SCAMS: Simultaneous Clustering and Model Selection

Zhuwen Li[1], Loong-Fah Cheong[1] and Steven Zhiying Zhou[1,2]
[1]Dept. of Electrical & Computer Engineering, National University of Singapore
[2]National University of Singapore (Suzhou) Research Institute
{lizhuwen, eleclf, elezzy}@nus.edu.sg

Abstract

While clustering has been well studied in the past decade, model selection has drawn less attention. This paper addresses both problems in a joint manner with an indicator matrix formulation, in which the clustering cost is penalized by a Frobenius inner product term and the group number estimation is achieved by a rank minimization. As affinity graphs generally contain positive edge values, a sparsity term is further added to avoid the trivial solution. Rather than adopting the conventional convex relaxation approach wholesale, we represent the original problem more faithfully by taking full advantage of the particular structure present in the optimization problem and solving it efficiently using the Alternating Direction Method of Multipliers. The highly constrained nature of the optimization provides our algorithm with the robustness to deal with the varying and often imperfect input affinity matrices arising from different applications and different group numbers. Evaluations on the synthetic data as well as two real world problems show the superiority of the method across a large variety of settings.

1. Introduction

Many computer vision problems, such as image segmentation, multi-structure recovery and so on, involve solving the clustering problem at some point. Often, an affinity graph is set up and then fed into a spectral clustering framework [20] to infer the clustering of the data into groups. Such spectral graph methods include Ratio Cut [15], Normalized Cut [30], *etc*. However, deciding on the number of clusters remains an open problem for all such algorithms.

The simplest way to estimate the group number is to count the number of zero eigenvalues of the Laplacian matrix of the affinity graph. However, it performs not very well in practice when data contain structures at different scales of size and density, and when data are contaminated by noise. In these cases, these eigenvalues deviate from zero in a complex manner, and it is non-trivial to determine the number of eigenvalues close to zero in a robust manner.

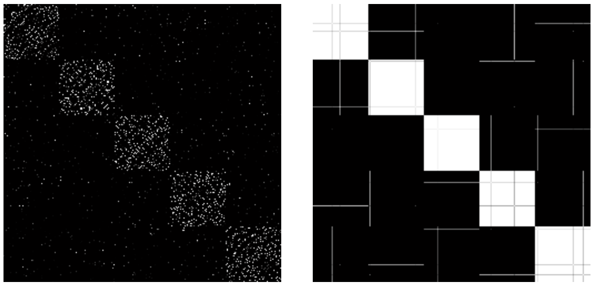


Figure 1. Left: A contaminated affinity matrix $\mathbf{A}$ with 5 clusters. Right: The recovered $\mathbf{G}$ contains 5 almost perfect blocks. Further processing by the proposed Boolean matrix factorization algorithm will obtain perfect blocks from this $\mathbf{G}$.

In this paper, we propose a novel algorithm to perform simultaneous clustering and model selection (SCAMS). Given an affinity matrix $\mathbf{A}$ with non-negative entries, our task can be conceptually viewed as discovering which $\mathbf{A}(i,j)$ are small enough; this is essentially saying that elements i and j are dissimilar and should be placed in different clusters. Just as importantly, we should also ensure that elements i and j are not linked indirectly through other elements in the graph. This is realized by adopting an indicator matrix formulation explained as follows. We take the Frobenius inner product of the affinity matrix $\mathbf{A}$ and $\mathbf{G} = \mathbf{Z}\mathbf{Z}^T$, where $\mathbf{Z}$ is an indicator matrix whose rows indicate to which group a point belongs. We maximize this Frobenius inner product term $\langle \mathbf{A}, \mathbf{G} \rangle$ so as to keep $\mathbf{G}$ as close to the data term $\mathbf{A}$ as possible, while at the same time, we impose several constraints so as to ensure meaningful solutions for $\mathbf{G}$. Firstly, there should be a trade-off between the complexity of the model and goodness of fit. The model complexity is indicated by the rank of $\mathbf{G}$ (see Section 3); thus, we seek to minimize the rank of $\mathbf{G}$ to discriminate against a more complex model. Secondly, we should also limit the cardinality of $\mathbf{G}$ — the number of nonzero entries in $\mathbf{G}$ — so as to discover structure in the data (indicated by the sparsity pattern of $\mathbf{G}$). In fact, without this penalty term on cardinality, we will end up with the trivial solution of

$\mathbf{G}$ being the all-one matrix (all data belong to one cluster). Together with the $\{0,1\}$ constraint on $\mathbf{G}$, this formulation in effect examines the connectivity of the entire graph and tends to set $\mathbf{G}(i,j)$ to one if elements i and j are linked indirectly through other elements. This highly constrained formulation also provides our algorithm with the robustness to deal with the varying and often imperfect input affinity matrices generated from different applications and different group numbers (despite the best efforts of works to generate these matrices [12, 19, 37]). Figure 1 shows a recovery result of our algorithm. Notice that our algorithm is able to recover a nearly perfect 0-1 block diagonal $\mathbf{G}$ from the contaminated affinity matrix.

Our problem now involves solving for a low-rank and sparse matrix $\mathbf{G}$, subject to a number of constraints over the integer variables, all of which lead to an NP-hard problem. In many problem instances, the convex proxy to an NP-hard problem may not be a good approach. Instead, there might be a need to represent the original problem more faithfully — an approximate solution to the right problem can be better than the exact solution to the wrong problem. In our case, we take full advantage of the particular structure present in the optimization problem, optimizing over the rank and ℓ_0-norm directly and yet solving the problem efficiently using the Alternating Direction Method of Multipliers (ADMM) method [9, 18].

A common heuristic to obtain the final clustering is to factorize $\mathbf{G}$ back to $\mathbf{Z}\mathbf{Z}^T$ using Cholesky decomposition [14], and assign each data point to the index with the maximum value in each row of $\mathbf{Z}$. However, Cholesky decomposition occasionally produces bad results even if $\mathbf{G}$ contains nearly perfect blocks because it does not impose any Boolean constraint on the factor matrices. Thus, we propose a variant of an existing Boolean matrix factorization (BMF) algorithm [23] to finesse a better decomposition.

The contribution of this paper is summarized as follows. 1) We formulate the model selection as a rank minimization problem, leading to a joint optimization of clustering and model selection. Trivial solution is avoided by adding a sparsity penalty term. The low rank penalty, together with other constraints that enforce the indicator matrix formulation, highly constrains the solution space and provide our algorithm with the ability to repair imperfections in the affinity matrix, *e.g.* filling in the connectivity gap or ignoring dubious connections. 2) The inner optimization subproblems in each iteration are designed to take full advantage of the particular structure present in our problem. This results in an effective and efficient algorithm that represents the original problem more faithfully and works well under a wider range of changing conditions such as increasing group number and noise level. Our extensive experiments shed light on how the different attributes of the affinity matrices constructed by different methods impact on model selection, further highlighting the strength of our algorithm. 3) We propose a novel Boolean matrix factorization algorithm to obtain a better decomposition which lends itself to more accurate clustering.

2. Related works

There have been many algorithms devised for the clustering problem; we will briefly review some major approaches here. In the spectral graph approach, one needs to determine the number of zero eigenvalues of the Laplacian matrix of the affinity graph in a robust manner. Heuristics particularly designed for this purpose include the eigengap heuristic, the elbow criterion, the gap statistic [33], the silhouette index [28], and several recent measures [3, 19, 31]. In the information-theoretic approach, one aims to balance the goodness of fit against the complexity of the model. A classical measure is the AIC [2], which is followed by many variants [16, 34]. Another measure is based on compression efficiency, such as the Minimum Description Length (MDL) [21, 27, 32]. The major drawback of this kind of methods is that they are usually model-dependent. Among the many clustering methods, one can also distinguish another category which is based on the stability of the solutions [8, 17]. The stability is measured by the pairwise similarities between clustering results with respect to perturbations such as sub-sampling or the addition of noise, and the optimal number of clusters is then given by the most stable solution. Many of the above methods involve particular choices to be made at the outset, for example the value of a particular thresholding parameter. Many of them also require that the number of clusters to be found by another criterion. That is, a two-step procedure is performed: a clustering criterion determines the optimal assignments for a given number of clusters and a separate criterion measures the goodness of the classification to determine the number of clusters. Our method involves very little domain-specific assumptions, and it performs a joint optimization of clustering and model selection in one single step. While our algorithm also involves choice of weights, the experimental results show that these chosen values works well across a wide range of different settings, which is not what can be said about other compared methods.

Our method is also related to the probabilistic mixture model approach in the sense that both combine clustering and model selection in a single step. However, in the probabilistic mixture approach, one needs to assume that the data can be described by a mixture of multivariate distributions with some parameters that determine their shape with known distribution of the data. Our method involves no such assumption. Another similarity between such probabilistic mixture model approach and our method lies in the objective function. In fact, if we view our affinity matrix $\mathbf{A}$ as a covariance matrix, the objective functions are identical except for the integer constraint(*e.g.* see [5, 10, 24]).

Lastly, we have in the preceding section likened the optimization as one of discovering which affinity values are small enough to be set as zero. This can be regarded as a thresholding operation on the affinity values. In fact, if we know the threshold, we can convert our problem into a correlation clustering (CC) problem [6]. We can either use the original unweighted form of CC, in which the affinity matrix $\mathbf{A}$ defines a graph with all edges assigned weights of either $+1$ or -1 (representing "similar" and "dissimilar" respectively), or one can use the general form of CC with real edge weights [4, 11]. In either case, CC maximizes the Frobenius inner product term $\langle \mathbf{A}, \mathbf{Z}\mathbf{Z}^T \rangle$ which is identical to our problem. The difficulty of this line of approach is in determining a proper threshold to distinguish between "similar" and "dissimilar". Our method eschews such direct thresholding and instead utilizes the generic low rank and sparsity assumption to perform the operation. Furthermore, the CC problem is an instance of the quadratic semi-assignment problem (QSAP) [36], which is NP-complete when the cluster number is unknown. Our method provides a tractable solution via carefully exploiting the structure of the problem and appropriate relaxations, and we show in our experiments that the results are of good quality and stable across a range of noise level and cluster number.

3. Clustering with Model Selection

3.1. Problem formulation

Suppose we are given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{A})$, where $\mathcal{V} = \{v_i\}_{i=1}^N$ is the set of the N nodes, $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ denotes the set of the edges between the nodes, and $\mathbf{A} \in \mathbb{R}^{N \times N}$ is an affinity matrix constructed by some method, with each element $\mathbf{A}(i,j) \geq 0$ being the affinity between sample v_i and v_j. $\mathbf{A}(i,j) = 0$ suggests that v_i and v_j are completely dissimilar, and thus likely to be disconnected, while $\mathbf{A}(i,j) > 0$ means there is the possibility for the two nodes to be clustered into the same group. The larger the value, the more likely these two nodes should be in the same group. Now the task is to cluster these N nodes into K groups, where the group number K is unknown a priori and needs to be estimated.

For ease of problem formulation, let us assume for now that K is known. Denote $\mathbf{Z} \in \mathbb{R}^{N \times K}$ as the indicator matrix, whose row entries indicate to which group the points belong, *i.e.*, if point i belongs to group k, $\mathbf{Z}(i,k) = 1$ and the remaining entries of the i-th row are all 0's. Thus, if point i and j belong to the same group, $\langle \mathbf{Z}(i,:), \mathbf{Z}(j,:) \rangle = 1$; otherwise, $\langle \mathbf{Z}(i,:), \mathbf{Z}(j,:) \rangle = 0$, where $\langle \cdot, \cdot \rangle$ denote the inner product of two vectors, or the Frobenius inner product of two matrices, as the case may be. As discussed before, we want to maximize the following objective function:

$$f(\mathbf{Z}) = \langle \mathbf{A}, \mathbf{Z}\mathbf{Z}^T \rangle = tr(\mathbf{A}^T \mathbf{Z}\mathbf{Z}^T), \tag{1}$$

where $tr(\cdot)$ indicates the trace operator of the given matrix.

From the preceding, we have $\mathbf{G} = \mathbf{Z}\mathbf{Z}^T$; therefore, $\mathbf{G}$ is positive semi-definite (PSD) and the rank of $\mathbf{G}$ is exactly K. We can convert the above problem into the following minimization problem over $\mathbf{G}$ by adding a negative sign in front of the affinity matrix and denoting $\mathbf{W} = -\mathbf{A}$:

$$\begin{aligned} \min. \quad & tr(\mathbf{W}^T \mathbf{G}), \\ s.t. \quad & \mathbf{G} \in \mathbf{S}_+, \\ & diag(\mathbf{G}) = 1, \\ & rank(\mathbf{G}) = K, \\ & \mathbf{G} \in \{0,1\}^{N \times N}, \end{aligned} \tag{2}$$

where $\mathbf{S}_+$ is the PSD cone and $diag(\cdot)$ are the diagonal entries of the matrix, this constraint merely reflecting the fact that the same point cannot be split into different groups.

Since K is unknown a priori and usually $K \ll N$, we estimate it by minimizing the rank of $\mathbf{G}$. However, this will result in a trivial solution for $\mathbf{G}$, *i.e.*, the all one matrix, which is rank-one and "covers" all the entries of the affinity matrix by 1. To avoid the trivial solution, we further add an ℓ_0 penalty on $\mathbf{G}$ to enforce sparsity on its entries. This would force the optimization to only insert ones at those $\mathbf{G}(i,j)$ locations where the magnitude of the corresponding $\mathbf{A}(i,j)$ is large. Accordingly, we now have

$$\begin{aligned} \min. \quad & tr(\mathbf{W}^T \mathbf{G}) + \lambda rank(\mathbf{G}) + \gamma ||\mathbf{G}||_0, \\ s.t. \quad & \mathbf{G} \in \mathbf{S}_+, \\ & diag(\mathbf{G}) = 1, \\ & \mathbf{G} \in \{0,1\}^{N \times N}, \end{aligned} \tag{3}$$

where $|| \cdot ||_0$ is the ℓ_0 norm, which counts the number of nonzero elements, and λ and γ are the parameters to weigh the respective penalty terms. To make the problem tractable, we first relax the constraint $\mathbf{G} \in \{0,1\}^{N \times N}$ to obtain real-valued entries $\mathbf{G} \in [0,1]^{N \times N}$. Next, instead of replacing the rank and the ℓ_0 norm with their convex proxies, we optimize them directly by taking full advantage of the particular structure present in the problem. In particular, as we will show later, the resulting inner optimization problems can be solved analytically by eigen-decomposition and soft-thresholding operations. By now, the problem to be solved has the following form

$$\begin{aligned} \min. \quad & tr(\mathbf{W}^T \mathbf{G}) + \lambda rank(G) + \gamma ||\mathbf{G}||_0, \\ s.t. \quad & \mathbf{G} \in \mathbf{S}_+, \\ & diag(\mathbf{G}) = 1, \\ & \mathbf{G} \in [0,1]^{N \times N}. \end{aligned} \tag{4}$$

3.2. Solver

For efficiency, we adopt the ADMM method [9, 18] to solve this problem. We first convert (4) to the following equivalent problem:

$$\begin{aligned} \min. \quad & tr(\mathbf{W}^T \mathbf{G}) + \lambda rank(\mathbf{G}) + \gamma ||\mathbf{H}||_0 + g(\mathbf{H}), \\ s.t. \quad & \mathbf{G} \in \mathbf{S}_+, \\ & \mathbf{G} = \mathbf{H} - diag(\mathbf{H}) + \mathbf{I}, \end{aligned} \tag{5}$$

where g is the indicator function of the convex set $[0,1]^{N\times N}$, which returns 0 if it is in the set, ∞ otherwise, and $\mathbf{H}$ is an intermediate variable introduced to make the problem tractable. The augmented Lagrange function is

$$\begin{aligned} \mathcal{L} = \quad & tr(\mathbf{W}^T\mathbf{G}) + \lambda rank(\mathbf{G}) + \gamma||\mathbf{H}||_0 + g(\mathbf{H}) + \\ & tr(\mathbf{Y}^T(\mathbf{G}-\mathbf{H}+diag(\mathbf{H})-\mathbf{I})) + \\ & \tfrac{1}{2\mu}\|\mathbf{G}-\mathbf{H}+diag(\mathbf{H})-\mathbf{I}\|_F^2, \\ s.t. \quad & \mathbf{G}\in\mathbf{S}_+, \end{aligned} \tag{6}$$

where $\mathbf{Y}$ is the Lagrange parameter, and $\mu > 0$ is a penalty parameter. The function can be minimized with respect to $\mathbf{G}$ and $\mathbf{H}$ alternatingly, by fixing the other variable, and then updating the Lagrange multipliers $\mathbf{Y}$. The overall framework of the alternating direction method is shown in Algorithm 1, with the detailed solver for each subproblem to be described later.

Algorithm 1 Solving (4) by ADMM

```
Input: Negative affinity matrix $\mathbf{W}$, parameters $\lambda$ and $\gamma$.
  Initialize: $\mathbf{G} = \mathbf{H} = \mathbf{Y} = \mathbf{0}_{N\times N}$, $\mu = 10^6$, $\rho = 1.1$,
  $\mu_{\min} = 10^{-10}$ and $\epsilon = 10^{-8}$.
  while not converged do
    Step 1 Fix the others and update $\mathbf{G}$ as
    $\mathbf{G} = arg\min_{\mathbf{G}} \|\mathbf{G}-\mathbf{H}+\mu(\mathbf{W}+\mathbf{Y})\|_F^2 + 2\mu\lambda rank(\mathbf{G})$,
    $s.t.\ \mathbf{G}\in\mathbf{S}_+$.
    Step 2 Fix the others and update $\mathbf{H}$ as
    $\mathbf{H}' = arg\min_{\mathbf{H}} \|\mathbf{H}-\mathbf{G}-\mu\mathbf{Y}\|_F^2 + 2\mu\gamma\|\mathbf{H}\|_0 + g(\mathbf{H})$,
    $\mathbf{H} = \mathbf{H}' - diag(\mathbf{H}') + \mathbf{I}$.
    Step 3 Update the multipliers
    $\mathbf{Y} = \mathbf{Y} + \frac{1}{\mu}(\mathbf{G}-\mathbf{H})$.
    Step 4 Update the parameter $\mu$ by $\mu = \max(\frac{\mu}{\rho}, \mu_{\min})$.
    Step 5 Check the convergence conditions:
    $\|\mathbf{G}-\mathbf{H}\|_\infty \le \epsilon$.
  end while
```

Solving G. In step 1 of Algorithm 1, the solution of $\mathbf{G}$ involves minimizing the rank plus a convex quadratic function in the PSD cone. It can be efficiently solved using the following theorem. The proof is analogous to that of Theorem 16 in [25], with the nuclear norm replaced by the rank.

Theorem 1. *For any square matrix $\mathbf{S}\in\mathbb{R}^{N\times N}$, the unique closed form solution to the optimization problem*

$$\begin{aligned} \mathbf{G}^* = \quad & arg\min_{\mathbf{G}} \|\mathbf{G}-\mathbf{S}\|_F^2 + \lambda rank(\mathbf{G}), \\ s.t. \quad & \mathbf{G}\in\mathbf{S}_+. \end{aligned} \tag{7}$$

takes the form

$$\mathbf{G}^* = \mathbf{Q}\mathcal{H}_\lambda(\mathbf{\Lambda})\mathbf{Q}^T, \tag{8}$$

where $\mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^T$ is the spectrum(eigen-) decomposition of $\widehat{\mathbf{S}} = (\mathbf{S}+\mathbf{S}^T)/2$ and $\mathcal{H}_\lambda(\cdot)$ is the thresholding operator acting on each element of the matrix, and defined as

$$\mathcal{H}_\lambda(v) = \begin{cases} 0 & \text{if} \quad v<0 \ \text{or} \ v^2 \le \lambda, \\ v & \text{otherwise} . \end{cases} \tag{9}$$

Solving H. In step 2 of Algorithm 1, the update of $\mathbf{H}'$ involves minimizing the ℓ_0 norm plus a convex quadratic function in the convex set $[0,1]^{N\times N}$. Since this problem is obviously separable, each elements can be optimized individually and simple manipulation suggests the following theorem.

Theorem 2. *For any matrix $\mathbf{M}\in\mathbb{R}^{M\times N}$, the unique closed form solution to the optimization problem*

$$\mathbf{H}^* = arg\min_{\mathbf{H}} \|\mathbf{H}-\mathbf{M}\|_F^2 + \gamma\|\mathbf{H}\|_0 + g(\mathbf{H}), \tag{10}$$

takes the form

$$\mathbf{H}^* = \mathcal{T}_\gamma(\mathbf{M}). \tag{11}$$

where $\mathcal{T}_\gamma(\cdot)$ is the thresholding operator acting on each element of the matrix, and is defined as

$$\mathcal{T}_\gamma(v) = \begin{cases} 1 & \text{if} \quad v>1 \ \text{and} \ \min(v^2, 2v-1) > \gamma \\ 0 & \text{else if} \quad v<0 \ \text{or} \ v^2 \le \gamma \ \text{or} \ v>1 \\ v & \text{otherwise} . \end{cases} \tag{12}$$

With the closed-form solutions, global minimums are assured for both sub-problems. Nevertheless, the algorithm as a whole does not have guarantee to convergence as the two sub-problems are non-convex. As far as we know, there is no general convergence theory for ADMM applied to non-convex problems, but numerical results in [29] on low-rank matrix factorization show that ADMM performed well for solving certain non-convex models. Indeed, our algorithm also has strong convergence behavior empirically.

4. Constrained Boolean Matrix Factorization

As the Cholesky decomposition occasionally yields poor binary result of $\mathbf{Z}$ even if $\mathbf{G}$ is nearly a 0-1 block diagonal matrix, we adapt the idea of BMF to achieve a better decomposition. Our proposed BMF method is similar to the Asso algorithm [22, 23] but takes into account the additional PSD constraint, and that each row of $\mathbf{Z}$ contains only one 1 (this latter constraint can be interpreted as an orthonormal constraint under Boolean algebra).

For the sake of completeness, we first give a brief introduction of BMF; for more details, see [22, 23]. We then formally define our BMF problem with its PSD and orthonormal constraints.

BMF aims to (approximately) represent a Boolean matrix as the Boolean product of two Boolean matrices. Here "Boolean" matrix means that the matrix contains only 0's and 1's. Using the superscript b to stand for Boolean matrix, let $\mathbf{B}^b \in \{0,1\}^{N\times K}$ and $\mathbf{C}^b \in \{0,1\}^{K\times M}$ be the two

Boolean matrices, whose *Boolean matrix product*, $\mathbf{B}^b \circ \mathbf{C}^b$ yields $\mathbf{A}^b$, with $\mathbf{A}^b(i,j) = \vee_{k=1}^{K} \mathbf{B}^b(i,k)\mathbf{C}^b(k,j)$, and the *OR* operation $\vee$ is the normal sum but with addition defined as $1+1=1$. Our problem can now be formally defined as

Problem 1. ***Constrained Boolean Matrix Factorization (CBMF)*** *with the PSD and Boolean orthonormal constraints. Given a Boolean matrix* $\mathbf{G}^b \in \{0,1\}^{N\times N}$ *and an upper bound* K_0*, find Boolean matrix* $\mathbf{Z}^b \in \{0,1\}^{N\times K}$*,* $K \leq K_0$*, such that* $\mathbf{Z}^b$ *satisfies*

$$\begin{aligned} \min. \quad & |\mathbf{G}^b \oplus (\mathbf{Z}^b \circ \mathbf{Z}^{b^T})|, \\ s.t. \quad & \mathbf{Z}^{b^T} \circ \mathbf{Z}^b = \mathbf{I}_{K\times K}, \end{aligned} \tag{13}$$

where $|\cdot|$ is the norm of a Boolean matrix and defined as the number of 1's in it, *i.e.*, $|\mathbf{A}^b| = \sum_{i,j} \mathbf{A}^b(i,j)$, and $\oplus$ is the *Exclusive-OR* operation applied element-wise, and defined as the normal addition but with $1+1=0$.

The original Asso algorithm solves the BMF problem via the heuristic approach of generating the candidate columns using pairwise association accuracies. More specifically, it generates a matrix $\mathbf{D}$ with $\mathbf{D}(i,j) = \langle \mathbf{G}^b(i,:), \mathbf{G}^b(j,:)\rangle / \langle \mathbf{G}^b(j,:), \mathbf{G}^b(j,:)\rangle$, *i.e.*, $\mathbf{D}(i,j)$ is the association accuracy as defined in association rule mining [1] for rule $\mathbf{G}^b(j,:) \Rightarrow \mathbf{G}^b(i,:)$. After $\mathbf{D}$ is binarized to a Boolean matrix $\mathbf{D}^b$ (see Algorithm 2), the columns of the factor matrices are selected from the columns of $\mathbf{D}^b$ in a greedy fashion. In the context of our problem with the two additional constraints, the algorithm is modified as follows. Firstly, each candidate column of $\mathbf{D}^b$ is concatenated to the current $\mathbf{Z}^b$, and the next best $\mathbf{Z}^b$ is the one that minimizes (13). Note that by virtue of the formulation, the PSD constraint is automatically satisfied. This step is repeated $K \leq K_0$ times until there is no candidate column in $\mathbf{D}^b$ left or (13) cannot be reduced anymore. Secondly, to reduce the probability that a row of $\mathbf{Z}^b$ contains multiple 1's and violates the Boolean orthonormal constraint, we only retain as candidate those columns which are sufficiently different from the selected columns (based on some threshold t_d) for the next iteration. The full details are presented in Algorithm 2, in which the input K_0 is usually selected as the rank of $\mathbf{G}$.

Since we only approximately enforce the orthonormal constraint, it is possible for a row of $\mathbf{Z}^b$ to contain multiple 1's. Usually, these constitute a very small proportion of the rows. Thus, most points can be uniquely assigned to clusters and the clusters are adequately populated. As a result, we can resolve the assignment conflict by a simple post-processing step as follows. We postpone the cluster assignment of all those points with conflicts. Assuming the resultant clustering is $\mathbf{X} = \{X_1, \ldots, X_K\}$ and that there is an unassigned data point i, we assign the point i to the group $X_{K'}$ with whose members it has the largest affinity; that is, $K' = arg\max_k \sum_{j\in X_k} \mathbf{A}(i,j)$, where $\mathbf{A}$ is the affinity matrix as defined in Section 1.

Algorithm 2 The AssoCBMF algorithm

```
Input: $\mathbf{G}$, $K_0$
  Initialize: Construct the Boolean matrix $\mathbf{G}^b$ from $\mathbf{G}$ with
  rounding threshold $t_b = 0.5$, $\mathbf{Z}^b \leftarrow [\,]$, $e = \infty, t_d = 0.1$.
  for $\tau = 0.1, 0.2, \ldots, 1$ do
    Construct $\mathbf{D}^b$ with $\mathbf{D}^b(i,j) = \frac{\langle \mathbf{G}^b(i,:),\mathbf{G}^b(j,:)\rangle}{\langle \mathbf{G}^b(j,:),\mathbf{G}^b(j,:)\rangle} > \tau$.
    for $k = 1, 2, ..., K_0$ do
      $i = arg\min_i |\mathbf{G}^b \oplus ([\mathbf{Z}^b\ \mathbf{D}^b(:,i)] \circ [\mathbf{Z}^b\ \mathbf{D}^b(:,i)]^T)|$.
      $\mathbf{Z}^b \leftarrow [\mathbf{Z}^b\ \mathbf{D}^b(:,i)]$.
      Delete all $j$-th columns with $\frac{\langle \mathbf{D}^b(:,i),\mathbf{D}^b(:,j)\rangle}{\|\mathbf{D}^b(:,i)\|\|\mathbf{D}^b(:,j)\|} > t_d$
      from $\mathbf{D}^b$.
      if $\mathbf{D}^b$ is empty or (13) is not reduced in this loop
        break
      end if
      if $\|\mathbf{G} - \mathbf{Z}^b\mathbf{Z}^{b^T}\|_F^2 < e$
        $\mathbf{Z}^{b^*} = \mathbf{Z}^b$.
        $e = \|\mathbf{G} - \mathbf{Z}^b\mathbf{Z}^{b^T}\|_F^2$.
      end if
    end for
  end for
  return $\mathbf{Z}^{b^*}$
```

5. Experiments

In this section, we compare our method with various model selection methods. In the spectral graph approach, the key to performance lies in how well one is able to determine the number of eigenvalues close to zero in the Laplacian matrix. We choose as representatives of these spectral graph methods both the basic gap heuristic (GH) method [20] as baseline, as well as one of the most robust ones—the soft thresholds (ST) method [19] which produces the best result reported in the motion segmentation problem so far. In addition to these two methods, we also compare with a model specific method—the second order difference (SOD) method [38], which reports state-of-the-art results in several datasets. A potential disadvantage of SOD is that it requires knowledge of the model; in particular, the subspace dimension is assumed known and constant. Note also that its model selection does not depend solely on the affinity matrix, hence requiring the original data as input. Since the performance of the model selection step also depends on the type of affinity matrix passed in, we also experiment with different ways of constructing the affinity matrix. We choose the two state-of-the-art algorithms in subspace clustering, SSC [12] and LRR [19] [1], to construct affinity matrices. For ST and SOD, we use the same parameter settings as in the original papers; for SCAMS, we use the fixed values of

[1] Here, by SSC and LRR, we refer only to those part of the respective algorithms that produce the affinity matrix, *i.e.*, not including the original model selection step proposed by the authors.

$\lambda = 2$ and $\gamma = 0.005$ in all the experiments.

To evaluate algorithm performance, we adopt the Rand index [26] as a measure of similarity between two data clusterings. This metric counts the pairs of points on which two clusterings agree or disagree. It is a better metric compared to the classification error rate when the number of groups is unknown. It is defined as follows.

Definition 1. *Given a set of N elements $\mathcal{V} = \{v_i\}_{i=1}^N$ and two clusterings of $\mathcal{V}$, namely $\mathbf{X} = \{X_1, \ldots, X_r\}$ with r clusters and $\mathbf{Y} = \{Y_1, \ldots, Y_s\}$ with s clusters. We define*

- *a: the number of pairs that are in the same cluster in both $\mathbf{X}$ and $\mathbf{Y}$.*
- *b: the number of pairs that are in the different clusters in both $\mathbf{X}$ and $\mathbf{Y}$.*
- *c: the number of pairs that are in the same cluster in $\mathbf{X}$ but in the different clusters in $\mathbf{Y}$.*
- *d: the number of pairs that are in the different clusters in $\mathbf{X}$ but in the same cluster in $\mathbf{Y}$.*

The Rand index, RI, is

$$RI = \frac{a+b}{a+b+c+d}. \tag{14}$$

Note that RI has a value between 0 and 1, with 0 indicating that the two data clusters do not agree on any pair of points and 1 indicating that the data clusters are exactly the same.

5.1. Synthetic data

We first investigate the performance of the various methods using synthetic data with different noise levels and varying number of groups. Similar to [31], we sample K subspaces chosen uniformly at random from d-dimensional subspaces in $\mathbb{R}^{50}$. We then sample 50 points on each subspace and normalize them to unit-norm vectors for the experiments.

5.1.1 Different noise levels

In the noise level test, we fix $K = 5$, with each group having different dimensions of $d = [2, 4, 6, 8, 10]$ respectively. The latter is to reflect model degeneracy, quite a common occurrence in real-world applications. As per [31], we perturb each unit-norm data point by adding a noisy vector chosen independently and uniformly at random on the sphere of radius ρ (noise level) in $\mathbb{R}^{50}$. We consider 11 different noise levels: $\rho = 0, 0.05, \ldots, 0.5$. The test runs 20 times and the average results are reported in the top of Figure 2.

As can be seen, despite the increasing noise, SCAMS performs consistently well (above 0.9) using either SSC or LRR to construct the affinity matrix. SOD performs less well although its performance also does not degrade much with increasing noise level. In contrast, the performances of GH and ST degrade significantly when the noise level increases. This experiment shows that SCAMS is more robust to noise. One may also notice that when the affinity matrix is provided by SSC, the RIs of all methods are somewhat off the perfect score of 1 even with the noise level at 0. This is probably because the LASSO version of SSC that we use is designed for noisy data at all levels. Unfortunately, this results in a slight loss of accuracy in the affinity matrix when the noise level is 0.

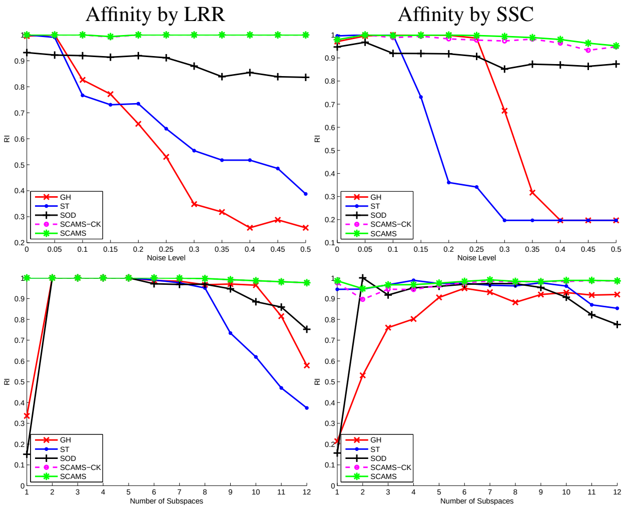


Figure 2. Comparison on the Synthetic Data. Top: RI on the synthetic data when the noise level changes. Bottom: RI on the synthetic data when the number of subspaces changes.

5.1.2 Varying group numbers

In the group number test, we fix the noise level $\rho = 0.05$ and gradually increase the group number K from 1 to 12. For a given K, each of the K groups has a different dimension d ranging from $[2, 4, \ldots, 2K]$ respectively. Note that the sum of the dimension of the subspaces is greater than the ambient dimension of 50 when $K > 6$. As K increases still more, the various subspaces become increasingly dependent, posing difficulties for the construction of affinity matrix by SSC and LRR. This raises the spectre of poor-quality affinity matrix as the number of groups increases. We again repeat the experiment 20 times and report the average results in the bottom of Figure 2.

As is evident again, SCAMS performs consistently well (above 0.9) with both versions of affinity matrix. SOD is a second order method, and its mechanism can only handle those cases when group number is greater than one. Other than this drawback, SOD again produces fairly competitive results, its performance not degrading significantly until group number exceeds 8 or 9. ST is also fairly competitive but degrades earlier when the affinity matrix is constructed by LRR. GH and SOD perform badly when the group number is 1. In general, one can say that the performances of most methods are affected by the declining quality of affinity matrices when the subspaces or groups increasingly overlap, with the effect being more pronounced in the case of LRR-constructed affinity matrix. On the other hand, some methods (notably GH) are seemingly affected by the sparser connectivity of the SSC-constructed affinity matrix, especially when the group number is small. Only our method is adequate to the handling of the varied attributes of the affinity matrices produced by different methods and under changing conditions.

To show the improvement brought about by the CBMF algorithm in Section 4, we also report the result of SCAMS using just Cholesky decomposition (SCAMS-CK) to perform the $\mathbf{G} = \mathbf{Z}\mathbf{Z}^T$ factorization. While the improvement is not significant in the case of the affinity matrix produced by LRR, it is significant when the affinity matrix is constructed by SSC and the group number is small. This performance boost is further corroborated in the later motion segmentation experiment in which CBMF improves the RI score by about 0.02.

5.2. Motion segmentation

We further evaluate the performance of SCAMS in dealing with real world problems. In this subsection, we tackle the motion segmentation problem using the *Hopkins155* [35] as dataset. This dataset comprises 155 sequences containing either two or three motions. This problem can be formulated as a subspace clustering problem, because the trajectories of a rigid motion across multiple frames lie in an affine subspace with a dimension of no more than 3, or a linear subspace with a dimension of at most 4 under the affine camera assumption [35]. In our experiments, we use the original $2F$-dimensional feature trajectories without any compression, where F is the number of frames in each sequence. The results in Table 1 report the RI scores averaged over the 155 sequences.

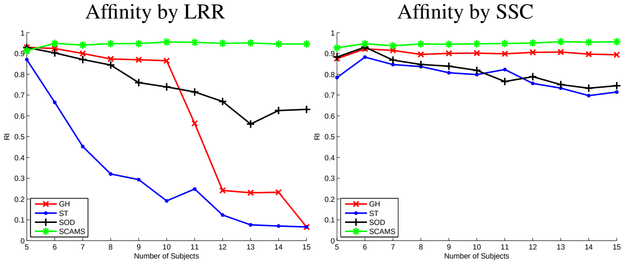
Table 1. RI on *Hopkins155*

	Affinity by LRR		Affinity by SSC	
Method	Mean	Median	Mean	Median
GH	0.6584	0.6490	0.7699	0.7418
ST	0.9154	0.9815	0.9095	0.9972
SOD	0.9026	0.9923	0.8834	0.9944
SCAMS	0.9202	0.9827	0.9068	0.9740

Since this dataset is almost noise-free and contains a small number of subspaces in each sequence, all the methods except GH perform well and there is no significant difference among these methods. GH's poor performance can be correlated with the corresponding simulation results in the preceding section. Firstly, when the affinity matrix is produced by LRR, slight noise can be detrimental to the GH method. Secondly, when the affinity matrix is produced by SSC, GH performs badly with a small group number.

5.3. Face clustering

The other real world problem that we address is the face clustering problem. In this subsection, we test the algorithms on the Extended YaleB dataset [13], which contains cropped frontal human face images of 38 subjects. Each subject has 64 images taken under different light illuminations. This problem can also be cast as a subspace clustering problem, because images of a subject with a fixed pose and varying illumination lie close to a linear subspace of dimension 9 [7]. To evaluate the performance of our algorithm, we randomly pick K subjects (K ranging from 5 to 15) and cluster the features associated with these subject images. As a preprocessing step, we resize the images to 42×48, and then use PCA to reduce the dimensionality of the vectorized raw pixel features to 30. We repeat the experiment 20 times and show the average results in Figure 3.


Affinity by LRR

Affinity by SSC

Figure 3. Comparison on the Extended YaleB dataset with increasing number of subjects.

As can be seen from Figure 3 and has been observed earlier, the affinity matrix constructed by LRR still poses problems for most methods (though to varying degrees) when the number of groups increases. In contrast, SCAMS performs consistently well (above 0.9) even when the LRR-constructed affinity matrix is not in an obliging form for most other methods. With SSC-constructed affinity matrix, all methods yield promising and more stable results, at least with respect to the number of subjects tested in this experiment. SCAMS performs consistently better than most other algorithms, with GH also turning in a stable performance. This latter phenomenon is again consistent with the results of the synthetic experiment.

6. Conclusion and Discussion

We simultaneously solve the model selection and clustering problems in a unified optimization scheme. The original structure of the affinity matrix is preserved by the Frobenius inner product (the data term) and the sparsity penalty, both terms acting locally. The rank minimization enforces global smoothness and tends to reduce the complexity of the model. These global and local considerations reveal the underlying structure of the clusters, resulting in a near-perfect

0-1 block diagonal matrix. Our highly-constrained indicator matrix formulation also has the effect of rectifying imperfections in the affinity matrix, such as filling in connectivity gap in the SSC-constructed affinity matrix. We then propose a constrained BMF to obtain a better decomposition and this in turn yields better assignments of data points. The experiments on the synthetic data as well as two real world problems show that our method performs significantly better with noisy data and large number of groups. Our experiments with both the LRR- and SSC-constructed affinity matrix reveal their different characters, and further showcase the strength of our proposed SCAMS method in handling different types of affinity matrices.

Acknowledgements. This work was partially supported by the Singapore PSF grant 1321202075 and the grant from the National University of Singapore (Suzhou) Research Institute (R-2012-N-002).

References

[1] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD*, 1993.

[2] H. Akaike. A new look at the statistical model identification. *IEEE Trans. on Automatic Control*, 19(6):716–723, 1974.

[3] C. Alzate and J. A. K. Suykens. Multiway spectral clustering with out-of-sample extensions through weighted kernel pca. *IEEE TPAMI.*, 32(2):335–347, 2010.

[4] S. Bagon and M. Galun. Large scale correlation clustering optimization. *CoRR*, abs/1112.2903, 2011.

[5] O. Banerjee, L. E. Ghaoui, and A. d'Aspremont. Model selection through sparse maximum likelihood estimation for multivariate gaussian or binary data. *Journal of Machine Learning Research*, 9:485–516, 2008.

[6] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *FOCS*, 2002.

[7] R. Basri and D. Jacobs. Lambertian reflection and linear subspaces. *IEEE TPAMI*, 25(3):218–233, 2003.

[8] A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific Symposium on Biocomputing*, 2002.

[9] S. Boyd, N. Parikh, E. Chu, and B. Peleato. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 1(3):1–122, 2011.

[10] V. Chandrasekaran, P. A. Parrilo, and A. S. Willsky. Latent variable graphical model selection via convex optimization. *The Annals of Statistics*, 40(4):1935–1967, 2012.

[11] E. Demaine and N. Immorlica. Correlation clustering with partial information. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, volume 2764 of *Lecture Notes in Computer Science*, pages 1–13. 2003.

[12] E. Elhamifar and R. Vidal. Sparse subspace clustering: Algorithm, theory, and applications. *IEEE TPAMI*, 35(11):2765–2781, 2013.

[13] 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*, 23(6):643–660, 2001.

[14] M. Goemans and D. Williamson. Improved approximation algorithms for maximum cut and satifisability probelms using semidefinite programming. *Journal of ACM*, 42.

[15] L. W. Hagen and A. B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 11(9):1074–1085, 1992.

[16] K. Kanatani. Geometric information criterion for model selection. *IJCV*, 26(3):171–189, 1998.

[17] T. Lange, V. Roth, M. Braun, and J. Buhmann. Stability based validation of clustering solutions. *Neural Computation*, 16(6):1299–1323, 2004.

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

[19] G. Liu, Z. Lin, S. Yan, J. Sun, Y. Yu, and Y. Ma. Robust recovery of subspace structures by low-rank representation. *IEEE TPAMI*, (99):1, 2012.

[20] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

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

[22] P. Miettinen. *Matrix Decomposition Methods for Data Mining: Computational Complexity and Algorithms*. PhD thesis, University of Helsinki, 2009.

[23] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, and H. Mannila. The discrete basis problem. *IEEE Trans. Knowl. Data Eng.*, 20(10):1348–1362, 2008.

[24] K. Mohan, M. J.-Y. Chung, S. Han, D. M. Witten, S.-I. Lee, and M. Fazel. Structured learning of gaussian graphical models. In *NIPS*, pages 629–637, 2012.

[25] Y. Ni, J. Sun, X. Yuan, S. Yan, and L. F. Cheong. Robust low-rank subspace segmentation with semidefinite guarantees. In *ICDM Workshops*, 2010.

[26] W. M. Rand. Objective criteria for the evaluation of clustering methods. *American Statistical Association*, 66(336):846–850, 1971.

[27] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[28] P. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Computational and Applied Math*, 20(1):53–65, 1987.

[29] Y. Shen, Z. Wen, and Y. Zhang. Augmented lagrangian alternating direction method for matrix separation based on low-rank factorization. *Optimization Methods and Software*, 29(2):239–263, 2014.

[30] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE TPAMI*, 22(8):888–905, 2000.

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

[32] S. Still and W. Bialek. How many clusters? an information theoretic perspective. *Neural Computation*, 16(12):2483–2506, 2004.

[33] R. Tibshirani, G. Walther, and T. Hastie. Estimating the number of clusters in a dataset via the gap statistic. *J. Royal Statistical Soc. B*, 63.

[34] P. H. S. Torr. Bayesian model estimation and selection for epipolar geometry and generic manifold fitting. *IJCV*, 50(1):35–61, 2002.

[35] R. Tron and R. Vidal. A benchmark for the comparison of 3-d motion segmentation algorithms. In *CVPR*, 2007.

[36] S. N. P. Vitaladevuni and R. Basri. Co-clustering of image segments using convex optimization applied to em neuronal reconstruction. In *CVPR*, pages 2203–2210, 2010.

[37] B. Wang and Z. Tu. Affinity learning via self-diffusion for image segmentation and clustering. In *CVPR*, 2012.

[38] T. Zhang, A. Szlam, Y. Wang, and G. Lerman. Hybrid linear modeling via local best-fit flats. *IJCV*, 100(3):217–240, 2012.