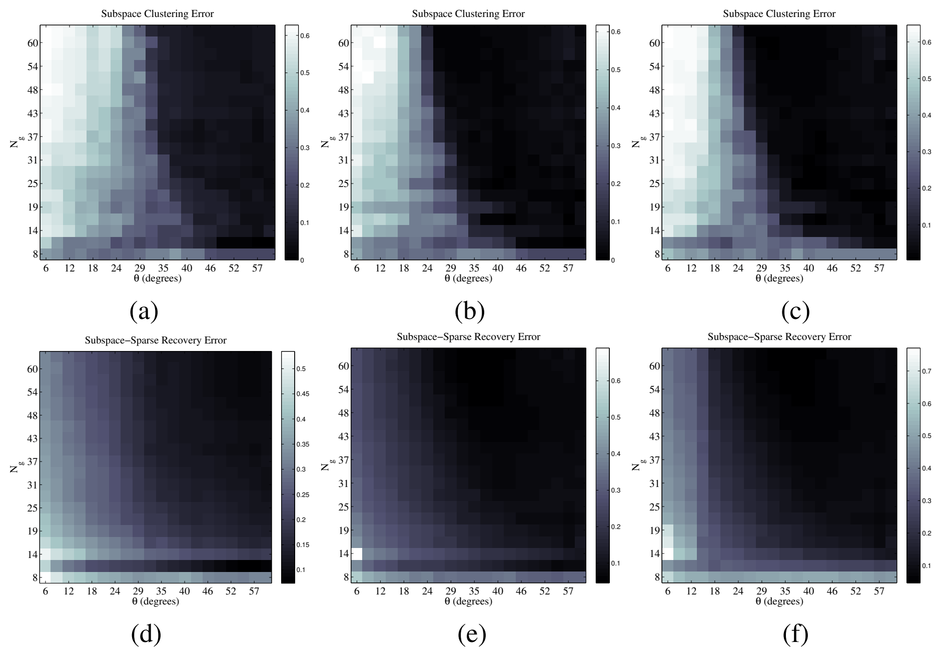

Figure 2. Subspace clustering errors and subspace-sparse recovery errors for three randomly generated disjoint subspaces with different smallest principle angles and different number of points. The average subspace clustering and subspace-sparse recovery errors of the SSC method are shown in (a) and (d), respectively. The average subspace clustering and subspace-sparse recovery errors of the LS3C method are shown in (b) and (e), respectively. The average subspace clustering and subspace-sparse recovery errors of the NLS3C method are shown in (c) and (f), respectively.

Algorithms (2 Motions)	SSC	LRR	SCC	LSA	LRSC	LS3C	NLS3C
Mean	1.83	3.41	3.04	3.61	3.87	**1.62**	1.79
Median	0.00	0.00	0.00	0.51	0.26	0.00	0.00
Algorithms (3 Motions)	SSC	LRR	SCC	LSA	LRSC	LS3C	NLS3C
Mean	4.40	4.86	7.91	7.65	7.72	**4.38**	4.89
Median	0.56	1.47	1.14	1.27	3.80	0.56	0.85
Algorithms (All)	SSC	LRR	SCC	LSA	LRSC	LS3C	NLS3C
Mean	2.41	3.74	4.14	4.52	4.74	**2.31**	2.56
Median	0.00	0.00	0.00	0.57	0.58	0.00	0.00

Table 1. Clustering errors on the Hopkins155 dataset with the $4n$-dimensional data points.

we vary the subspace dimensions. We project the data onto the following dimensional subspaces: $\{2n, 6n, 8n, 10n\}$. For the LRR and SSC methods, we project the data onto the low-dimensional space using random projections. Random projections have been used for dimensionality reduction in many sparsity-based algorithms [28], [5] and they have been shown to preserve the sparsity of data provided certain conditions are met [1]. Let $\mathbf{P}$ be an $t \times D$ random matrix with $t \leq D$ such that each entry $p_{i,j}$ of $\mathbf{P}$ is an independent realization of q, where q is a random variable on a probability measure space (Ω, ρ). It has been shown that given any set of points Λ, the following are some of the matrices that provide the sparsest solution via ℓ_1 minimization problem provided that enough measurements are taken [1]:

- RP1: $t \times D$ random matrix $\mathbf{P}$ whose entries $p_{i,j}$ are independent realizations of Gaussian random variables $p_{i,j} \sim \mathcal{N}\left(0, \frac{1}{t}\right)$.

- RP2: Independent realizations of ± 1 Bernoulli random variables

$$p_{i,j} \doteq \begin{cases} +1/\sqrt{t}, & \text{with probability } \frac{1}{2} \\ -1/\sqrt{t}, & \text{with probability } \frac{1}{2}. \end{cases}$$

We use both RP1 and RP2 to project the data points onto a low-dimensional space. The average clustering error results are summarized in Table 2. As can be seen from this table that LS3C and NLS3C consistently outperform the other methods in all dimensions. It is also interesting to note that the performance of both SSC and LRR varies depending on the projection matrix used for dimensionality reduction. In other words, features are important for both SSC and LRR. RP2 performs a little worse than the RP1. In contrast, our method automatically learns the features directly from the data and consistently performs better than LRR and SSC.

5.3. Rotated MNIST Dataset

The rotated MNIST benchmark [13] contains gray scale images of hand-written digits of size 28×28 pixels. The

Algorithms (2 Motions)	SSC (RP2)	SSC (RP1)	LRR (RP2)	LRR (RP1)	LS3C	NLS3C
$2n$-Dim	5.08	4.29	13.91	12.73	$\mathbf{3.23}$	3.86
$6n$-Dim	2.40	2.65	8.35	8.44	$\mathbf{2.31}$	2.57
$8n$-Dim	2.60	2.92	7.79	7.36	$\mathbf{2.29}$	2.57
$10n$-Dim	2.40	2.59	7.90	7.74	$\mathbf{2.29}$	2.57

Table 2. Average clustering errors on the Hopkins155 dataset with different dimensional data points.

images were originally taken from the MNIST dataset introduced in [14], and transformed in several ways to create more challenging classification problems. In the first dataset, called the $mnist\text{-}rot$, digits are rotated by random angles generated uniformly between 0 and 2π radians. The second dataset, called the $mnist\text{-}rot\text{-}back\text{-}image$, is created by inserting random backgrounds into $mnist\text{-}rot$ dataset. The $mnist\text{-}back\text{-}rand$ dataset is created by inserting random backgrounds in the original MNIST digit images. For all 3 datasets, there are 10000, 2000, and 50000 images for training, validation, and testing, respectively. Figure 3 shows sample images from the above datasets.

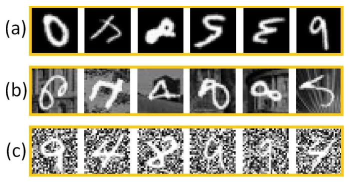

Figure 3. Sample digits from the rotated MNIST dataset. (a) Digits with random rotations, (b) Digits with random rotations and image backgrounds, (c) Digits with random backgrounds.

We evaluate the clustering performance of LS3C and NLS3 as well as SSC and LRR on this challenging dataset. It was shown in [10] that handwritten digits with some variations lie on 12-dimensional subspaces. Hence, n handwritten digits can be modeled as data points lying close to a union of 12-dimensional subspaces. Since this dataset contains a large amount of samples (about 62,000 samples), we only use samples from the training and the validation sets (12,000 samples) for clustering. In particular, we select 10 samples per digit and generate a small subset containing 100 samples from 10 digits. We use these samples for clustering and repeat the process 120 times so that all the samples from the training and the validation sets are used for clustering.

We report the average clustering performances of different methods in Table 3. As can be seen from this table, in all cases, NLS3C performances compare favorably to the state-of-the-art. By non-linearly projecting the data we are able to capture the compact structure of data that is more robust against noise. Polynomial kernel with $a = 1, b = 4$ is used in this experiment. The performance of LS3C is also comparable to that of SSC. Even though LRR and SSC can separate the background and remove noise from the data, they do not perform well on this dataset. This is the case because these methods can not find the sparse and low-rank representation of the samples when the data contains random rotations. In contrast, our non-linear projection learns the rotation mapping directly from the data. Figure 4 displays the transformations learned by our method on the $mnist\text{-}rot$ dataset. Each subplot of Figure 4 corresponds to a row of the matrix $\mathbf{P} = \mathbf{\Psi}^T\mathbf{Y}^T$. They have a strong similarity to circular harmonic functions, thus, can capture more rotational invariant features. These transformations make a good sense given that the dataset consists of a lot of variations along the circular direction.

Dataset	SSC	LRR	LS3C	NLS3C
(a)	67.75	75.48	67.62	$\mathbf{66.49}$
(b)	74.31	80.06	74.30	$\mathbf{72.38}$
(c)	58.59	77.75	58.68	$\mathbf{54.63}$
Avg. Time (sec)	13.86	16.20	$\mathbf{13.06}$	13.10

Table 3. Average clustering errors on the rotated MNIST datasets: (a) $mnist\text{-}rot$, (b) $mnist\text{-}rot\text{-}back\text{-}image$, (c) $mnist\text{-}back\text{-}rand$.

Two most computationally heavy steps of SSC and our method are the computation of sparse coefficients and spectral clustering. Since the sparse coefficients are found in the low-dimensional space, computation of sparse coefficients is more efficient in our case than SSC. This can be seen by comparing the average times in Table 3. On average our method takes about 13 seconds to cluster 100 digits of size 28×28, whereas SSC takes about 14 seconds. LRR is the most computationally heavy method compared to LS3C and SSC.

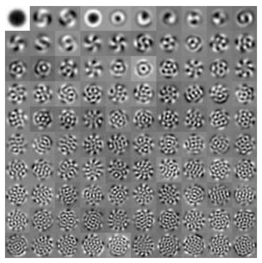

Figure 4. Example of transformations learned by the LS3C method from the rotated MNIST dataset.

6. Conclusion

We have proposed a simultaneous dimensionality reduction and sparse coding method in the low-dimensional latent space for SSC. An efficient optimization method is presented. Furthermore, the method is made non-linear so that it can deal with the small principle angle problem. Through extensive clustering experiments on several datasets, it was shown that the proposed LS3C and NLS3C methods are robust and can perform significantly better than many state-of-the-art subspace clustering methods.

Acknowledgement

The work of VMP and HVN was partially supported by an ONR grant N00014- 12-1-0124. The work of RV was partially supported by NSF grant 11-1218709.

References

[1] R. Baraniuk, M. Davenport, R. DeVore, and M. Wakin. A simple proof of the restricted isometry property for random matrices. *Constructive Approximation*, 28(3):253–263, 2008.

[2] R. Basri and D. W. Jacobs. Lambertian reflectance and linear subspaces. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(2):218–233, 2003.

[3] G. Chen and G. Lerman. Spectral curvature clustering (SCC). *International Journal of Computer Vision*, 81(3):317–330, 2009.

[4] J. P. Costeira and T. Kanade. A multibody factorization method for independently moving objects. *International Journal of Computer Vision*, 29(3):159–179, 1998.

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

[6] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2013.

[7] P. Favaro, R. Vidal, and A. Ravichandran. A closed form solution to robust subspace estimation and clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2011.

[8] 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.

[9] A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[10] T. Hastie and P. Y. Simard. Metrics and models for handwritten character recognition. *Statistical Science*, 13(1):54–65.

[11] J. Ho, M. H. Yang, J. Lim, K. Lee, and D. Kriegman. Clustering appearances of objects under varying illumination conditions. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2003.

[12] K. Kanatani. Motion segmentation by subspace separation and model selection. In *IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 586–591, 2001.

[13] H. Larochelle, D. Erhan, A. Courville, J. Bergstra, and Y. Bengio. An empirical evaluation of deep architectures on problems with many factors of variation. In *International Conference on Machine Learning*, pages 473–480. ACM, 2007.

[14] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[15] G. Liu, Z. Lin, and Y. Yu. Robust subspace segmentation by low-rank representation. In *International Conference on Machine Learning*, 2010.

[16] G. Liu and S. Yan. Latent low-rank representation for subspace segmentation and feature extraction. In *IEEE International Conference on Computer Vision*, pages 1615–1622, 2011.

[17] J. Mairal, F. Bach, and J. Ponce. Task-driven dictionary learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):791–804, 2012.

[18] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *Neural Information Processing Systems*, volume 2, pages 849–856, 2002.

[19] H. V. Nguyen, V. M. Patel, N. M. Nasrabadi, and R. Chellappa. Sparse embedding: A framework for sparsity promoting dimensionality reduction. In *European Conference on Computer Vision*, pages 414–427, Oct. 2012.

[20] Q. Qiu and G. Sapiro. Learning transformations for clustering and classification. *Preprint*, 2013.

[21] R. Rubinstein, M. Zibulevsky, and M. Elad. Double sparsity: Learning sparse dictionaries for sparse signal approximation. *IEEE Transactions on Signal Processing*, 58(3):1553–1564, 2010.

[22] A. H. Sameh and J. A. Wisniewski. A trace minimization algorithm for the generalized eigenvalue problem. *SIAM J. Numer. Anal.*, 19(6):1243–1259, 1982.

[23] M. Soltanolkotabi and E. J. Candes. A geometric analysis of subspace clustering with outliers. *Annals of Statistics*, 40(4):2195–2238, 2011.

[24] R. Tron, R. Vidal, and A. Ravichandran. A benchmark for the comparison of 3-d motion segmentation algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007.

[25] R. Vidal. Subspace clustering. *IEEE Signal Processing Magazine*, 28(2):52–68, 2011.

[26] R. Vidal and P. Favaro. Low rank subspace clustering (LRSC). *Pattern Recognition Letters*, 2013.

[27] 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.

[28] 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):210–227, 2009.

[29] 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*, page 94106, 2006.

[30] D. Zhang, M. Yang, Z. Feng, and D. Zhang. On the dimensionality reduction for sparse representation based face recognition. In *International Conference on Pattern Recognition*, pages 1237–1240, 2010.

[31] T. Zhang, A. Szlam, and G. Lerman. Median k-flats for hybrid linear modeling with many outliers. In *Workshop on Subspace Methods*, 2009.