Exploiting Unsupervised and Supervised Constraints for Subspace Clustering

Han Hu, Jianjiang Feng, *Member, IEEE,* and Jie Zhou, *Senior Member, IEEE*

Abstract—Data in many image and video analysis tasks can be viewed as points drawn from multiple low-dimensional subspaces with each subspace corresponding to one category or class. One basic task for processing such kind of data is to separate the points according to the underlying subspace, referred to as subspace clustering. Extensive studies have been made on this subject, and nearly all of them use *unconstrained subspace models*, meaning the points can be drawn from everywhere of a subspace, to represent the data. In this paper, we attempt to do subspace clustering based on a *constrained subspace assumption* that the data is further restricted in the corresponding subspaces, e.g., belonging to a submanifold or satisfying the spatial regularity constraint. This assumption usually describes the real data better, such as differently moving objects in a video scene and face images of different subjects under varying illumination. A unified integer linear programming optimization framework is used to approach subspace clustering, which can be efficiently solved by a branch-and-bound (BB) method. We also show that various kinds of supervised information, such as subspace number, outlier ratio, pairwise constraints, size prior and etc., can be conveniently incorporated into the proposed framework. Experiments on real data show that the proposed method outperforms the state-of-the-art algorithms significantly in clustering accuracy. The effectiveness of the proposed method in exploiting supervised information is also demonstrated.

Index Terms—subspace clustering, motion segmentation, face clustering, linear programming, branch and bound, constrained clustering

1 Introduction

One inherent nature of vision problems reveals that the observed data is often of high-dimension, but it usually contains low-dimensional structure which enables intelligent modeling and processing. Linear subspace perhaps heads the popularity list of such structures by vision scientists, due to its generality, efficiency and effectiveness. Many types of visual data, e.g., point trajectories of a moving object captured by an affine camera [1], images of an object under varying illumination [2], face shape/appearances of a subject under different poses [3], optical images of a same character written by different persons [4], local patches [5] or texture features [6] of pixels/superpixels belonging to the same image segment, and etc, have been empirically shown to be well-approximated by a low-dimensional linear subspace. The ubiquitous of such data in vision applications has driven the development of several techniques to find a low-dimensional representation of the original high-dimensional data, such as the well known principal component analysis (PCA), singular value decomposition (SVD) and their variants.

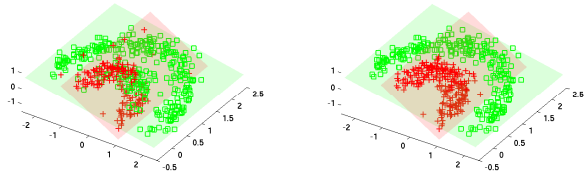
In some applications, however, the observed data would come from multiple categories thus lying on a union of subspaces. To learn from this kind of data, first of all, we may need to separate it according to the underlying subspaces, also known as subspace clustering. For example, to learn the 3D shapes and activity patterns of multiple targets in a video scene, one should first segment the scene into differently moving objects according to the underlying motion subspaces. Due to the numerous applications in computer vision and image processing, during the past two decades, subspace clustering has been extensively studied and many approaches have been proposed [7]. When the subspaces are independent and noise/outlier free, many existing algorithms are able to perfectly address the problem [8]. However, in the cases where partially dependent subspaces[1] exist, the existing algorithms often fail at least for points around the intersection of the dependent subspaces. Fig. 1 shows two toy examples where one of the state-of-the-art algorithms, Low Rank Representation (LRR) [10], fails in separating the data points especially for the ones near the intersection. This is probably because LRR, and so do most of the other existing algorithms, uses *unconstrained subspace models*, meaning the points can be drawn from everywhere of a subspace, to describe the data, which leads to the ambiguity of categorizing points at the intersection area.

In this paper, we advocate a *constrained subspace model* assuming that the data is usually further restricted. We consider two types of restriction: manifold constraint and spatial regularity constraint.

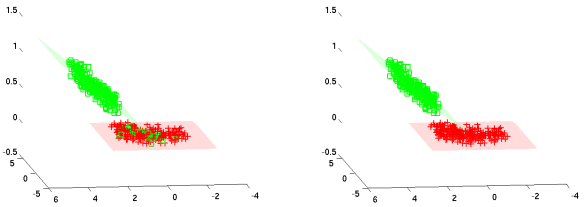
The manifold constraint says that the data is shaped not only by the subspace constraints, but may also be further restricted on a submanifold. This constraint is very common in subspace clustering applications. For instance, the problem of motion segmentation given a monocular video sequence can be formulated as a subspace clustering problem in the case of an affine camera model [11], since the image coordinates of an object could be factorized by the camera matrix and the 3D shape, leading to a linear subspace with dimension no more than 4. Apart from the linear subspace constraint, the point trajectories are further restricted to an affine subspace with dimension no more than 3. Moreover, if the camera is orthogonal, the camera matrix will be on a Stiefel manifold after registering the image coordinates to their centroid [1]. In the problem of face clustering [12], ignoring shadows, the face images of a subject with varying illumination can be approximated by the multiplication of 3 factors: the dense normal to the surface of the object, the albedo, and the lighting directions [2]. In this problem, apart from the subspace

• *Han Hu is with the Department of Automation, Tsinghua University and Baidu Research, Beijing, China. Jianjiang Feng and Jie Zhou are with the Department of Automation, Tsinghua University, Beijing 100084, China.*
E-mail: huhan02@baidu.com, {jfeng,jzhou}@tsinghua.edu.cn

1. There exists a subspace intersecting with the union of other subspaces. A formal definition of *partially dependent subspaces* can be found in [9].



(a) data points with a circle-manifold constraint.



(b) data points with spatial regularity constraint.

Fig. 1. Effect of using manifold constraint and spatial regularity constraint in subspace clustering. Left: the clustering results using LRR method [10]. Right: the clustering results using our method.

constraint, the face images should further satisfy the surface normal constraint, namely surface normals are on a specified manifold.

The spatial regularity constraint assumes that the spatially close data points usually belong to a same cluster. This is also a very common assumption made in many computer vision researches [13]. Also take motion segmentation as an example. The points from different moving objects are usually far away in their spatial locations (point trajectories), which could act as a strong prior for separating differently moving objects.

Revisit Fig. 1. In Fig. 1(a), the data points from a cluster are restricted to a submanifold $\|\mathbf{X}\|_2 = r$ (radius r unknown) on a linear subspace. The principle angle distance [14] between two subspaces is so close to each other that they can hardly be separated by *unconstrained subspace model*. But if circle-manifold model is used, the two subspaces can be correctly segmented. In Fig. 1(b), the data points from different subspaces appear in disconnected areas. By incorporating spatial regularity constraint, the points in the intersection area of the two subspaces can be correctly categorized.

Most existing methods are hard to be adapted to exploit the above constraints. In this paper, we encode them by a unified integer programming problem which can be conveniently solved by the branch-and-bound (BB) method. By encoding the manifold and spatial regularity constraints, we achieve the state-of-the-art performance in motion segmentation, face clustering and handwritten digit clustering applications. The main contributions of this work involve:

- We propose the concept of *constrained subspace model* instead of the commonly used *unconstrained subspace model* for subspace clustering. The concept is instantiated by two kinds of constraints (manifold constraint and spatial regularity constraint) and formulated by a unified integer programming problem. By exploiting such constraints, we beat the state-of-the-art algorithms on several popular benchmark data, e.g., Hopkins155 datasets [15] for motion segmentation problem, Extended Yale Face B dataset [16], [17] for face clustering and USPS dataset [18] for handwritten digit clustering.
- Supervised information often plays a key role in bridging the gap between low-level features and high-level concepts in clustering tasks [19], [20]. To the best of our knowledge, we are the first to systematically study the problem of constrained subspace clustering and we successfully encode several common types of supervised constraints, including subspace number, outlier ratio, pairwise constraints and size prior.

Our code are publicly available at https://sites.google.com/site/hanhushomepage/projects-researches.

2 Related Works

In this section, we review the existing subspace clustering methods. They can be roughly grouped into three categories: algebraic, spectral clustering based and model estimation/selection based.

Algebraic Methods. There are two kinds of well known algebraic methods worth mentioned: factorization based method and Generalized Principal Component Analysis (GPCA). Factorization based method and its variants [11], [21] are almost the only choice for subspace clustering in the early researches. They are based on the observation that for two points from two independent subspaces, the corresponding entry in the shape interaction matrix [11] is zero when the data is noise free. However, although some enhancing techniques have been proposed [22], [23], generally the performance of these algorithms drops quickly in the presence of noise, degeneracy, or partially dependent subspaces. The GPCA method [24] is another direction which fits a polynomial model to the data points. It gains much concern for its elegant formulation and the ability to handle degenerated and partially dependent subspaces. However, the complexity and trajectories required for this method increase dramatically when the number and dimension of subspaces increase, which significantly limits the application of this method.

Spectral Clustering based Methods. Inspired by the success of spectral clustering method [25] for the general clustering problem, a lot of spectral clustering based methods have been proposed to address the more specific subject, subspace clustering [26], [27], [9], [28], [29], [28], [12], [30], [8], [31]. The main differences among these methods lie in the way they build the similarity matrix. There are mainly three fashions. The first one is to compute similarity matrix directly from algebraic methods [26], [9]. Some other methods form similarity matrix by defining a point-to-subspace or subspace-to-subspace distance metric [27], [28], [32]. More recently, more works exploit the self-reconstruction properties to compute similarities [29], [12], [6], [30], [33], [8], [34], [35], [31]. To the present, the spectral clustering based methods achieve the state-of-the-art performance on several benchmark datasets, e.g., Hopkins155 [36] and Extended Yale Face B [37]. The success is partly due to the powerfulness and adaptiveness of the spectral clustering method but also partly because some tricks used for datasets and parameters tuning [38], [39].

Model Estimation/Selection based Methods. There are also some methods which address subspace clustering problem by explicitly estimating subspace models and assigning data points to them. The two popular methods for general mixture model estimation, i.e. Random Sample Consensus (RANSAC) [40] and Expectation Maximization (EM), have also been used for mixture

subspace estimation, e.g., Multi-stage Learning (MSL) [41]. Recent progress mainly focuses on developing more robust model estimation methods, e.g., Median K-flats [42], Ordered Residual Kernel (ORK) [43], Generalized Projection based M-Estimation (GpbM) [44], and etc, and more robust model selection methods, e.g., Agglomerative Lossy Compression (ALC) [45], Uncapacitated Facility Location [46], [47], [48]. Our framework is of this type and our main contribution is introducing various unsupervised and supervised constraints into the model selection framework, which is demonstrated to beat the state-of-the-art methods on three popular applications of subspace clustering: motion segmentation, face clustering and handwritten digit clustering.

3 Integer Programming for Subspace Clustering

3.1 Subspace Clustering Problem

Given a set of D-dimensional data samples $X \in \mathbb{R}^{D\times N}$ drawn from a union of K subspaces $\{S_j\}_{j=1}^K$ with the dimension of S_j be $\mathbf{r}_j$, the goal of subspace clustering is to recover the K subspaces $\{S_j\}_{j=1}^K$ and to find the relationship between X and $\{S_j\}_{j=1}^K$.

3.2 Mixture of Subspaces

A linear subspace S with dimension r can be represented by a column orthogonal matrix $U \in \mathbb{R}^{D\times r}$. Denote the distance from a data sample $\mathbf{x}$ to a subspace model S by $d(\mathbf{x}, S)$, where one popular choice is the L_2-Hausdorff distance[14]

$$d_H(\mathbf{x}, S) = \min_{\mathbf{s}\in S} \|\mathbf{x}-\mathbf{s}\|_2 = \|U_\perp \mathbf{x}\|_2, \tag{1}$$

with $U_\perp = I - UU^T$ representing the orthogonal complement space of S in $\mathbb{R}^D$ space. Then a data point $\mathbf{x}$ belonging to a subspace S satisfies: $d(\mathbf{x}, S) = 0$.

For a set of noise free data points X drawn from a union of K subspaces $\{S_j\}_{j=1}^K$, given the relationship between X and $\{S_j\}_{j=1}^K$, $L \in \{0,1\}^{N\times K}$ with $L_{ij} = 1$ indicating the i^{th} point belongs to the j^{th} subspace, and $L_{ij} = 0$ otherwise, we have

$$\begin{aligned} &\sum_{j=1}^K L_{ij} d(\mathbf{x}_i, S_j) = 0, \\ &s.t. \ \sum_{j=1}^K L_{ij} = 1, L_{ij} \in \{0,1\}. \end{aligned} \tag{2}$$

3.3 Integer Programming Formulation

In reality, the data points cannot strictly lie on the corresponding subspaces, and we thus formulate the assignment problem as

$$\begin{aligned} &\min_{S,L,K} \sum_{j=1}^K L_{ij} d(\mathbf{x}_i, S_j), \\ &s.t. \ \sum_{j=1}^K L_{ij} = 1, L_{ij} \in \{0,1\}. \end{aligned} \tag{3}$$

Considering only the data fitting errors as in eq. (3) would lead to overfitting, since higher dimensional subspaces or more subspaces always have lower cost in eq. (3). Hence, an additional model complexity term $\mathcal{P}_j$ is usually preferred, resulting in a model selection framework as

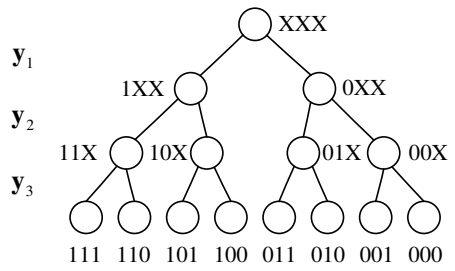


Fig. 2. A solution tree with 3 candidate subspaces.

$$\begin{aligned} \min_{L,\mathbf{y},S,K} \quad & \sum_{i=1}^N \sum_{j=1}^M L_{ij} d(\mathbf{x}_i, S_j) + \alpha \sum_{j=1}^M \mathcal{P}_j \mathbf{y}_j \\ s.t. \quad & \sum_{j=1}^M L_{ij} = 1, \forall i; \mathbf{y}_j = \max\nolimits_{1\le i\le N}\{L_{ij}\}, \forall j; \\ & L \in \{0,1\}^{N\times M}. \end{aligned} \tag{4}$$

Supposing that somehow we have already obtained a list of candidate subspace models $\{S_1, \ldots, S_M\}$, and the K true subspaces are contained in the list, we can eliminate the optimization variables S and K in eq. (4) and the cost function becomes linear to $L, \mathbf{y}$. Furthermore, the nonlinear constraints $\mathbf{y}_j = \max_{1\le i\le N}\{L_{ij}\}, j = 1, \ldots, M$ can also be converted into several linear ones:

$$L_{ij} \le \mathbf{y}_j, \forall i, j. \tag{5}$$

As a result, we get a binary-integer linear programming problem as

$$\begin{aligned} \min_{L,\mathbf{y}} \quad & \sum_{i=1}^N \sum_{j=1}^M L_{ij} d(\mathbf{x}_i, S_j) + \alpha \sum_{j=1}^M \mathcal{P}_j \mathbf{y}_j \\ s.t. \quad & \sum_{j=1}^M L_{ij} = 1, \forall i; L_{ij} \le \mathbf{y}_j, \forall i, j; \\ & L \in \{0,1\}^{N\times M}, \mathbf{y} \in \{0,1\}^{M\times 1}. \end{aligned} \tag{6}$$

One can find that eq. (6) is actually the classical uncapacited facility location problem [49], which has been long studied in the past fifty years. In this paper, we use the popular branch-and-bound method [50], [51], [52] for solving (7), which can in fact achieve the global minimum of the objective function. In the following, we first present the branch-and-bound method used for the basic problem (6). Then we show that this method can be conveniently extended to the problems with additional costs and constraints.

3.4 Branch-and-Bound Optimization

The components of the branch-and-bound method for solving eq. (6) are:

- *Solution Tree and Branching*. Observing that given $\mathbf{y}$ fixed, the optimal values of variables L can be easily found, in this paper, we use a solution tree related to only variables $\mathbf{y}$ (see Fig. 2 for an example). Each node T represents a solution set, with each character indicating the state of a candidate. State "0" means the corresponding candidate abandoned; "1" represents the candidate adopted; and "X" stands for the candidate undetermined. For each node, one of the remaining undetermined candidate subspaces will be selected as the branching factor.

- *Upper Bound*. Denote $\mathbf{y}^{ub}$ as the solution where all "X" characters equal 0. We set the upper bound at node T as the optimal cost when $\mathbf{y} = \mathbf{y}^{ub}$. The optimal subspace candidate for point $\mathbf{x}_i$ is

$$\arg \min_{\{j:\mathbf{y}_j^{ub}=1\}} d(\mathbf{x}_i,\mathrm{S}_j), \tag{7}$$

and the optimal cost is

$$\mathcal{J}_U(T) = \sum_{i=1}^{N} \min_{\{j:\mathbf{y}_j^{ub}=1\}} d(\mathbf{x}_i,\mathrm{S}_j) + \alpha \sum_{j=1}^{M} \mathcal{P}_j \mathbf{y}_j^{ub}. \tag{8}$$

When we branch right, the upper bound will remain the same and we do not need to recompute it.

- *Lower Bound*. A popular way to compute the lower bound is to relax the integer constraints and solve the relaxed LP problem [53]. However, the computational cost would be high. We use a more efficient way to compute the lower bound: separate the cost function into the data fitting part and the model penalty part, and set the lower bound as the summation of minimal values of the two parts which are independently optimized. The minimum of the first part is reached when all undetermined candidate models are adopted (we denote the corresponding indicator vector by $\mathbf{y}^{lb}$). The minimum of the second part is obtained when all undetermined candidate models are abandoned ($\mathbf{y} = \mathbf{y}^{ub}$). Hence we get a lower bound as

$$\mathcal{J}_L(T) = \sum_{i=1}^{N} \min_{\{j:\mathbf{y}_j^{lb}=1\}} d(x_i, S_j) + \alpha \sum_{j=1}^{M} \mathcal{P}_j \mathbf{y}_j^{ub}. \tag{9}$$

When we branch left, the first part of the lower bound will remain the same and we only need to add a cost $\alpha\mathcal{P}_j$ to the parent one.

The branch-and-bound method for subspace clustering is summarized in Algorithm 1. It is trivial to check that: 1) $\mathcal{J}_L(T) \leq \mathcal{J}^*(T) \leq \mathcal{J}_U(T)$, where $\mathcal{J}^*(T)$ is the optimal solution at node T; and 2) when T is a leaf node, $\mathcal{J}_U(T) = \mathcal{J}^*(T)$. According to [54], it is guaranteed that Algorithm 1 achieves global optimization.

Algorithm 1 Branch-and-Bound (BB) Method for Subspace Clustering

Require: Data points $X \in \mathbb{R}^{D\times N}$, a penalty parameter α

Generate a set of M candidate subspace models $\{S_1, \ldots, S_M\}$ by a certain scheme, e.g., RANSAC and over-segmentation (see Section 6.2.3 for details),
compute the normalized distance $\hat{d}(\mathbf{x}_i, S_j)$ between each point $\mathbf{x}_i$ and each subspace model S_j,
initialize Ψ as a priority queue with only one element $(T^0, \mathcal{J}_U(T^0), \mathcal{J}_L(T^0))$ (J_U is used as the "priority" measure), where T^0 is the root node with all candidate models undetermined (state "X"); $\mathcal{J}_U(T^0) = \infty$ and $\mathcal{J}_L(T^0)$ are the upper bound and lower bound on T^0, respectively. Set the initial optimal value as $\mathcal{J}^* = \infty$.
repeat
  retrieve the top element $(T^p, \mathcal{J}_U(T^p), \mathcal{J}_L(T^p))$ from Ψ by checking the lowest $\mathcal{J}_U$,
  if there exist undetermined candidates, branch the node T^p to obtain two child nodes T^{cl} and T^{cr} according to the fetched undetermined candidate,
  for two child nodes T^cs **do**
    compute the upper and lower bounds $\mathcal{J}_U(T^c)$ (with solution L^{ub}) and $\mathcal{J}_L(T^c)$,
    if $\mathcal{J}_U(T^c) < \mathcal{J}^*$, set $\mathcal{J}^* = \mathcal{J}_U(T^c)$, $\mathbf{y}^* = \mathbf{y}^{ub}$, $L^* = L^{ub}$,
    if $\mathcal{J}_L(T^c) < \mathcal{J}^*$, push $(T^c, \mathcal{J}_U(T^c), \mathcal{J}_L(T^c))$ into Ψ,
  end for
  pop the element $(T^p, \mathcal{J}_U(T^p), \mathcal{J}_L(T^p))$ from Ψ,
until no elements in Ψ
return $\mathbf{y}^*$, L^*

3.5 Adaption to Additional Constraints and Costs

Additional priors (such as cluster number, outlier ratio, pairwise constraint and cluster size) could be encoded as either constraints or costs into the integer linear programming framework. In Section 3.4, we have presented an efficient branch-and-bound method to solve the framework. In this section, we will show that when the additional constraints are linear in L and the additional cost is a summation of a linear function w.r.t L and a linear function w.r.t $\mathbf{y}$, we can always solve the problem by the branch-and-bound method.

Denote the additional costs by $\mathcal{C}(L)$ and $\mathcal{C}(\mathbf{y})$, and the two sets of additional constraints by $\Omega(L, \mathbf{y})$ and $\Theta(\mathbf{y})$. We can adapt the branch-and-bound method presented in Section 3.4 to the new constrained problems as follows.

- *Branching and Bounding Strategies*. If $\Theta(\mathbf{y}) \neq \emptyset$, we may need to modify the branching and bounding strategies. The solution tree remains the same. When a node is reached, we verify whether it is a feasible solution (all "X" characters are replaced by 0), and the upper bounds are computed only for feasible nodes. Since only a verifying step is needed only for each node, the constraints $\Theta(\mathbf{y})$ can be in arbitrary form.

- *Upper Bound*. The upper bound at a node is set as the optimal cost when $\mathbf{y} = \mathbf{y}^{ub}$. Thus we get an optimization problem related to only L as

$$\begin{aligned} \min_{L} \quad & \sum_{i=1}^{N}\sum_{j=1}^{M} L_{ij} d(\mathbf{x}_i, S_j) + \alpha \sum_{j=1}^{M} \mathcal{P}_j \mathbf{y}_j^{ub} + \mathcal{C}(L) + \mathcal{C}(\mathbf{y}^{ub}) \\ s.t. \quad & \sum_{j=1}^{M} L_{ij} = 1, \forall i; \Omega(L, \mathbf{y}^{ub}); L \in \{0,1\}^{N\times M}. \end{aligned} \tag{10}$$

Eq. (10) is an integer linear programming problem. We will show later that for some specific $\mathcal{C}$s and Ωs, there exist efficient algorithms to solve it. An alternative but more general way is to relax the binary-integer constraints of L as

$$0 \leq \widehat{L}_{ij} \leq 1, \forall i, j. \tag{11}$$

After relaxation, eq. (10) becomes a linear programming problem. It can be solved in polynomial time. However, the relaxed solution may contain non-binary values, and a rounding step may be needed to convert $\widehat{L}$ into $\{0,1\}$. Several schemes can be chosen for this purpose [53]. One choice is: let $L_{ij} = 1$ if $j = \arg\max_{k=1,\ldots,M} \{\widehat{L}_{ik}\}$; and $L_{ij} = 0$ otherwise.

- *Lower Bound*. The objective function can be separated into two parts $\mathcal{O}(L)$ and $\mathcal{O}(\mathbf{y})$. The lower bound is set as the summation of lower bounds of the two parts. Let $\Omega(L)$ be the maximal subset of $\Omega(L, \mathbf{y})$ relating to only L. Then the minimum of eq. (12) is a lower bound of the minimum of

the first part:

$$\begin{aligned} \min_{L} \quad & \mathcal{O}(L) \\ s.t. \quad & \sum_{j=1}^{M} L_{ij} = 1, \forall i; \Omega(L); L \in \{0,1\}^{N \times M}. \end{aligned} \tag{12}$$

Since $\mathcal{O}(\mathbf{y})$ is linear, the minimum of the second part is obtained when all undetermined candidate models with negative cost coefficients are adopted while others abandoned.

4 Exploiting Unsupervised Constraints for Subspace Clustering

In Section 3, we use an *unconstrained subspace model* for subspace clustering. As stated in Section 1, for many real subspace clustering applications, a *constrained subspace model* which assumes that the points are further restricted, is usually more reasonable. Such restrictions usually come from the nature of an application and hence are unsupervised.

In this section, we consider two types of unsupervised constraints: manifold constraints and spatial regularity constraints.

4.1 Manifold Constraints

The manifold constraints are encoded by defining manifold distance metrics replacing eq. (1) used in Section 3.2. Here, we take two popular applications as examples: motion segmentation and face clustering.

4.1.1 Motion Segmentation

Using different camera models, the manifold constraints are different. We consider two linear camera models: affine camera and orthogonal camera.

Let $\{\mathbf{x}_{fi} = (u_{fi}, v_{fi})^T \in \mathbb{R}^2\}_{f=1,\ldots,F}^{i=1,\ldots,N}$ be the 2D projections in F frames of N 3D homogeneous points $\{\mathbf{Z}_i \in \mathbb{R}^4\}_{i=1}^{N}$ from a rigid structure. Under the affine camera model, the trajectories and their 3D points satisfy the following equalities [55],

$$\mathbf{x}_{fi} = A_f \mathbf{Z}_i, \text{ and} \tag{13}$$

$$X = \overbrace{\begin{bmatrix} \mathbf{x}_{11} & \ldots & \mathbf{x}_{1N} \\ \vdots & \ddots & \vdots \\ \mathbf{x}_{F1} & \ldots & \mathbf{x}_{FN} \end{bmatrix}}^{2F \times N} = \overbrace{\begin{bmatrix} A_1 \\ \vdots \\ A_F \end{bmatrix}}^{2F \times 4} \overbrace{\begin{bmatrix} \mathbf{Z}_1^T \\ \vdots \\ \mathbf{Z}_N^T \end{bmatrix}^T}^{4 \times N} = M^m S^m, \tag{14}$$

where $A_f = K_f \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} R_f & \mathbf{t}_f \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbb{R}^{2\times 4}$ is an affine matrix at frame f, which depends on the camera intrinsic parameters K_f and the object pose relative to the camera $(R_f, \mathbf{t}_f)$.

- *Subspace Model*. Eq. (14) indicates that

$$rank(X) \leq 4. \tag{15}$$

Eq. (15) assumes that trajectories from the same rigid motion lie in a linear subspace of $\mathbb{R}^{2F}$ with dimension no more than 4. Given a set of trajectories X, a subspace model S can be represented by $U \in \mathbb{R}^{2F\times r}, r \leq 4$ from truncated SVD of X: $X = U\Sigma V^T$. The distance between a trajectory $\mathbf{x}$ and the subspace model is computed as eq. (1).

- *Affine Model*. Averaging the columns of X and S in eq. (14), we get

$$\overline{X} = M^m \overline{S^m}. \tag{16}$$

Then we have

$$X - \overline{X} = M^m (S^m - \overline{S^m}). \tag{17}$$

Since the last row of $S^m - \overline{S^m}$ is all-zero, the dimension of $X - \overline{X}$ will be no more than 3. Given a set of trajectories X, an affine model S can be represented by $\{U, r \leq 3, \overline{X}\}$, where $U \in \mathbb{R}^{2F\times r}$ is from the truncated SVD of $X - \overline{X}$: $X - \overline{X} = U\Sigma V^T$. The distance between a trajectory $\mathbf{x}$ and the subspace model is defined as

$$d(\mathbf{x}, S) = \|U_\perp (\mathbf{x} - \overline{X})\|_2. \tag{18}$$

- *Metric Model*. Under an orthogonal camera model, $\widehat{A}_f$, the first 3 columns of A_f, should further satisfy the metric constraints

$$\widehat{A}_f \widehat{A}_f^T = \mathbf{s}_f I, \tag{19}$$

where I is the identity matrix of size 2×2, and $\mathbf{s}_f$ is a scale factor. Thus a motion model can be represented by $\{\widehat{M^m} \in \mathbb{R}^{2F\times 3}, \overline{X}\}$, where $\widehat{M^m} = (\widehat{A}_1, \ldots, \widehat{A}_F)^T$. Given a set of trajectories X, there have been several methods to recover the motion and shape, e.g., [56], [57]. Then the distance between a trajectory $\mathbf{x}$ and the subspace model is defined as

$$d(\mathbf{x}, S) = \|(\mathbf{x} - \overline{X}) - \widehat{M^m}(\widehat{M^m}^T \widehat{M^m})^{-1} \widehat{M^m}^T (\mathbf{x} - \overline{X})\|_2. \tag{20}$$

4.1.2 Face Clustering

The image brightness given varying illumination could be approximated in a bilinear form [58], [16]. Assuming the absence of all shadows, given a set of images of a Lambertian object with varying illumination, the brightness at pixel i for the j^{th} image can be modeled as,

$$X_{ij} = \mathbf{l}_j^T \rho_i (1, \mathbf{z}_i^T)^T, \tag{21}$$

where $\mathbf{l} \in \mathbb{R}^4$ is the lighting directions; ρ is the albedo; and $\mathbf{z} \in \mathbb{R}^3$ is the dense normal to the surface of the object. Written in matrix form, we get

$$X = \begin{bmatrix} \rho_1 \begin{bmatrix} 1 & \mathbf{z}_1^T \end{bmatrix} \\ \vdots \\ \rho_D \begin{bmatrix} 1 & \mathbf{z}_D^T \end{bmatrix} \end{bmatrix} \begin{bmatrix} \mathbf{l}_1 & \cdots & \mathbf{l}_N \end{bmatrix} = S^f M^f. \tag{22}$$

- *Rank-4 Subspace Model*. Eq. (22) indicates that X lies on a linear subspace with dimension no more than 4.
- *Metric Model*. $\mathbf{z}$ further satisfies constraints

$$\mathbf{z} \cdot \mathbf{z}^T = 1, \tag{23}$$

and several methods have been proposed to recover the shape $\widehat{S^f}$ from X, e.g., [57]. We could define the distance between an image instance $\mathbf{x}$ and the manifold model as

$$d(\mathbf{x}, S) = \|\mathbf{x} - \widehat{S^f}(\widehat{S^f}^T \widehat{S^f})^{-1} \widehat{S^f}^T \mathbf{x}\|_2. \tag{24}$$

- *Rank-9 Subspace Model*. In the presence of both attached and cast shadows, the above models will be inaccurate. Lee et al. [17] have argued that the shadows can be approximately counted by a rank-9 subspace model. We will try this kind of model as well.

Optimization. The manifold constraint affects only the computations of eq. (1) and eq. (39). Other steps in Section 3.4 remain the same.

4.2 Spatial Regularity Constraints

The spatial regularity constraints penalize the changes of cluster labels between spatially close points. We encode this constraint by adding a penalty term $\mathcal{J}_{\text{sp}}$ to the cost function of eq. (6) as

$$\mathcal{J}_{\text{sp}} = \sum_{i=1}^{N} \sum_{j \in \mathcal{N}(i)}^{N} \beta \sum_{k=1}^{M} |L_{ik} - L_{jk}|, \tag{25}$$

where $\mathcal{N}(i)$ indicates the neighboring set of point i (by Euclidean distance of feature vectors); β is the cost of two points belonging to different clusters.

There are absolute operators in $\mathcal{J}_{sp}$. To avoid them, we replace each $\sum_{k=1}^{M} |L_{ik} - L_{jk}|$ as follows,

$$\begin{aligned} &\sum_{k=1}^{M} |L_{ik} - L_{jk}| = 1 - \sum_{k=1}^{M} L_{ik,jk}, \\ &\sum_{k=1}^{M} L_{ik,jm} = L_{jm}, m = 1, \ldots, M, \\ &\sum_{m=1}^{M} L_{ik,jm} = L_{ik}, k = 1, \ldots, M, L_{ik,jm} \in \{0, 1\}, \end{aligned} \tag{26}$$

where $L_{ik,jm}$ is an auxiliary binary variable indicating whether simultaneously point i belongs to candidate model k and point j belongs to candidate model m.

Optimization. There are $N_{\text{sp}}M^2$ additional auxiliary variables and $2N_{\text{sp}}M$ additional equality constraints, where N_{sp} is the number of neighboring pairs. In addition, the coefficient matrix in the constraints is very sparse, resulting in affordable time and space complexity when using the general method presented in Section 3.5. When there are no other constraints, e.g. the ones described in Section 5.2 and Section 5.4, the problems of eq. (10) and eq. (12) are actually Markov Random Field (MRF) problems. As a result, we can utilize more efficient algorithms, e.g., the Primal-Dual solver introduced in [59].

5 Exploiting Supervised Constraints for Subspace Clustering

Subspace clustering is hard to be perfectly solved due to noises, outliers, partially dependent subspaces, and semantic gap between low-level observations and high-level concepts. In the more general unsupervised learning domain, it has been shown that high-level supervised information is a key road to correct the errors [19]. In this section, we will encode several common types of supervised constraints.

5.1 Subspace Number Priors

Although the previous framework is capable of automatically determining the number of subspacesclusters, it benefits from the prior knowledge of subspace number.

Our framework can conveniently incorporate various priors about subspace number by adding the following constraint to the proposed framework:

$$\sum_{j=1}^{M} \mathbf{y}_j < (\leq, =, \geq, \text{or} >) K, \tag{27}$$

where K is a given prior on the subspace number.

Optimization. Only the *branching and bounding strategies* component in Section 3.5 needs to be modified when subspace number priors are added. Apart from the strategies in Section 3.5, some other strategies are incorporated to further reduce the computational cost:

- *the case of "$<$" (or "$\leq$").* The descendants of the node where the number of adopted candidate models is $K-1$ (or K) are removed from the original solution tree;
- *the case of "$=$".* The descendants of the node where the number of adopted candidate models is K or the number of abandoned candidate models is $M-K$ are removed from the original solution tree. The model penalty part of the lower bound is set as summation of the minimal K model penalties including the adopted ones;
- *the case of "$>$" (or "$\geq$").* The descendants of the node where the number of abandoned candidate models is $M-K-1$ (or $M-K$) are removed from the original solution tree. The model penalty part of the lower bound is set as summation of the minimal K model penalties including the adopted ones.

5.2 Outlier Ratio Priors

In reality, the obtained data is usually contaminated with noises and errors. We will show in this subsection that the proposed framework could be easily adapted to outlying data.

We add a *virtual* subspace model S^o into the candidate list, which is supposed to cover the outliers. All the points has equal costs to this virtual model: $d(\mathbf{x}_i, S^o) \equiv C_o$. One can decrease or increase C_o to find more or less outliers.

The outliers finding process can be more accurate when the outlier ratio is known as a priori. We encode the outlier ratio priors by adding a new constraint to the proposed framework:

$$\sum_{i=1}^{N} \mathbf{o}_i \leq \gamma N, \tag{28}$$

where $\mathbf{o}_i$ is an indicator specifying whether point i is an outlier, and γ indicates the maximal outlier ratio. Then we get an optimization problem as

$$\begin{aligned} \min_{L, \mathbf{y}, \mathbf{o}} \quad & \sum_{i=1}^{N} (\sum_{j=1}^{M} L_{ij} d(\mathbf{x}_i, S_j) + \mathbf{o}_i d(\mathbf{x}_i, S^o) + \alpha \sum_{j=1}^{M} \mathcal{P}_j \mathbf{y}_j \\ s.t. \quad & \sum_{j=1}^{M} L_{ij} + \mathbf{o}_i = 1, \forall i; L_{ij} \leq \mathbf{y}_j, \forall i, j; \sum_{i=1}^{N} \mathbf{o}_i \leq \gamma N; \\ & L \in \{0, 1\}^{N \times M}, \mathbf{o} \in \{0, 1\}^{N \times 1}, \mathbf{y} \in \{0, 1\}^{M \times 1}. \end{aligned} \tag{29}$$

With prior in eq. (28), we can set $d(\mathbf{x}_i, S^o)$ as a small value near *infenitesimal*, i.e., -10^4.

Optimization. The number of additional variables and constraints are N and 1, respectively. By replacing L with an augmented matrix $[L \ \ \mathbf{o}]$, the method in Section 3.5 can be used to solve the new problem. When optimizing eq. (10) and eq. (12), there exists a more efficient algorithm: all points are firstly categorized to the subspace with minimum fitting cost, and then γN points with maximal minimum fitting costs are re-categorized as outliers.

5.3 Pairwise Constraints

Pairwise constraint is one of the most popular supervised information used for constrained clustering [19], [20]. It specifies whether two points belong to a same cluster or not, referred to as must-link constraint and cannot-link constraint respectively. We can encode the must-link and cannot-link constraints between point i and j by introducing linear equalities eq. (30) and eq. (31), respectively.

$$L_{ik} - L_{jk} = 0, k = 1, \ldots, M. \tag{30}$$

$$\sum_{k=1}^{M} |L_{ik} - L_{jk}| = 1. \tag{31}$$

The equality constraints are equivalent to adding the following additional costs,

$$\mathcal{J}_{pc} = \sum_{i,j:(i,j)\in\mathcal{PL}} \gamma_{(i,j)} \sum_{k=1}^{M} |L_{ik} - L_{jk}|, \tag{32}$$

where $\mathcal{PL}$ is the set of point pairs with pairwise priors; $\gamma_{(i,j)}$ specifies the degree of penalty on constraint (i, j) and we set $\gamma_{(i,j)} = 10^4$ when (i, j) is a must-link constraint and $\gamma_{(i,j)} = -10^4$ when (i, j) is a cannot-link one.

In practice, it has been pointed out that enforcing only sparse pairwise constraints usually results in unsmooth solutions [19]. Hence, we advocate an additional regularity term to enforce the smoothness of the solution, e.g., the spatial regularity constraints presented in Section 4.2 or the regularity defined by other similarity matrices [29], [12].

Optimization. There are $N_{\text{pc}}M^2$ additional auxiliary variables and $2N_{\text{pc}}M$ equality constraints, where N_{pc} is the number of pairwise constraints. One can find that eq. (32) has the same form as the spatial regularity cost in eq. (25), and hence eq. (10) and eq. (12) are also MRF problems. When there exist cannot-link constraints, the corresponding pairwise terms become non-submodular [60], and some solvers are available to optimize the problems, e.g., Quadratic Pseudo-Boolean Optimization with Probing (QPBOP) [60].

5.4 Size Priors

Another significant supervised constraint is the size prior [61]. Possible size priors include: (1) the sizes of all clusters are no greater (or no smaller) than Z; (2) the size of cluster containing point i is no greater (or no smaller) than Z; (3) the size of cluster containing point i is no greater (or no smaller) than the size of cluster containing point j at a multiple of Z.

We encode the first type of priors by a set of inequality constraints linear in L. For "no greater" case, the constraints are

$$\mathbf{y}_k \sum_{i=1}^{N} L_{ik} \leq Z, k = 1, \ldots, M. \tag{33}$$

For "no smaller" case, the constraints are

$$\sum_{i=1}^{N} L_{ik} \geq Z\mathbf{y}_k, k = 1, \ldots, M. \tag{34}$$

The size of a cluster related to point i could be calculated as,

$$\sum_{j=1}^{N}(1 - \sum_{k=1}^{M} |L_{jk} - L_{ik}|). \tag{35}$$

Therefore, the second and third type of priors could be encoded by

$$\sum_{j=1}^{N}(1 - \sum_{k=1}^{M} |L_{jk} - L_{ik}|) \leq (\geq)Z, \tag{36}$$

and

$$\sum_{p=1}^{N}(1 - \sum_{k=1}^{M} |L_{pk} - L_{ik}|) \leq (\geq)Z \sum_{q=1}^{N}(1 - \sum_{m=1}^{M} |L_{qm} - L_{jm}|), \tag{37}$$

respectively, which can be further converted into linear constraints by the method presented in Section 5.3.

Optimization. Since all the introduced constraints are linear in L and the auxiliary variables, the method in Section 3.5 can be used to solve the new optimization problem. For the three types of priors, the numbers of additional auxiliary variables are 0, NM^2 and $2NM^2$, respectively. The number of additional constraints are M, $2MN + 1$, and $4MN + 1$, respectively. Eq. (10) and eq. (12) are optimized by the standard linear programming solver, e.g., MOSEK [62].

6 Experiments

In this section, we evaluate the proposed method (referred to as B-B) on three real-world applications of subspace clustering: motion segmentation, face clustering and handwritten digit clustering. Apart from comparing with the state-of-the-art methods on accuracy and efficiency, we also show the effectiveness of applying the proposed BB method to several supervised constraints.

6.1 Experimental Data and Evaluation Metrics

6.1.1 Experimental Data

We use three datasets for experiments: Hopkins155 [15], Extended Yale Face B [16], [17] and USPS [18], which are the most popular benchmark datasets used in literatures for evaluating subspace clustering algorithms [7].

Hopkins155 [15] is a motion segmentation dataset, composed by 155 video sequences with extracted feature points and their tracks across frames. Each video has 2 or 3 motions and the motions may be degenerate or partially dependent with each other, which act as big challenges for segmentation algorithms. Some sample videos with trajectories on them are shown in Fig. 3(a).

Extended Yale Face B [16], [17] is a face clustering dataset, which consists of 192×168 pixel cropped face images under varying poses and illuminations from 38 human subjects. We use all the 64 frontal face images per subject for experiment, and resize the images to 48×42 for efficiency. Note that We use the cropped images instead of original ones to ease the effect of backgrounds, to avoid overestimating the performance of clustering algorithms [7]. Fig. 3(b) shows some sample images of the dataset.

USPS [18] is a handwritten digit dataset of 9298 images, with each image having 16×16 pixels. We use the first 100 images of each digit for experiments. Fig. 3(c) shows some sample digit images.

6.1.2 Evaluation Metrics

We evaluate the proposed algorithms using clustering accuracy and efficiency. The same as in most literatures, we use clustering error (*CE*) to measure the accuracy [15], [7]:

$$CE = 1 - \frac{1}{N} \sum_{i=1}^{N} \delta(p_i, map(q_i)), \tag{38}$$

where q_i, p_i represent the output label and the ground truth one of the ith point; $\delta(x, y) = 1$ if $x = y$, and $\delta(x, y) = 0$ otherwise; $map(q_i)$ is the best mapping function that permutes clustering labels to match the ground truth labels and can be computed by the Kuhn-Munkres algorithm [63].

Fig. 3. Samples from the three benchmark datasets, Hopkins155 (top), Extended Yale Face B (middle) and USPS (bottom).

6.2 Details of Our Method

In the following, we describe the details of applying the proposed method to motion segmentation, face clustering and handwritten digit clustering problems, including distance measure, exploited unsupervised constraints, model generation and penalty terms.

6.2.1 Distance Measure

To make the point-to-model distances of candidate models with different ranks and point lengths defined in a same scale, we do not directly use the popular L_2-Haussdorf distance as in eq. (1), but use the normalized distance by introducing the rank-depended noise level of candidate models, $\sigma_{(\mathbf{r}_j)}$,

$$d_N(\mathbf{x}_i, S_j) = -\ln \frac{1}{\sigma_{(\mathbf{r}_j)}} \exp(-\frac{d_H^2(\mathbf{x}_i, S_j)}{2\sigma_{(\mathbf{r}_j)}^2}) = \frac{d_H^2(\mathbf{x}_i, S_j)}{2\sigma_{(\mathbf{r}_j)}^2} + \ln \sigma_{(\mathbf{r}_j)} \tag{39}$$

where $\sigma_{(\mathbf{r}_j)}$ is estimated in the following way: first the standard deviation of each candidate model is estimated according to the supported points, and then for each subspace rank, the median of the K minimal deviations is selected as the estimated noise level.

The Gaussian normalization not only makes the point-to-model distances of candidate models with different ranks defined in the same scale, which benefits the subspace clustering, but it also normalizes the distances of different subspace clustering instances into a similar scale, such that the parameter settings across different instances can be shared.

6.2.2 Exploited unsupervised constraints

For motion segmentation problem, both manifold constraints and spatial regularity constraints are incorporated. For face clustering problem, there are no improvements by introducing spatial regularity constraints and hence we exploit only the manifold constraints. For handwritten digit clustering problem, we consider only spatial regularity constraints.

6.2.3 Model generation

We consider two types of methods to generate initial candidate models: over-segmentation and randomized local models (RLM). We first use the existing subspace clustering algorithms, e.g., angle similarity based clustering (ASC) and LRR [12], to achieve over-segmentation[2]. Then the points in each over-segment are used as support points to compute candidate models. Randomized local models are computed by initially sampling $M \ll N$ random points, and then forming the models around each of these points and their $(P-1)$-nearest neighbors in the ambient Euclidean distance.

We try subspace, affine and metric models for motion segmentation problem, and try rank-4 subspace, rank-9 subspace and metric models for face clustering problem. For handwritten digit clustering problem, we use subspace models with rank varying from 5 to 15. Since degenerations are common in motion segmentation, we also generate degenerated models, i.e. rank-2 affine models or rank-3 subspace models. For face clustering and handwritten digit clustering, we compute fixed-rank models. All the subspace and affine models are computed by singular value decomposition (SVD). The metric models for motion segmentation and face clustering are both computed by the methods in [57].

6.2.4 Penalty terms

We consider two types of penalties for candidate models: model complexity and model uncertainty. We use geometric Akainke information criterion (G-AIC) [64], [65] to count model complexity and use the ratio of standard variation computed by the estimated inliers, e.g. TSSE-estimator [66], $\widetilde{\sigma}$, to the one by the support points, σ, to count model uncertainty. The total penalty is:

$$\mathcal{P}_j = (\mathbf{r}_j + \frac{2\widetilde{\sigma}_j}{\sigma_j}) \cdot (N + D - \mathbf{r}_j). \tag{40}$$

6.3 Baseline Algorithms

To the present, spectral clustering based algorithms perform as the state-of-the-art. We compare our BB algorithm with these algorithms, including ASC, LSA [27], SCC [28], SBLF [32], SSC [29], [67], LRR [12], LRR-PP [10], LatLRR [68], SSQP [33], LSR [8], NLS [69] and DiSC [34]. The results of GPCA [24] is also listed.

We also compare our BB algorithm with the mixture-model based methods, including RANSAC [40], MSL [41], ALC [45], ORK [43] and GpbM [44]. Since ORK [43] and GpbM [44] cannot utilize cluster number constraints, we report clustering errors without number constraints when comparing with these two methods. For comparison with other methods, we report clustering errors with number constraints.

Furthermore, our BB algorithm can be regarded as exploiting additional nonlinear structures contained in the original subspace. Hence we also do comparison with several manifold clustering methods, e.g. LLMC [70] and SMCE [71].

For all algorithms except ASC, we report results from the corresponding literatures or by running the codes provided by the authors (the parameters are chosen by grid searching). For SSC, we use the ADMM version[3] if the corresponding number is not reported in the paper. For ASC, we use our own implementation.

2. ASC and LRR are both spectral clustering based methods. The over-segmentation is obtained by setting cluster number larger than the ground truth one in the spectral clustering step.

3. http://www.cis.jhu.edu/~ehsan/Codes/SSC_ADMM_v1.1.zip

TABLE 1
Clustering errors (*CE*) of mixture model based methods on Hopkins155 datasets with known motion number, where "ave." stands for "average", and "med." stands for "median". The parameters used by BB are: $\alpha = 0.1$; $\beta = 500$; 4-nn. These settings are also used in Table 2 and Table 4.

method	RANSAC	MSL	ALC	BB (RLM)	BB (os-LRR)
ave. (%)	9.76	5.06	3.37	2.45	**0.63**
med. (%)	3.21	0	0.49	0	0

TABLE 2
Comparison to the algebraic methods and spectral clustering based methods on Hopkins155 datasets with known motion number.

method	ASC	GPCA	LSA	SLBF
ave. (%)	26.14	10.34	4.94	1.35
med. (%)	27.24	2.54	0.90	0
method	SC	SCC	SSC	LRR
ave. (%)	1.20	2.70	1.25	4.31
med. (%)	0	0	0	0
method	LRR-PP	LatLRR	SSQP	NLS
ave. (%)	1.59	0.85	1.49	0.76
med. (%)	0	0	0	0
method	LSR	DiSC	BB (os-ASC)	BB (os-LRR)
ave. (%)	3.32	1.25	5.96	**0.63**
med. (%)	0	0	0	0

TABLE 3
Clustering errors (*CE*) on Hopkins155 datasets with unknown motion number. The parameters used by BB are: $\alpha = 0.4$; $\beta = 500$; 4-nn.

method	ORK	GpbM	BB (os-LRR)
ave. (%)	8.91	7.44	**6.09**
med. (%)	-	-	0

TABLE 4
Comparison of the BB method to the manifold clustering methods on Hopkins155 datasets using *CE*.

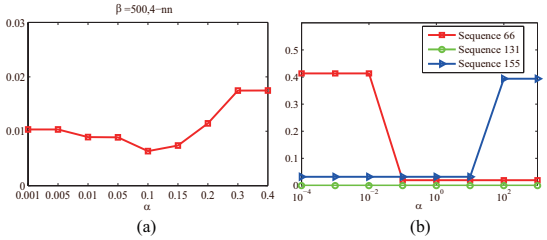
method	LLMC	SMCE	BB (os-LRR)
ave. (%)	4.87	13.34	**0.63**
med. (%)	0	9.97	0

6.4 Motion Segmentation Results

6.4.1 Parameter Settings

We use affine models to indicate motions, and generate candidate models by two kinds of strategies: randomized local models (RLM) and over-segmentation (using LRR and ASC, referred to as os-LRR and os-ASC respectively). For RLM, we generate 24 candidate models and the reported numbers are averaged over 10 trials. For os-ASC and os-LRR, we generate $K + 2$ rank-2 candidates and $K + 2$ rank-3 candidates by over-segmentation. For all the variants[4] and all the sequences in Hopkins155 dataset, we set $\alpha = 0.1$ and use the same spatial regularities: $\mathcal{N}(i)$ corresponds to the 4-nearest neighbors of point i and $\beta = 500$[5].

For all baseline algorithms except ASC and SMCE, we report the results listed in the corresponding literatures. For SMCE, we use the code provided by the authors and choose the parameters by grid searching, i.e., $k \in 2:2:50$ and $\lambda \in 2^{-10:10}$. The best parameters for SMCE are $k = 10, \lambda = 2^{-7}$.



(a) (b)

Fig. 4. The influence of parameter α on performance of motion segmentation with known motion number and constant spatial regularity as: $\mathcal{N}(i)$ corresponds to the 4-nearest neighbors of point i and $\beta = 500$. (a) plots the average *CE* for all sequences in Hopkins155 datasets; (b) shows the *CE* on three sequences: 66, 131 and 155.

6.4.2 Segmentation Accuracy

We compare the BB method in this paper to the mixture model based and the other types of methods using *CE* as listed in Table 1 and Table 2, respectively, assuming that the motion number is known as a priori. It can be seen that our methods with both model generation strategies outperform all the mixture model based ones significantly in clustering accuracy. Compared to the algebraic methods and spectral clustering based methods, our method with os-LRR still performs the best and the RLM variant is comparable to them. Note that even when the generated models are poor, e.g., using os-ASC, our method still produces good results. We also compare BB to ORK [43] and GpbM [44] which cannot utilize cluster number constraints (see Table 3). Without motion number prior, BB performs the best too.

We also compare BB to the state-of-the-art manifold clustering methods, e.g. LLMC [70] and SMCE [71] in Table 4. Similar as our spatial regularity constraints, LLMC and SMCE also consider the spatial information but encode it in a different way that spatially distant points are assigned very small or even zero affinities. BB performs much better than these methods probably because: 1) BB encodes both the manifold constraints and the global subspace structure, while LLMC and SMCE model the data purely as nonlinear manifolds; 2) LLMC and SMCE cut the connections between distant points which may break the graph connectivity of a same subspace. The BB method encodes spatial information by penalizing the discontinuity of labels, making it immune to such problem.

6.4.3 The Influence of Parameter α

The parameter α is used to balance the data fitting term and the model penalty term. When the motion number is known as a priori, this balancing influences only the selection of good candidate models; with motion number unknown, the balancing also influences the choice of motion number. In this part, we investigate only the case of known motion number. Generally

4. An exception is the experiments of Table 3 and Table 6 where the motion number is unknown. In these experiments, the parameters are: $\alpha = 0.4$, $\beta = 500$ and 4-nn.

5. When the cluster number is known as a priori, large β may lead to cluster number reduction in the final clustering. To avoid such case, we reduce β by a factor of 0.5 until the cluster number constraint is satisfied.

TABLE 5
Average clustering errors (*CE*) on Hopkins155 datasets using different models. All candidate models are generated using os-LRR strategy.

model	type	affine		subspace		metric
	rank	2 and 3	3	3 and 4	4	4
CE (%)		**0.63**	2.25	2.14	2.30	3.02

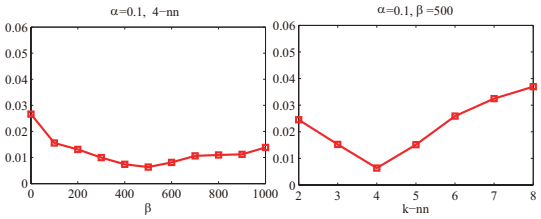


Fig. 5. The influence of spatial regularity parameters (β and $\mathcal{N}(i)$) on performance of motion segmentation with known motion number.

TABLE 6
Accuracy of motion number estimation on Hopkins155 datasets. The parameters of BB are: $\alpha = 0.4$, $\beta = 500$, 4-nn.

method	RD	ORK	KO	BB (os-LRR)
correct (%)	62.58	65.80	74.84	**80.00**

speaking, the choice of parameter α depends on the powerfulness of data fitting and prior knowledge in telling good and bad candidate models apart (see Fig. 4(b)): when the data fitting part is more powerful, smaller α is preferred, e.g., Sequence 155; when the prior knowledge part is more powerful, larger α is better, e.g., Sequence 66; when both are good, we could choose α arbitrarily, e.g., Sequence 131.

Fig. 4(a) shows the segmentation results over all 155 sequences in Hopkins155 datasets: while α ranges from 0.001 to 0.15, the segmentation errors, *CE* remain almost unchanged, slightly varying from 0.63% to 1.03%; when α increases from 0.15 to 0.4, the *CE* rise up rapidly to 1.75%. These results advocate a balanced combination of the data fitting and model penalty terms.

6.4.4 The Effect of Unsupervised Constraints

We investigate the effects of two kinds of unsupervised constraints: manifold constraint and spatial regularity constraint.

The manifold constraints take affects at the model generation stage. We compare the segmentation results using different manifold constraints, as listed in Table 5. It can be seen that the affine models work better than the original subspace models, maybe due to its more accurate representation. But the metric models perform worst. This phenomenon is probably due to the following two reasons. First, degenerated data, very common in Hopkins155 datasets, usually leads to poor metric model fitting results. Second, the depth ranges for some sequences in Hopkins155 datasets are very big, making the orthogonal assumption invalid.

To evaluate the effect of spatial regularity constraints, we draw the *CE* curves with varying β and $\mathcal{N}(i)$ as shown in Fig. 5. We observe an improvement of *CE* from 2.66% to 0.63% ($\beta = 0$ indicates that no spatial regularity is included). However, too much spatial smooth may harm the segmentation.

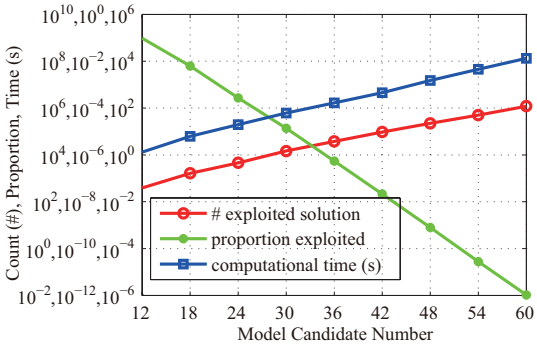


Fig. 6. The average computational time and #/proportion of exploited nodes vs. # candidate models on Hopkins155 datasets.

TABLE 7
Comparison of uncapacitated facility location methods on Hopkins155 datasets when no constraints (except motion number prior) are exploited.

method	LP Relaxation [72]	Message Passing [47]	BB
CE (%)	2.75	2.85	**2.66**
time (s)	6.81	**0.035**	0.060

6.4.5 Comparison with Other Uncapacitated Facility Location Methods

We compare the branch and bound method with the linear programming relaxation [72] and the message passing method [47][6] in order to solve the problem in (6). Table 7 shows the clustering errors and the computational time of different algorithms. It can be seen that BB produces the lowest clustering error thanks to its global optimality. The BB is more efficient than LP relaxation but less efficient than message passing method.

6.4.6 Motion Number Estimation

Our BB method can automatically determine the motion number by searching through all the solution space. Table 6 shows the results of BB on motion number estimation compared with existing methods, i.e. rank detection (RD) [65], ordered residual kernel (ORK) [43] and Kernel Optimization (KO) [73]. We correctly predict the true motion numbers on 124 sequences, by using os-LRR to generate the candidate models (the candidate number is set constant as 10) and choosing parameters: $\alpha = 0.4$; $\mathcal{N}(i)$ corresponds to the 4-nearest neighbors of point i; $\beta = 500$. It can be seen that the BB method performs the best among these methods.

6.4.7 Computational Efficiency

We ran all the experiments on a PC with a 2.53GHz CPU. The average computational time[7] as well as the number and proportion[8] of exploited nodes for each sequence as a function of the number of candidate models are shown in Fig. 6. It can be seen that the proportion of exploited nodes is very small,

6. The heuristic method in [46] is used to enforce the motion number constraints.

7. In this test, we use RLM to generate candidate models. The cluster number is unknown and the spatial regularity constraint is involved. Also note that the time of candidate model generation is excluded, because it relies on the adopted strategies.

8. The proportion is computed by $\frac{\#\text{exploited nodes}}{2^M}$, where M is the number of candidate models.

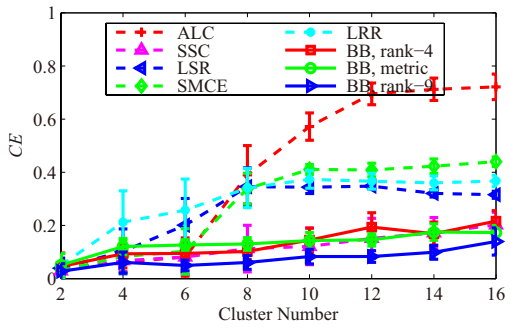


Fig. 7. The average clustering error (*CE*) with error bars of different methods on Extended Yale Face B dataset. For all variants of BB, $\alpha = 0.04$ and no spatial regularity is involved. For other methods, the parameters are chosen by grid searching.

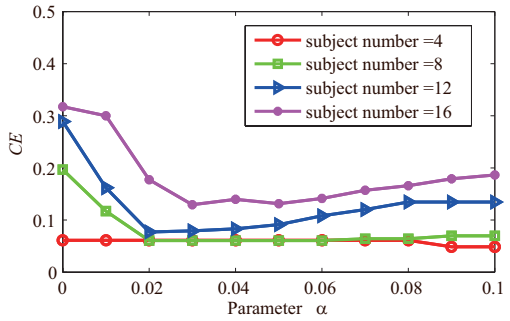


Fig. 8. The effect of parameter α on the performance of face clustering using rank-9 subspace model.

and it dramatically drops when the number of candidate models increases.

For the experiments in Table 2, "BB (os-LRR)" takes only 0.23 seconds for each sequence on average, mainly due to the reduction of searching space thanks to the known cluster number. When no spatial regularity constraints are incorporated, the average computational time is further reduced to 0.06 seconds.

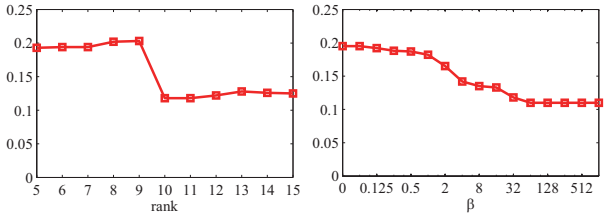
Nevertheless, it is worth noting that in our experiments, the number of candidate model is small, in which case our method has a reasonable computational time. However, as Figure 6 shows the computational complexity of our method is approximately exponential in the number of candidates models. In fact, our method becomes inefficient when the number of candidate models is more than about 50. In order to utilize our method for such cases, one possible way is to first discard bad candidates by some simpler strategies and then input the remaining candidates into our method for global model selection. Another direction is to develop approximate but more efficient method for the optimization. These will be our future directions.

6.5 Face Clustering Results

To evaluate the performance of different methods, we form 8 tasks with varying subject numbers, $\{2, 4, 6, 8, 10, 12, 14, 16\}$. Each task is repeated 10 times, containing images from randomly picked subjects. For the BB method, we generate $1.5K$ candidate models by over-segmentation of LRR [12] and try the rank-4 subspace, rank-9 subspace and metric model. The *CE* curves of different methods are shown in Fig. 7. BB with rank-9 model outperforms all the other methods, and its variants using metric

TABLE 8
Clustering errors (*CE*) and the corresponding parameters on USPS dataset. In our method, the parameters are $\alpha = 0.001$, $\beta = 32$ and $\mathcal{N}(i)$ corresponding to 4-nn of point i.

method	ALC	SSC	LRR	LSR	BB (rank-10)
CE (%)	71.50	32.10	22.60	26.10	**12.60**
para.	-	$\lambda = 2^{-6}$	$\lambda = 2^{-14}$	$\lambda = 2^{7}$	-



(a) (b)

Fig. 10. The clustering error (*CE*) on USPS dataset with varying parameters. (a) $\alpha = 0.001, \beta = 32$, 4-nn, model rank varies; (b) $\alpha = 0.001$, 10-rank models, 4-nn, β varies.

and rank-4 models achieves comparable results with the state-of-the-arts. Also note that all variants of our BB methods use LRR as an initial step to get the model candidates and they all perform much better than the pure LRR method, which demonstrates the advantages of incorporating additional unsupervised constraints by our methods, e.g., the explicit physical models and the model penalties.

We also investigate the effect of parameter α on the performance of face clustering as shown in Fig. 8. It can be seen: when subject number is small, the data fitting term alone is good enough to select good candidates; when subject number grows bigger, the prior knowledge on models will be indispensable to achieve good clustering.

The computation is also very efficient that the branch and bound optimization takes only 1.9 seconds excluding the candidate model generation step.

6.6 Handwritten Digit Clustering Results

We generate $1.5K = 15$ candidate models by over-segmentation of LRR [12]. Different from rigid motions and faces under varying illuminations where the knowledge about model rank is clear from the physical view, the handwritten digits do not have such clear physical knowledge on the model rank. Hence we try subspace models with rank from 5-15.

Table 8 lists the *CE* of the BB method compared with the state-of-the-art ones. It can be seen that the BB method outperforms the other ones significantly. The *CE* curves with varying model ranks and the spatial regularity parameter β are shown in Figure 10. We can see: 1) models with rank from 10 to 15 all work well on the USPS dataset; 2) the clustering performance is significantly improved by incorporating spatial regularities.

The branch and bound optimization takes 33.0s to get the results.

6.7 Incorporating Supervised Information

In Section 5, we have successfully encoded four types of supervised constraints: subspace number prior, outlier ratio prior, pairwise constraints and size prior.

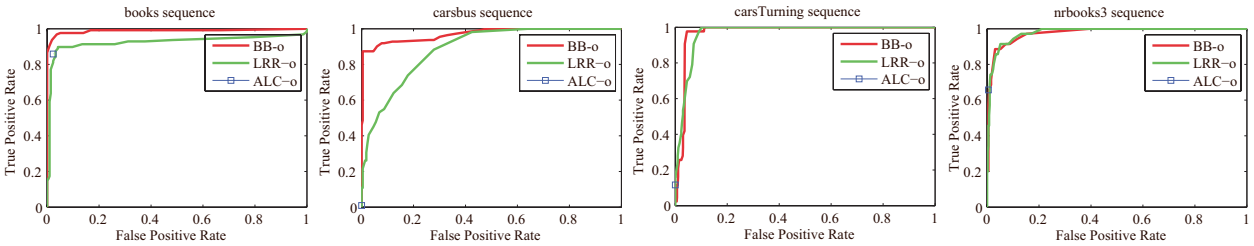


Fig. 9. ROC curves for outlier detection on four real motion sequences. Note that for ALC-o, there are no tuning parameters available to get a whole ROC curve.

TABLE 9
CE (%) on 4 motion sequences with real outliers.

seq.	ALC-o	LRR-o	BB-o
books	27.14	20.79	**0**
carbus3	0	4.32	**0**
carsTurning	10.35	41.15	**2.32**
nrbooks3	23.58	1.32	**0**

In the experiments described in Section 6.4, 6.5 and 6.6, subspace number prior has already been used. In this section, we will test the effectiveness of our algorithm in encoding three other types of supervised constraints. We use the same parameter settings as in the unsupervised cases: for motion segmentation, $\alpha = 0.1$, $\beta = 500$, 4-nn; for face clustering, $\alpha = 0.04$, $\beta = 0$; for digit clustering, $\alpha = 0.001$, $\beta = 32$, 4-nn.

6.7.1 Clustering with outliers

We apply the BB algorithm to four motion sequences with real outlying trajectories [45], *books*, *carbus3*, *carsTurning* and *nrbooks3*. The trajectories were obtained with an automatic tracker, and the ground-truth segmentation was manually determined. A trajectory was labeled as an inlier if it is correctly tracked in all frames, and an outlier if it is incorrectly tracked.

We use receiver operating characteristic (ROC) curve to evaluate the performance of outlier detection, where "positive" and "negative" correspond to outliers and inliers, respectively. The segmentation accuracies are also computed. We compare our method (referred to as BB-o) to two baselines: a) ALC-o [45]. A sample is an outlier if it belongs to a cluster with less than 5 samples after running ALC; b) LRR-o [10]. The sample is predicted as an outlier if the 2-norm of its corresponding column in the error matrix E of LRR is larger than a preset threshold. For our BB-o method, the RLM strategy is used to generate initial candidate models, and the ROC curves are obtained by varying C_o.

Fig. 9 shows the ROC curves for different algorithms. Table 9 lists the segmentation accuracies of different algorithms[9]. It can be seen that BB-o performs better than ALC-o and LRR-o in terms of both outlier detection and segmentation. The big gap of BB-o and the other ones in segmentation accuracy may be due to the following reason: for ALC-o and LRR-o, the segmentation relies so heavily on the relationship between samples that the wrongly retained outliers will seriously affect the final results; for BB-o, the relationship between samples and models is the dominant factor in the segmentation stage, resulting in its robustness to outliers.

9. Trajectories labeled and predicted as inliers both are included when we compute the *CE*. For LRR-o, the reported numbers are under parameters when the predicted outlier ratio equals the real one. For our method, we use the real outlier ratio as a priori and achieve segmentation by solving eq. (29).

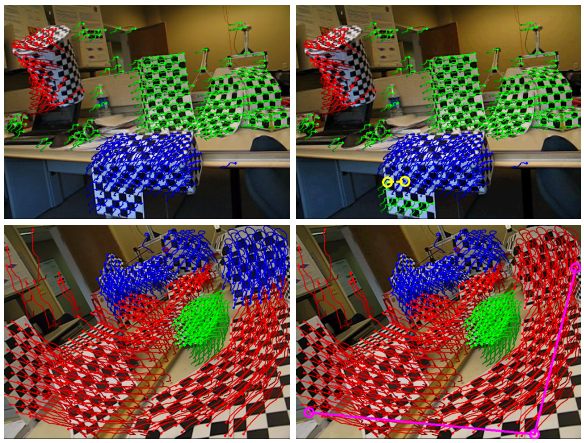


Fig. 11. Segmentation using pairwise constraints on *1R2RCT_B* (top) and *2T3RCRT* (bottom) sequences in Hopkins155 dataset. Different colors of trajectories indicate different segments. (a) The sample frames and the segmentation results without pairwise constraints. (b) The segmentation results using pairwise constraints. The yellow dotted lines on the ends represent cannot-link constraints between the two points marked by circles and the magenta solid ones represent must-link constraints.

6.7.2 Pairwise constraints

Pairwise constraints can be obtained by either human labeling or prior knowledge, e.g., the trajectories near the corners of a video usually belong to background. We choose the sequences where the BB method (unsupervised) makes errors, e.g., the *1R2RCT_B* with *CE*=3.55% and *2T3RCRT* with *CE*=9.58% as shown in Fig. 11, to do the experiments. By manually selecting one or several pairs of points, our method successfully corrects the errors.

6.7.3 Size priors

We demonstrate the effectiveness of our method in encoding the size priors by three experiments. Firstly, we test the encoding of the first size prior type: all the cluster sizes equal a number, e.g. 64 for the Extended Yale Face B dataset and 100 for the USPS dataset. By adding this knowledge, the face clustering accuracies are improved as shown in Fig. 12. For the USPS dataset, we also observe an improvement of *CE* from 11.80% to 8.80%.

Then we verify the other two types of size priors: the size of cluster containing point i is no smaller than Z; the size of cluster containing point i is no smaller than the size of cluster containing point j at a multiple of Z. The values of Z are assigned from the

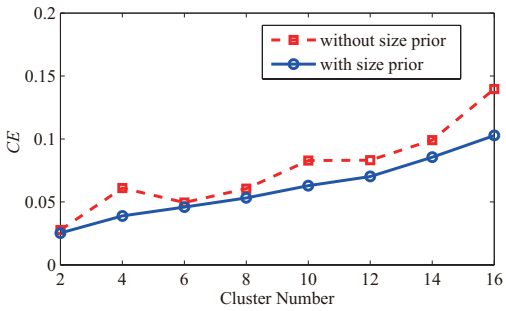
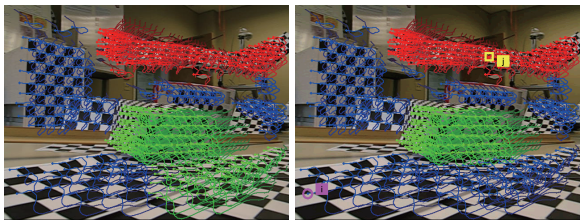


Fig. 12. The improvements of face clustering by adding equal size priors (rank-9 subspace model).



(a) (b)

Fig. 13. Segmentation using size priors on *1RT2RTCRT_B* sequence in Hopkins155 dataset. (a) The sample frames and the segmentation results using the "BB (os-LRR)" method in Table 2. (b) The segmentation results using two types of size priors separately. Note both settings produce the same segments, and therefore, only one image is drawn for each sequence to stand for both results.

ground truths. We correct the errors for *1RT2RTCRT_B* sequence after encoding the size priors as shown in Fig. 13.

6.7.4 Computational and Memory Costs

Table 10 lists the computational times and the peak memory[10] for different variants of the BB algorithm using the *2T3RCRT* sequence in Hopkins155 datasets. The *2T3RCRT* sequence consists of 543 29-frame trajectories and has 3 motions. In the experiments, we set the parameters the same as in Section 6.4.1. For the experiments encoding pairwise constraints, we use 1 must-link and 1 cannot-link constraints. For MRF optimization, the primal-dual solver [59] is used when the energy function is submodular and the QPBOP solver [60] is used when the energy function is non-submodular. For Linear programming, we use the mosek solver [62]. Observing that usually 1∼100 seconds are needed for over-segmentation methods, e.g. LRR, to generate initial candidate models, the computational times by our branch-and-bound optimization are very reasonable.

7 Conclusions

In subspace clustering, nearly all existing algorithms use *unconstrained subspace models*, meaning the points can be drawn from everywhere of a subspace, to describe the data, which usually produce poor results when severe noises, outliers or partially dependent subspaces exist. In this paper, we have proposed the alternative *constrained subspace model* for subspace clustering and instantiated it by several unsupervised and supervised constraints. We used a unified integer linear programming optimization framework for all the constraints. Applying the proposed

10. Both the computational times and peak memory are evaluated by the profiler tool in Matlab.

TABLE 10
Computational times (seconds) and peak memory (Mb) under different constraints for motion segmentation of the *2T3RCRT* sequence. "s.r." stands for "spatial regularity"; "p.w." stands for "pairwise constraints"; and "size{1,2,3}" represent the three types of size priors.

		basic	outlier	p.w.	size1	size2	size3
-s.r.	time	0.1	0.1	0.4	7.6	17.3	18.0
	memory	0.8	0.8	0.8	7.8	9.8	13.6
+s.r.	time	0.5	11.0	92.5	16.0	30.3	31.4
	memory	1.0	7.8	1.0	10.7	16.7	22.2

method with manifold and spatial regularity constraints to two popular applications of subspace clustering, motion segmentation and face clustering, we achieve much better performance than the state-of-the-art. We have also shown the effectiveness of the proposed framework in exploiting various supervised constraints.

The main limitation of our method is the high computational complexity when the number of candidate models is large. Although it can be alleviated by pre-discarding bad candidates using some simpler strategies, a better way is to develop more efficient algorithms, which will be our main future direction.

References

[1] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: a factorization method," *IJCV*, vol. 9, no. 2, pp. 137–154, 1992.
[2] R. Basri and D. Jacobs, "Lambertian reflectance and linear subspaces," *IEEE TPAMI*, vol. 25, no. 2, pp. 218–233, 2003.
[3] J. Gonzalez-Mora, F. D. la Torre, R. Murthi, N. Guil, and E. L. Zapata, "Bilinear active appearance models," in *ICCV*, 2007, pp. 1–8.
[4] G. E. Hinton, M. Revow, and P. Dayan, "Recognizing handwritten digits using mixtures of linear models," in *NIPS*, 1994, pp. 1015–1022.
[5] Y. Ma, H. Derksen, W. Hong, and J. Wright, "Segmentation of multivariate mixed data via lossy data coding and compression," *IEEE TPAMI*, vol. 29, no. 9, pp. 1546–1562, 2007.
[6] B. Cheng, G. Liu, J. Wang, Z. Huang, and S. Yan, "Multi-task low-rank affinity pursuit for image segmentation," in *ICCV*, 2011, pp. 2439–2446.
[7] R. Vidal, "Subspace clustering," *IEEE Signal Processing Magazine*, vol. 28, no. 2, pp. 52–68, 2011.
[8] C.-Y. Lu, H. Min, Z.-Q. Zhao, L. Zhu, D.-S. Huang, and S. Yan, "Robust and effficient subspace segmentation via least squares regression," in *ECCV*, 2012, pp. 347–360.
[9] R. Vidal, R. Tron, and R. Hartlet, "Multiframe motion segmentation with missing data using powerfactorization and GPCA," *IJCV*, vol. 79, no. 1, pp. 85–105, 2008.
[10] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma, "Robust recovery of subspace structures by low-rank representation," *IEEE TPAMI*, 2012.
[11] J. Costeira and T. Kanade, "A multibody factorization method for independently moving objects," *IJCV*, vol. 29, no. 3, pp. 108–121, 1998.
[12] G. Liu, Z. Lin, and Y. Yu, "Robust subspace segmentation by low-rank representation," in *ICML*, 2010, pp. 663–670.
[13] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE TPAMI*, vol. 23, no. 11, pp. 1222–1239, 2001.
[14] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge University Press, 2007.
[15] R. Tron and R. Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithm," in *Proc. CVPR*, 2007.
[16] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE TPAMI*, vol. 23, no. 6, pp. 643–660, 2001.
[17] K.-C. Lee, J. Ho, and D. J. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE TPAMI*, vol. 27, no. 5, pp. 684–698, 2005.
[18] J. J. Hull, "A database for handwritten text recognition research," *IEEE TPAMI*, vol. 16, no. 5, pp. 550–554, 1994.
[19] S. X. Yu and J. Shi, "Segmentation given partial grouping constraints," *IEEE TPAMI*, vol. 26, no. 2, pp. 173–183, 2004.
[20] H. Hu, J. Feng, C. Yu, and J. Zhou, "Multi-class constrained normalized cut with hard, soft, unary and pairwise priors and its applications to object segmentation," *IEEE TIP*, 2013.

[21] C. W. Gear, "Multibody grouping from motion images," *IJCV*, vol. 29, no. 2, pp. 133–150, 1998.
[22] L. Zelnik-Manor and M. Irani, "Degeneracies, dependencies and their implications in multi-body and multi-sequence factorizations," in *CVPR (2)*, 2003, pp. 287–293.
[23] Z. Fan, J. Zhou, and Y. Wu, "Multibody grouping by inference of multiple subspaces from high-dimensional data using oriented frames," *IEEE T-PAMI*, vol. 28, no. 1, pp. 91–105, 2006.
[24] R. Vidal, Y. Ma, and S. Sastry, "Generalized principal component analysis (GPCA)," *IEEE TPAMI*, vol. 27, no. 12, pp. 1945–1959, 2005.
[25] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE TPAMI*, vol. 22, no. 8, pp. 888–905, 2000.
[26] J. H. Park, H. Zha, and R. Kasturi, "Spectral clustering for robust motion segmentation," in *ECCV (4)*, 2004, pp. 390–401.
[27] J. Yan and M. Pollefeys, "A general framework for motion segmentation: Independent, articulated, rigid, non-rigid, degenerate and non-degenerate," in *Proc. ECCV*, 2006.
[28] G. Chen and G. Lerman, "Spectral curvature clustering (SCC)," *IJCV*, vol. 81, no. 3, pp. 317–330, 2009.
[29] E. Elhamifar and R. Vidal, "Sparse subspace clustering," in *CVPR*, 2009, pp. 2790–2797.
[30] R. Li, B. Li, K. Zhang, C. Jin, and X. Xue, "Groupwise constrained reconstruction for subspace clustering," in *ICML*, 2012.
[31] H. Hu, Z. Lin, J. Feng, and J. Zhou, "Smooth representation clustering," in *CVPR*, 2014, pp. 3834–3841.
[32] T. Zhang, A. Szlam, Y. Wang, and G. Lerman, "Randomized hybrid linear modeling by local best-fit flats," in *CVPR*, 2010, pp. 1927–1934.
[33] S. Wang, X. Yuan, T. Yao, S. Yan, and J. Shen, "Efficient subspace segmentation via quadratic programming," in *AAAI*, 2011, pp. 519–524.
[34] V. Zografos, L. Ellisy, and R. Mester, "Discriminative subspace clustering," in *CVPR*, 2013.
[35] C. Lu, J. Feng, Z. Lin, and S. Yan, "Correlation adaptive subspace segmentation by trace lasso," in *ICCV*, 2013, pp. 1345–1352.
[36] R. Tron and R. Vidal, "A benchmark for the comparison of 3-D motion segmentation algorithms," in *CVPR*, 2007.
[37] K. Lee, J. Ho, and D. Kriegman, "Acquiring linear subspaces for face recognition under variable lighting," *IEEE TPAMI*, vol. 27, no. 5, pp. 684–698, 2005.
[38] F. Lauer and C. Schnorr, "Spectral clustering of linear subspaces for motion segmentation," in *ICCV*, 2009.
[39] L. Zappella, X. Lladó, E. Provenzi, and J. Salvi, "Enhanced local subspace affinity for feature-based motion segmentation," *Pattern Recognition*, vol. 44, no. 2, pp. 454–470, 2011.
[40] 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*, vol. 24, no. 6, pp. 381–395, 1981.
[41] K. Kanatani and Y. Sugaya, "Multi-state optimization for multi-body motion segmentation," in *Australia-Japan Advanced Workshop on Computer Vision*, 2003.
[42] T. Zhang, A. Szlam, and G. Lerman, "Median k-flats for hybrid linear modeling with many outliers," in *Workshop on Subspace Methods*, 2009.
[43] T.-J. Chin, H. Wang, and D. Suter, "The ordered residual kernel for robust motion subspace clustering," in *NIPS*, 2009, pp. 333–341.
[44] S. Mittal, S. Anand, and P. Meer, "Generalized projection based M-estimator: Theory and applications," *IEEE TPAMI*, 2012.
[45] S. Rao, R. Tron, R. Vidal, and Y. Ma, "Motion segmentation in the presence of outlying, incomplete, or corrupted trajectories," *IEEE TPAMI*, vol. 32, no. 10, pp. 1832–1845, 2010.
[46] N. Lazic, I. E. Givoni, B. J. Frey, and P. Aarabi, "Floss: Facility location for subspace segmentation," in *ICCV*, 2009, pp. 825–832.
[47] N. Lazic, B. J. Frey, and P. Aarabi, "Solving the uncapacitated facility location problem using message passing algorithms," in *AISTATS*, 2010, pp. 429–436.
[48] C. M. Lee and L. Cheong, "Minimal basis facility location for subspace segmentation," in *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, 2013, pp. 1585–1592.
[49] M. Balinski, "Integer programming: Methods, uses, computation," *Management Science*, vol. 12, no. 3, pp. 253–313, 1965.
[50] M. A. Efroymson and T. L. Ray, "A branch-bound algorithm for plant location," *Operations Research*, vol. 14, no. 3, pp. 361–368, 1966.
[51] A. Klose, "A branch and bound algorithm for an uncapacitated facility location problem with a side constraint," *International Transactions in Operational Research*, vol. 5, no. 2, pp. 155 – 168, 1998.
[52] K. Holmberg, M. Rnnqvist, and D. Yuan, "An exact algorithm for the capacitated facility location problems with single sourcing," *European Journal of Operational Research*, vol. 113, no. 3, pp. 544 – 559, 1999.
[53] A. Schrijver, *Theory of linear and integer programming*. Wiley, 1998.
[54] E. L. Lawler and D. E. Wood, "Branch-and-bound methods: A survey," *Operations Research*, vol. 14, no. 4, pp. 699–719, 1966.
[55] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge University Press, 2004.
[56] M. Marques and J. Costeira, "Estimating 3d shape from degenerate sequences with missing data," *CVIU*, vol. 113, no. 2, pp. 261–272, 2009.
[57] A. D. Bue, J. M. F. Xavier, L. de Agapito, and M. Paladini, "Bilinear modeling via augmented lagrange multipliers (BALM)," *IEEE TPAMI*, vol. 34, no. 8, pp. 1496–1508, 2012.
[58] A. Shashua, "On photometric issues in 3D visual recognition from a single 2D image," *IJCV*, vol. 21, no. 1-2, pp. 99–122, 1997.
[59] N. Komodakis and G. Tziritas, "Approximate labeling via graph cuts based on linear programming," *IEEE TPAMI*, vol. 29, no. 8, pp. 1436–1453, 2007.
[60] C. Rother, V. Kolmogorov, V. S. Lempitsky, and M. Szummer, "Optimizing binary MRFs via extended roof duality," in *CVPR*, 2007.
[61] A. P. Eriksson, C. Olsson, and F. Kahl, "Normalized cuts revisited: A reformulation for segmentation with linear grouping constraints," in *ICCV*, 2007, pp. 1–8.
[62] MOSEK ApS, "Mosek modeling manual," 2013. [Online]. Available: http://www.mosek.com
[63] H. Kuhn, "The Hungarian method for the assignment problem," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
[64] K. Kanatani, "Geometric information criterion for model selection," *IJCV*, vol. 26, no. 3, pp. 171–189, 1998.
[65] K. Kanatani, "Estimating the number independent motions for multibody segmentation," in *Proc. ACCV*, 2002.
[66] H. Wang and D. Suter, "Mdpe: A very robust estimator for model fitting and range image segmentation," *IJCV*, vol. 59, no. 2, pp. 139–166, 2004.
[67] E. Elhamifar and R. Vidal, "Sparse subspace clustering: Algorithm, theory, and applications," *IEEE TPAMI*, 2013.
[68] G. Liu and S. Yan, "Latent low-rank representation for subspace segmentation and feature extraction," in *ICCV*, 2011, pp. 1615–1622.
[69] A. Aldroubi and A. Sekmen, "Nearness to local subspace algorithm for subspace and motion segmentation," *IEEE Signal Processing Letters*, vol. 19, no. 10, pp. 704–707, 2012.
[70] A. Goh and R. Vidal, "Segmenting motions of different types by unsupervised manifold clustering," in *CVPR*, 2007.
[71] E. Elhamifar and R. Vidal, "Sparse manifold clustering and embedding," in *NIPS*, 2011, pp. 55–63.
[72] H. Hu, Q. Gu, L. Deng, and J. Zhou, "Multiframe motion segmentation via penalized map estimation and linear programming," in *BMVC*, 2009.
[73] T.-J. Chin, D. Suter, and H. Wang, "Multi-structure model selection via kernel optimisation," in *CVPR*, 2010, pp. 3586–3593.

Han Hu received the B.S. degree and ph.D. from Department of Automation, Tsinghua University, Beijing, China, in 2008 and 2014, respectively. From 2014, he has been a researcher in Institute of Deep Learning, Baidu Research, Beijing, China. Currently, his main interests are two-fold: high-level recognition using various low-level visual cues, and learning methods for low-level vision tasks.

Jianjiang Feng (M'10) is an associate professor in the Department of Automation at Tsinghua University, Beijing. He received the B.S. and Ph.D. degrees from the School of Telecommunication Engineering, Beijing University of Posts and Telecommunications, China, in 2000 and 2007, respectively. From 2008 to 2009, he was a Post Doctoral researcher in the Pattern Recognition & Image Processing laboratory at Michigan State University. His research interests include fingerprint recognition, palmprint recognition, and structural matching.

Jie Zhou (M'01-SM'04) received B.S. degree and M.S. degree both from Department of Mathematics, Nankai University, Tianjin, China, in 1990 and 1992, respectively. He received Ph.D. degree from Institute of Pattern Recognition and Artificial Intelligence, Huazhong University of Science and Technology (HUST), Wuhan, China, in 1995. From then to 1997, he served as a postdoctoral fellow in Department of Automation, Tsinghua University, Beijing, China. From 2003, he has been a full professor in Department of Automation, Tsinghua University. His research area includes computer vision, pattern recognition and image processing. Dr. Zhou is a senior member of IEEE and a recipient of the National Outstanding Youth Foundation of China.