Simultaneous Clustering and Model Selection for Tensor Affinities

Zhuwen Li[1], Shuoguang Yang[2]*, Loong-Fah Cheong[1] and Kim-Chuan Toh[1]
[1]National University of Singapore [2]Columbia University

Abstract

Estimating the number of clusters remains a difficult model selection problem. We consider this problem in the domain where the affinity relations involve groups of more than two nodes. Building on the previous formulation for the pairwise affinity case, we exploit the mathematical structures in the higher order case. We express the original minimal-rank and positive semi-definite (PSD) constraints in a form amenable for numerical implementation, as the original constraints are either intractable or even undefined in general in the higher order case. To scale to large problem sizes, we also propose an alternative formulation, so that it can be efficiently solved via stochastic optimization in an online fashion. We evaluate our algorithm with different applications to demonstrate its superiority, and show it can adapt to varying levels of unbalancedness of clusters.

1. Introduction

In graph clustering that is conventionally set forth, it is assumed that relationship between data points can be captured by a pairwise measure of affinity. However, in many graph systems from computer vision, it does not make sense to talk about affinities between a pair of data points. We can best illustrate the issue with the classic problem of vanishing point estimation [26]: two lines trivially define a point and thus there does not seem to exist any useful measure of affinity between two lines, whereas the degree to which three lines are coincident conceivably constitutes a much more useful measure of affinity for this estimation problem. Other examples in computer vision include multi-structure fitting [34], texture segmentation [14], etc. In all these problems, clustering must be performed on the basis of higher order affinities. Therefore, hypergraph clustering has been increasingly gaining attention. Hypergraphs are made up of hyperedges, which encode higher order affinities among a subset of data whereby the set size can be more than two.

In spite of this increased interest, the problem of model selection connected with any clustering problem has generally received less attention partly because it is very hard to provide a formal definition of what a cluster is. This lacuna is even more glaring in the case of the higher order setting. Recently Li et al. [18] has proposed a simultaneous clustering and model selection (SCAMS) method to solve this problem for the case of pairwise affinities. This is based on an indicator matrix formulation, together with the generic low rank and sparsity constraints that capture the notion of good clusters without making overly strong domain-specific assumptions about the clusters. In this paper, we will develop a non-obvious generalization of SCAMS to hypergraphs. Before that, we briefly review its essential points.

Given an affinity matrix $\mathbf{A}$ with non-negative entries, SCAMS introduces the following problem:

$$\begin{aligned} \min \ & -\langle \mathbf{A}, \mathbf{G} \rangle, \\ \text{s.t.} \ & \mathbf{G} \in \mathbf{S}_+, \mathrm{diag}(\mathbf{G}) = 1, \\ & \mathrm{rank}(\mathbf{G}) = R, \mathbf{G} \in \{0,1\}^{N \times N}, \end{aligned} \tag{1}$$

where $\langle \cdot, \cdot \rangle$ is the Frobenius inner product, which is defined as $\langle \mathbf{A}, \mathbf{G} \rangle = \sum_{i,j} \mathbf{A}(i,j)\mathbf{G}(i,j)$, $\mathbf{S}_+$ is the PSD cone, $\mathrm{diag}(\cdot)$ are the diagonal entries of the matrix, R is the number of clusters, and $\mathbf{G}$ is a binary relationship matrix encoding the pairwise relationships between elements. More specifically, if the i-th and j-th elements belong to the same cluster, $\mathbf{G}(i,j) = 1$; otherwise, $\mathbf{G}(i,j) = 0$. $\mathbf{G}$ can be factorized as $\mathbf{G} = \mathbf{Z}\mathbf{Z}^T$, where $\mathbf{Z}$ is an indicator matrix whose rows indicate to which group a point belongs. Intuitively, given $\mathbf{A}$, we approximate it with $\mathbf{G}$ with ideal attributes, in which "1" means similar and "0" means different.

With the group number R being unknown, SCAMS instead solves the following problem:

$$\begin{aligned} \min \ & -\langle \mathbf{A}, \mathbf{G} \rangle + \lambda \mathrm{rank}(\mathbf{G}) + \gamma ||\mathbf{G}||_0, \\ \text{s.t.} \ & \mathbf{G} \in \mathbf{S}_+, \mathrm{diag}(\mathbf{G}) = 1, \mathbf{G} \in \{0,1\}^{N \times N}, \end{aligned} \tag{2}$$

where $||\cdot||_0$ is the ℓ_0 norm, which counts the number of nonzero elements. In this problem, the first term $-\langle \mathbf{A}, \mathbf{G} \rangle$ keeps $\mathbf{G}$ as close to the affinity matrix $\mathbf{A}$ as possible; the second term $\mathrm{rank}(\mathbf{G})$ seeks a simplest model, i.e. the clustering result with the smallest cluster number; the third term reflects the sparse nature of an ideal affinity matrix and avoids the trivial solution of $\mathbf{G}$ with entries being all ones; the PSD constraint and $\{0,1\}$ integer constraint help to discover the authentic sparse structure underlying the data, removing spurious connections and filling in missing links.

*S. Yang worked on this project as a research engineer at NUS.

There are various non-trivial difficulties when we want to generalize the preceding SCAMS algorithm to hypergraphs. Given an affinity tensor $\mathcal{A}$ with non-negative entries, a direct extension would be to solve for a binary relationship tensor $\mathcal{G}$ in a similar manner. However, even though we expect that the low rank condition to be retained for the tensor, it is in general difficult to minimize a tensor rank as conventionally defined. There is no straightforward algorithm to determine the rank of a specific given tensor; in fact, this problem is NP-hard [16]. Moreover, while it is prevalent to use the nuclear norm as a convex surrogate of a matrix rank, it is NP-hard to find the nuclear norm of a tensor rank [8]. Thus the natural generalization of matrix rank to tensor rank will not work. Another serious issue is as follows. In the formulation set forth by SCAMS, the structure of $\mathbf{G}$ arising from its specific form of $\mathbf{G} = \mathbf{Z}\mathbf{Z}^T$ is captured by the PSD constraint. In the higher order case of tensor, the PSD constraint breaks down: firstly, there is no common definitions of tensor eigenvalue and eigenvector, and even with these definitions, positive semi-definiteness can only be defined on tensors of even orders [4]. Thus, there is a fundamental difficulty to define a meaningful notion of positive semi-definiteness. Lastly, there is also the practical issue of scalability: a K-order affinity tensor with N nodes results in N^K unknowns in $\mathcal{G}$. This can easily reach a prohibitively large number, rendering the problem intractable due to both time and memory constraint. All these represent significant obstacles when we extend SCAMS to the hypergraphs.

The SCAMS formulation has revealed and exploited the particular structure present in the clustering and model selection problem for the pairwise affinity case. We believe that there are also interesting structures in the higher order case, and it is the objective of this paper to arrive at a more complete knowledge of these structures, as well as to articulate the constraints in a form that circumvents the three problems mentioned in the preceding paragraph. Specifically, we make three main contributions in solving these three problems. First, to solve the tensor rank minimization problem, we unfold the tensor in a special way so that the problem becomes a matrix rank minimization problem. Note that it is not the same as the n-rank methods which will be reviewed in Section 1.2. By leveraging on the special structure of the binary relationship tensor, we show that the rank of the matrix obtained in this way is exactly the same as the tensor rank. Next, to handle the problem of the PSD constraint, we articulate the constraint on $\mathcal{G}$ in a different way: we regard $\mathcal{G}$ as the sum of the outer-products of several 1-D tensors (vectors), i.e., $\mathcal{G} = \sum_{r=1}^{R} \mathbf{z}_r \circ \mathbf{z}_r \circ \cdots \circ \mathbf{z}_r$, where $\circ$ represents the vector outer product and $\mathbf{z}_r$ is an indicator vector. The upshot is that the constraint on $\mathcal{G}$ can then be transformed into many small-sized rank-1 matrix approximation problems with the PSD constraints. Lastly, for huge problems, we propose an alternative formulation to significantly reduce the number of unknowns based on the special structure of the binary relationship in the tensor, and most importantly, to keep the size of the unknowns constant as the order increases. We solve this problem via stochastic optimization in an online fashion without having to construct the affinity tensor in the first place, thereby bypassing any potential memory bottleneck.

1.1. Notations

We mostly follow the terminology of tensors used in [16] for our paper. An K-mode (or order/way) tensor is written as $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_K}$. For example, a vector is a 1-mode tensor and a matrix is a 2-mode tensor. The $(i_1, i_2, \cdots, i_K)$-th entry of $\mathcal{X}$ is denoted as $\mathcal{X}(i_1, i_2, \cdots, i_K)$, where $1 \leq i_k \leq I_k$ and $1 \leq k \leq K$. We also define fibers (vectors) as higher order analogue of matrix rows and columns by fixing every index but one, denoted as $\mathcal{X}(:, i_2, \cdots, i_K)$. Similarly, we define slices (matrices) as two-dimensional sections of a tensor by fixing all but two indices, denoted as $\mathcal{X}(:, :, \cdots, i_K)$.

A tensor is called cubical if every mode is of the same size, i.e., $\mathcal{X} \in \mathbb{R}^{I \times I \times \ldots \times I}$. A cubical tensor is called supersymmetric if its entries remain constant under any permutation of the indices. An K-mode tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \ldots \times I_K}$ is rank one if it can be written as the outer product of K vectors, i.e., $\mathcal{X} = \mathbf{a}^{(1)} \circ \mathbf{a}^{(2)} \circ \cdots \circ \mathbf{a}^{(K)}$. The rank of a tensor, denoted as $\mathrm{rank}(\mathcal{X})$, is defined as the smallest number of rank-1 tensors that generate $\mathcal{X}$ as their sum.

Let $\mathcal{H} = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ denote a hypergraph with $\mathcal{V} = \{v_i\}_{i=1}^{N}$ the set of the N nodes, $\mathcal{E} = \{e_i\}_{i=1}^{L}$ the set of the L hyperedges containing subsets of nodes, and $\mathcal{A} \in \mathbb{R}^{N \times N \times \ldots \times N}$ an affinity tensor storing the weights of the hyperedges. It is customary to neglect the directedness of the hyperedges, if any, and thus, it follows that $\mathcal{A}$ is supersymmetric. The weight $\mathcal{A}(e_i) \geq 0$ is an entry in $\mathcal{A}$ which represents the affinity of the nodes in the hyperedge e_i. $\mathcal{A}(e_i) = 0$ suggests that the nodes in e_i are completely dissimilar, and thus likely to be disconnected, while $\mathcal{A}(e_i) > 0$ means there is the possibility for these nodes in e_i to be clustered into the same group. The larger the value, the more likely these nodes in e_i should be in the same group. The degree of a hyperedge e_i is the number of nodes in it and denoted as $|e_i|$. The order of the corresponding affinity tensor is thus $K = \max_{i=1}^{L}(|e_i|)$.

1.2. Related Works

Current approaches to hypergraph clustering can be roughly divided into the projection and generalization methods. The projection methods [1, 36] transform the hypergraph into a graph by mapping the higher order affinities to pairwise ones, after which conventional graph clustering method (such as spectral clustering [23]) is applied. The generalization methods [17, 19, 28, 35] extend the graph clustering methods and the attendant matrix analysis to hypergraphs and tensor analysis.

Choosing the number of clusters is still a difficult model-selection problem, especially in the hypergraph setting. A general approach to this problem is based on the information-theoretic principle [13], which balances the goodness of fit against the complexity of the model. This principle is not algorithm specific, and thus can be applied to any clustering method. The major drawback of this kind of methods is that the results are often sensitive to choice of parameters. Alternatively, graph based model selection algorithms can be applied via the projection approach. That is, we first project the hypergraph onto a graph, after which, say the spectral clustering (SC) approach [27] can be used to determine the number of zero eigenvalues of the Laplacian matrix of the affinity graph. Note that the SC approach usually does not deliver the number of clusters. One has to rely on the existence of large gaps between pairs of consecutive eigenvalues to suggest that number. This gap is not always apparent unless the clusters are weakly connected to each other. What is more, when the clusters are unbalanced, the SC approach faces limitations that are well-documented [22]. In the hypergraph setting, these issues are exacerbated by the need to perform the averaging operation in the projection step—such averaging favors large clusters as vertices in these clusters have many edges joining vertices of the same cluster. Our method works in the hypergraph space directly and does not suffer from the aforementioned issues. Furthermore, our method performs a tight joint optimization of clustering and model selection in one single step, whereas the two approaches mentioned above usually adopt separate criteria to measure the goodness of the classification and to determine the number of clusters.

Our method is based on an extension of SCAMS [18] to the higher order setting. As SCAMS is related to correlation clustering (CC) [3], our method is also closely related to higher order CC [14]. The difficulty of CC or its higher order variant is in determining a proper threshold to distinguish between "similar" and "dissimilar". While [14] learns this threshold in the image segmentation task, both SCAMS and our method are completely unsupervised.

The tensor rank minimization component of our algorithm is closely related to tensor decompositions. Two early works, based on the higher-order extensions of the matrix singular value decomposition, have very much shaped research directions in this field: CANDECOMP/PARAFAC (CP) [7] decomposes a tensor as a sum of rank-1 tensors, whereas Tucker decomposition [33] is a higher-order form of principal component analysis (PCA). Building upon Tucker decomposition, the low n-rank tensor approximation [10] has many practical ramifications due to its computational feasibility. Basically, the n-rank method "unfolds" a tensor by stacking all its fibers along every direction and forms several matrices (This operation is called matricization [16]). Thus, a n-mode tensor will result in n matrices, and the n-rank method performs low rank minimization by minimizing the sum of the rank of these n matrices. Our method is different from the n-rank method in the unfolding procedure, which in our case is governed by the physical meaning of the problem.

2. Hypergraph Clustering with Unknown Group Numbers

2.1. Problem Formulation

The goal is to cluster N nodes in a hypergraph into R groups, where the group number R is unknown a priori and needs to be estimated. To formulate the problem, we denote $\{\mathbf{z}_r \in \{0,1\}^N\}_{r=1}^R$ as the indicator vectors of the R clusters, whose entries indicate if the points belong to the r-th cluster, i.e., if the i-th point belongs to the r-th cluster, $\mathbf{z}_r(i) = 1$; otherwise, $\mathbf{z}_r(i) = 0$. In the standard definition for clusters, each node must belong to at least one cluster but cannot be in multiple clusters. Thus, there is one and only one '1' at the same i-th position of all $\mathbf{z}_r$, $r = 1 \dots R$. To generalize SCAMS as presented in (1) to the tensor case, we have the following problem:

$$\begin{aligned} \min \quad & -\langle \mathcal{A}, \mathcal{G} \rangle, \\ \text{s.t.} \quad & \mathcal{G} = \sum_{r=1}^{R} \mathbf{z}_r \circ \mathbf{z}_r \circ \cdots \circ \mathbf{z}_r, \mathbf{z}_r \in \{0,1\}^N, \\ & \mathcal{G} \in \{0,1\}^{N \times N \times \cdots \times N}, \sum_{r=1}^{R} \mathbf{z}_r = \mathbf{e}, \end{aligned} \tag{3}$$

where $\mathbf{e}$ is an all one vector of size N. The last constraint ensures that one node belongs to one and only one cluster, and we call it the exclusivity constraint. Note that (1) is a special case of (3) whereby the tensor is 2-mode. One may however observe that the PSD constraint in (1) is different from the first constraint in (3) (we call it the rank-1 sum constraint). Indeed, when the tensor is 2-mode, the rank-1 sum constraint leads to the PSD constraint since the sum of PSD matrices is PSD. Even though the reverse is generally not true, it can be readily proven that, together with the $\{0,1\}$-integer constraint and the exclusivity constraint in (3), a PSD $\mathbf{G}$ can be always factorized into the form of $\mathbf{G} = \sum_{r=1}^{R} \mathbf{z}_r \circ \mathbf{z}_r$. The good news is that the PSD constraint on tensor is not really required but rather the mathematical form of $\mathbf{G} = \sum_{r=1}^{R} \mathbf{z}_r \circ \mathbf{z}_r \circ \cdots \circ \mathbf{z}_r$ should be enforced. This formulation lends itself to a more tractable solution, as we will show in the following.

Let $\{\mathbf{G}_i\}_{i=1}^{n_t}$ be all the slice matrices extracted from $\mathcal{G}$ in all directions. Note that there are in total $n_t = N^{K-2}\binom{K}{2}$ slices[1]. However, since $\mathcal{A}$ and $\mathcal{G}$ are assumed to be supersymmetric, we only need to deal with the slice matrices in one direction; i.e. $n = N^{K-2}$ slices. When $K \geq 3$, if we carefully examine the slice in $\mathcal{G}$, it can only be either a rank-1 matrix or a $\mathbf{0}$ matrix. Formally, we have

[1] $\binom{a}{b} = \frac{a!}{b!(a-b)!}$ is the Choose function.

Theorem 1. *For any supersymmetric tensor* $\mathcal{X} \in \mathbb{R}^{N\times N\times\ldots\times N}$ *of order at least 3, if it has the form*

$$\mathcal{X} = \sum_{r=1}^{R} \mathbf{z}_r \circ \mathbf{z}_r \circ \cdots \circ \mathbf{z}_r, \quad \sum_{r=1}^{R} \mathbf{z}_r = \mathbf{e}, \quad \mathbf{z}_r \in \{0,1\}^N, \tag{4}$$

any slice of $\mathcal{X}$ *can only be either a rank-1 matrix* $\mathbf{z}_r \circ \mathbf{z}_r$ *or a* $\mathbf{0}$ *matrix.*

The proof follows from the fact that any slice in $\mathcal{X}$ has the form of $\sum_{i=1}^{R} c_i \cdot \mathbf{z}_r \circ \mathbf{z}_r$, where $c_i \in \{0,1\}$ depends on the assignments of the fixed indices and $\sum_{i=1}^{R} c_i \in \{0,1\}$. The detailed proof is provided in the supplementary material.

From Theorem 1, it is clear that there are exactly $R+1$ types of slices, namely, $\mathbf{z}_1 \circ \mathbf{z}_1, \ldots, \mathbf{z}_R \circ \mathbf{z}_R$, or a $\mathbf{0}$ matrix, and each of them is a PSD matrix of rank no larger than 1. Thus, we can transform (3) to

$$\begin{aligned} \min \ & -\langle \mathcal{A}, \mathcal{G}\rangle, \\ \text{s.t. } & \operatorname{rank}(\widehat{\mathbf{G}}) = R, \operatorname{diag}(\mathcal{G}) = 1, \mathcal{G} \in \{0,1\}^{N\times\ldots\times N}, \\ & \mathbf{G}_i \in \mathbf{S}_+, \operatorname{rank}(\mathbf{G}_i) \le 1, i = 1,2,\ldots n, \end{aligned} \tag{5}$$

where $\widehat{\mathbf{G}} = [\operatorname{vec}(\mathbf{G}_1)\ \operatorname{vec}(\mathbf{G}_2)\ldots\operatorname{vec}(\mathbf{G}_n)]$ and $\operatorname{vec}(\cdot)$ vectorizes a matrix to a column vector. Note that $\mathcal{G}$, $\widehat{\mathbf{G}}$ and $\{\mathbf{G}_i\}_{i=1}^n$ are the same variable in different forms: $\mathcal{G}$ is the tensor form; $\widehat{\mathbf{G}}$ is the unfolding of the tensor $\mathcal{G}$ in matrix form; and $\{\mathbf{G}_i\}_{i=1}^n$ is the slice matrices form. It is also necessary to note that the unfolded form $\widehat{\mathbf{G}}$ is not the commonly used matricization of a tensor in n-rank estimation[15]. In (5), $\operatorname{rank}(\widehat{\mathbf{G}})$ represents model complexity. Essentially it is the number of types of rank-1 matrices $\mathbf{G}_i$, which should be equal to the number of clusters, i.e., the tensor rank of $\mathcal{G}$. Writing $\mathcal{G}$ in these various forms permits the derivation of a tractable optimization scheme.

Since R is unknown, we instead solve

$$\begin{aligned} \min \ & -\langle \mathcal{A}, \mathcal{G}\rangle + \lambda \operatorname{rank}(\widehat{\mathbf{G}}) + \gamma\|\widehat{\mathbf{G}}\|_0, \\ \text{s.t. } & \mathcal{G} \in [0,1]^{N\times N\times\ldots\times N}, \operatorname{diag}(\mathcal{G}) = 1, \\ & \mathbf{G}_i \in \mathbf{S}_+, \operatorname{rank}(\mathbf{G}_i) \le 1, i = 1,2,\ldots n, \end{aligned} \tag{6}$$

where $\|\widehat{\mathbf{G}}\|_0$ is used to avoid the trivial solution and recover the underlying sparsity structure as in SCAMS [18]. Note that the $\{0,1\}$ integer constraint is also relaxed.

2.2. Solver

For ease of representation, we let $\mathcal{W} = -\mathcal{A}$ and unfold it in the same form of $\widehat{\mathbf{G}}$ as $\widehat{\mathbf{W}}$. We solve (6) by Alternating Direction Method of Multipliers(ADMM) [5] with three block variables $\widehat{\mathbf{G}}$, $\widehat{\mathbf{H}}$ and $\{\mathbf{J}_i\}_{i=1}^n$:

$$\begin{aligned} \min \ & \langle \widehat{\mathbf{W}}, \widehat{\mathbf{G}}\rangle + \lambda \operatorname{rank}(\widehat{\mathbf{G}}) + \gamma\|\widehat{\mathbf{H}}\|_0 + g(\widehat{\mathbf{H}}), \\ \text{s.t. } & \operatorname{unfolding\_diag}(\widehat{\mathbf{H}}) = 1, \widehat{\mathbf{G}} = \widehat{\mathbf{H}}, \widehat{\mathbf{G}} = \widehat{\mathbf{J}}, \\ & \mathbf{J}_i \in \mathbf{S}_+, \operatorname{rank}(\mathbf{J}_i) \le 1, i = 1,2,\ldots n, \end{aligned} \tag{7}$$

where $\widehat{\mathbf{J}} = [\operatorname{vec}(\mathbf{J}_1)\ \operatorname{vec}(\mathbf{J}_2)\ldots\operatorname{vec}(\mathbf{J}_n)]$, 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 $\operatorname{unfolding\_diag}(\cdot)$ are those entries of the unfolded form $\widehat{\mathbf{H}}$ corresponding to the diagonal entries of the tensor.

The augmented Lagrangian function is

$$\begin{aligned} \mathcal{L} =& \langle \widehat{\mathbf{W}}, \widehat{\mathbf{G}}\rangle + \lambda \operatorname{rank}(\widehat{\mathbf{G}}) + \gamma\|\widehat{\mathbf{H}}\|_0 + g(\widehat{\mathbf{H}}) + \langle \mathbf{Y}_1, \widehat{\mathbf{G}} - \widehat{\mathbf{H}}\rangle + \\ & \frac{1}{2\mu}\|\widehat{\mathbf{G}} - \widehat{\mathbf{H}}\|_F^2 + \langle \mathbf{Y}_2, \widehat{\mathbf{G}} - \widehat{\mathbf{J}}\rangle + \frac{1}{2\mu}\|\widehat{\mathbf{G}} - \widehat{\mathbf{J}}\|_F^2, \\ \text{s.t.} & \operatorname{unfolding\_diag}(\widehat{\mathbf{H}}) = 1, \mathbf{J}_i \in \mathbf{S}_+, \\ & \operatorname{rank}(\mathbf{J}_i) \le 1, i = 1,2,\ldots n, \end{aligned} \tag{8}$$

where $\mathbf{Y}_1$ and $\mathbf{Y}_2$ are the Lagrange parameters, and $\mu > 0$ is a penalty parameter. The function can be minimized with respect to $\widehat{\mathbf{G}}$, $\widehat{\mathbf{H}}$ and $\{\mathbf{J}_i\}_{i=1}^n$ alternatingly, by fixing the other two variables, and then updating the Lagrange multipliers $\mathbf{Y}_1$ and $\mathbf{Y}_2$. The subproblems in ADMM are:

Solving G.

$$\min_{\widehat{\mathbf{G}}} \ \|\widehat{\mathbf{G}} - \frac{1}{2}(\widehat{\mathbf{H}} + \widehat{\mathbf{J}} - \mu(\widehat{\mathbf{W}} + \mathbf{Y}_1 + \mathbf{Y}_2))\|_F^2 + \lambda\mu \operatorname{rank}(\widehat{\mathbf{G}}). \tag{9}$$

Solving H.

$$\begin{aligned} \min_{\widehat{\mathbf{H}}} \ & \|\widehat{\mathbf{H}} - (\widehat{\mathbf{G}} + \mu\mathbf{Y}_1)\|_F^2 + 2\mu\gamma\|\widehat{\mathbf{H}}\|_0 + g(\widehat{\mathbf{H}}), \\ \text{s.t. } & \operatorname{unfolding\_diag}(\widehat{\mathbf{H}}) = 1. \end{aligned} \tag{10}$$

Solving J.

$$\begin{aligned} \min_{\widehat{\mathbf{J}}} \ & \|\mathbf{J}_i - (\mathbf{G}_i + \mu\mathbf{Y}_{2i})\|_F^2, \\ \text{s.t. } & \mathbf{J}_i \in \mathbf{S}_+, \ \operatorname{rank}(\mathbf{J}_i) \le 1, \end{aligned} \tag{11}$$

where $\mathbf{G}_i$ is extracted from the i-th column of $\widehat{\mathbf{G}}$ and reorganized into a square matrix. Similarly $\mathbf{Y}_{2i}$ is extracted from the i-th column of $\mathbf{Y}_2$ and reorganized. These three subproblems can be solved similarly as in [18] and all of them have closed-form solutions. The overall framework is provided in the supplementary material.

In general, there is no convergence result for ADMM applied to non-convex problems, but it often works well in practice([18, 29], *etc.*). Empirically our algorithm has strong convergence behavior indeed.

3. Constrained Boolean Tensor Decomposition

We also extend the Constrained Boolean Matrix Factorization of [18] to tensor cases, for the purpose of decomposing the tensor $\mathcal{G}$ obtained from the preceding section into its constituent indicator vectors $\mathbf{z}_1, \ldots \mathbf{z}_R$. Here "Boolean" means that the matrix/tensor contains only 0's and 1's.

Before we define our Boolean Tensor Decomposition problem, we first introduce the necessary notations and Boolean operators. We use the superscript b to distinguish Boolean variables from the normal ones. $\vee$ is the *OR* operator applied element-wise, and defined as the normal sum but with $1 + 1 = 1$. $\oplus$ is the *Exclusive-OR* operator applied element-wise, and defined as the normal sum but with $1 + 1 = 0$. Finally, we denote $|\cdot|$ as the norm of Boolean tensor and defined it as the number of 1's in it, *i.e.*, $|\mathcal{X}^b| = \sum_{i_1,i_2,\cdots,i_K} \mathcal{X}^b(i_1, i_2, \cdots, i_K)$. Now we define the problem as

Problem 1. ***Constrained Boolean Tensor Decomposition (CBTD)*** *with the rank-1 sum and exclusivity constraints. Given a Boolean tensor* $\mathcal{G}^b \in \{0,1\}^{N\times N\times\cdots\times N}$ *of order* K*, and an upper bound* R_0*, find Boolean vectors* $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_R\}$, $R \leq R_0$ *satisfying*

$$\begin{aligned} &\min\ |\mathcal{G}^b \oplus \tilde{\mathcal{G}}^b|, \\ &\text{s.t. } \tilde{\mathcal{G}}^b = \vee_{r=1}^{R} \mathbf{z}_r \circ \mathbf{z}_r \circ \cdots \circ \mathbf{z}_r, \ \sum_{r=1}^{R} \mathbf{z}_r = \mathbf{e}, \end{aligned} \tag{12}$$

where $\mathbf{e}$ is an all-one vector of length N. Note that the exclusivity constraint can be interpreted as an orthonormal constraint under Boolean algebra.

To solve this problem, we follow the Asso algorithm [21] and its constrained variant [18] via a heuristic approach of generating the candidate columns using pairwise association accuracies. More specifically, we generate a matrix $\mathbf{D}$ with its entries being the association accuracy as defined in association rule mining [2], so that the candidate vectors can be extracted from $\mathbf{D}$ and used to construct $\tilde{\mathcal{G}}^b$ in a greedy fashion. We extend the association accuracy to tensor cases as follows. Without loss of generality, let's consider the $(K-1)$-mode tensor $\tilde{\mathcal{G}}^b(i,:,\cdots,:)$. Due to the exclusivity of $\{\mathbf{z}_r\}_{r=1}^R$, there exists an unique cluster p such that $\mathbf{z}_p(i) = 1$ and $\tilde{\mathcal{G}}^b(i,:,\cdots,:) = \underbrace{\mathbf{z}_p \circ \cdots \circ \mathbf{z}_p}_{K-1}$. Similarly, there exists an unique cluster q such that $\tilde{\mathcal{G}}^b(j,:,\cdots,:) = \underbrace{\mathbf{z}_q \circ \cdots \circ \mathbf{z}_q}_{K-1}$. If $p = q$, $\langle \tilde{\mathcal{G}}^b(i,:,\cdots,:), \tilde{\mathcal{G}}^b(j,:,\cdots,:)\rangle = \|\tilde{\mathcal{G}}^b(i,:,\cdots,:)\|_F^2 = \|\tilde{\mathcal{G}}^b(j,:,\cdots,:)\|_F^2$, and $\mathbf{z}_p(i) = \mathbf{z}_p(j) = 1$; otherwise, $\langle \tilde{\mathcal{G}}^b(i,:,\cdots,:), \tilde{\mathcal{G}}^b(j,:,\cdots,:)\rangle = 0$, and $\mathbf{z}_p(j) = \mathbf{z}_q(i) = 0$, which means $\mathbf{z}_p(i) \neq \mathbf{z}_p(j)$. The key aspect of this analysis is that if $\langle \tilde{\mathcal{G}}^b(i,:,\cdots,:), \tilde{\mathcal{G}}^b(j,:,\cdots,:)\rangle = \|\tilde{\mathcal{G}}^b(j,:,\cdots,:)\|_F^2$, it is likely that $\mathbf{z}_p(i) = \mathbf{z}_p(j)$. By this intuition, we construct

$$\mathbf{D}(i,j) = \frac{\langle \mathcal{G}^b(i,:,\cdots,:), \mathcal{G}^b(j,:,\cdots,:)\rangle}{\|\tilde{\mathcal{G}}^b(j,:,\cdots,:)\|_F^2}, \tag{13}$$

which is the association accuracy for rule $\mathcal{G}^b(j,:,\cdots,:) \Rightarrow \mathcal{G}^b(i,:,\cdots,:)$. Note that $\forall i, \mathbf{D}(i,i) = 1$.

If $\mathcal{G}^b$ has the perfect form of $\sum_{r=1}^R \mathbf{z}_r \circ \mathbf{z}_r \circ \cdots \circ \mathbf{z}_r$, the set of columns of $\mathbf{D}$ should be exactly $\{\mathbf{z}_1, \ldots \mathbf{z}_R\}$. However, $\mathcal{G}^b$ is often contaminated by some noise (e.g. it is not binary) or outliers and thus is not perfect. In this case, the closer $\mathbf{D}(i,j)$ is to 1, the more likely $\mathcal{G}^b(i,:,\cdots,:)$ and $\mathcal{G}^b(j,:,\cdots,:)$ are generated by the same vector $\mathbf{z}_r$ with some noise, and the more likely that if $\mathbf{z}_r(i) = 1$, then $\mathbf{z}_r(j) = 1$, i.e., i and j belong to the same cluster r. We could set a threshold τ to get a binary matrix $\mathbf{D}^b$ which contains all the candidate items for $\{\mathbf{z}_1, \ldots \mathbf{z}_R\}$, and choose the ones that could best describe $\mathcal{G}^b$ by the greedy Algorithm.

To reduce the probability that the same position of $\mathbf{z}_r$ contains multiple 1's and violates the Boolean orthonormal (i.e. exclusivity) 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 1, in which the input R_0 is selected as the rank of $\widehat{\mathbf{G}}$.

Algorithm 1 The AssoCBTD algorithm

Input: $\mathcal{G}$, R_0.
Output: A set of indicator vectors $\mathcal{Z}^* = \{\mathbf{z}_1, \ldots, \mathbf{z}_R\}$.
Initialize: $\mathcal{Z} = \emptyset$, $e = \infty, t_d = 0.1$, and construct $\mathcal{G}^b$ from $\mathcal{G}$ with rounding threshold $t_b = 0.5$.
for $\tau = 0.1, 0.2, \ldots, 1$ **do**
  Construct $\mathbf{D}$ by (13) and obtain the Boolean matrix $\mathbf{D}^b$ with threshold τ.
  for $k = 1, 2, ..., R_0$ **do**
    $i = \arg\min_i |\mathcal{G}^b \oplus \tilde{\mathcal{G}}^b|$, where $\tilde{\mathcal{G}}^b = \vee_{\mathbf{z}\in\mathcal{Z}\bigcup \mathbf{D}^b(:,i)} \mathbf{z}\circ\mathbf{z}\circ\cdots\circ\mathbf{z}$;
    $\mathcal{Z} \leftarrow \mathbf{D}^b(:,i)$; 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 $\|\mathcal{G} - \vee_{\mathbf{z}\in\mathcal{Z}} \mathbf{z}\circ\mathbf{z}\circ\cdots\circ\mathbf{z}\|_F^2 < e$, $\mathcal{Z}^* = \mathcal{Z}$; $e = \|\mathcal{G} - \vee_{\mathbf{z}\in\mathcal{Z}} \mathbf{z}\circ\mathbf{z}\circ\cdots\circ\mathbf{z}\|_F^2$, **end if**
    if $\mathbf{D}^b$ is empty **or** (12) is not reduced in this loop, **break**, **end if**
  end for
end for
return $\mathcal{Z}^*$

Since we only approximately enforce the Boolean orthonormal constraint, it is possible for a row of $\mathcal{Z} = \{\mathbf{z}_r\}_{r=1}^R$ to contain multiple 1's in the same positions of $\mathbf{z}_r$. Usually, these constitute a very small proportion. 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 $\mathcal{Y} = \{Y_1, \ldots, Y_R\}$ and that there is an unassigned data point i, we assign the point i to the group $Y_{R'}$ with whose members it has the largest affinities; that is $R' = \arg\max_r \sum_{j\in Y_r, e\in\mathcal{E}_{ij}} \mathcal{A}(e)$, where $\mathcal{E}_{ij}$ contains all hyperedges that including node i and j and $\mathcal{A}$ is the affinity tensor as defined in Section 1.1.

4. An Alternative Solver via Stochastic Optimization

When the order of the problem increases, the number of unknowns in the previous formulation can rapidly reach a very large number. As an example, multiple homography fitting has order 5; even as few as 100 points leads to 10^{10} unknowns. What is worse, we can easily exceed the memory limit of a modern desktop computer if we pre-compute and store the affinity tensor even for problem with moderate order and number of points. This motivates us to develop an

alternative solver which is scalable to large problems.

Recall from Theorem 1 that there are exactly $R+1$ types of slices (including a $\mathbf{0}$ slice) in the optimal solution for $\mathcal{G}$. Thus, instead of solving for the tensor $\mathcal{G}$, we solve for the R nonzero slices $\mathbf{G}_j$. While R is generally unknown, we do know that R is usually a small number. Thus, we give an upper bound R_0 to R:

$$\begin{aligned} \min_{\widehat{\mathbf{G}}} \ & \frac{1}{n}\sum_{i=1}^{n}\min_{j=1}^{R_0}\langle \mathbf{W}_i, \mathbf{G}_j\rangle + \gamma\sum_{j=1}^{R_0}||\mathbf{G}_j||_0^2, \\ \text{s.t. } & \widehat{\mathbf{G}} \in \{0,1\}^{NN\times R_0}, \mathbf{G}_j \in \mathbf{S}_+, \\ & \operatorname{rank}(\mathbf{G}_j) \le 1, \ \ j = 1,...R_0, \mathbf{S}\widehat{\mathbf{G}}\mathbf{e}_{R_0} = \mathbf{e}_N, \end{aligned} \tag{14}$$

where $\mathbf{W}_i$ is a slice from $\mathcal{W}$, $\widehat{\mathbf{G}} = [vec(\mathbf{G}_1) \ vec(\mathbf{G}_2)\dots vec(\mathbf{G}_{R_0})]$, $\mathbf{e}_{R_0}$ and $\mathbf{e}_N$ are all-one vectors of size R_0 and N respectively, and $\mathbf{S} \in \{0,1\}^{N\times NN}$ is a selection matrix with $\mathbf{S}(i,(N+1)i-N) = 1$, $i = 1,...,N$ and other elements being 0. Essentially, $\mathbf{S}\text{vec}(\mathbf{G}_j)$ selects the diagonal elements of $\mathbf{G}_j$ and organise them into a column vector. Recall that Theorem 1 implies that the (i,i) diagonal elelment of each slice can be a 0 or 1; the constraint $\mathbf{S}\widehat{\mathbf{G}}\mathbf{e}_{R_0} = \mathbf{e}_N$ requires that the (i,i) element must be a 1 in at least one and only one of the slice $\mathbf{G}_j$. In (14), $\|\cdot\|_0^2$ is the square of the ℓ_0-norm. Numerically, squaring the ℓ_0-norm enlarges the range of values spanned by the second term in (14), making (14) less sensitive to the setting of γ. Compared to the unsquared ℓ_0 norm in (6), we need this stretching here because the second term only contains $R_0\times N\times N$ entries and thus its ℓ_0-norm can only sum up to $R_0\times N\times N$, whereas that in (6) can attain a much larger value of $n\times N\times N$. Note that as a consequence of the squaring, this penalty term would also favour a partitioning with more balanced clusters.

Note that in (14), $\widehat{\mathbf{G}}$ has very few columns, more exactly, R_0 of them; R_0 can be considered as an upper bound of the tensor rank of the original $\mathcal{G}$. Comparing to (6), the number of unknowns is significantly reduced (from N^2n to N^2R_0), and importantly, it does not increase as the order increases.

For notational convenience, we denote the first term in the objective function as $f(\widehat{\mathbf{G}}) = \frac{1}{n}\sum_{i=1}^{n}\ell_i(\widehat{\mathbf{G}})$, and $\ell_i(\widehat{\mathbf{G}}) = \min_{j=1}^{R_0}\left\langle \widetilde{\mathbf{W}}_i^j, \widehat{\mathbf{G}}\right\rangle$, where $\widetilde{\mathbf{W}}_i^j \in \mathbb{R}^{NN\times R_0}$ with its j-th column given by $\text{vec}(\mathbf{W}_i)$ and 0 elsewhere. Now the problem is in a form suited for optimization by the stochastic ADMM [24][2] in an online fashion. In particular, we randomly obtain one slice $\mathbf{W}_i$ from $\mathcal{W}$ for each iteration, solve the following constituent subproblems once, and then initiate the next iteration with a different slice from $\mathcal{W}$, repeating the above until convergence.

Again we introduce the intermediate variables $\widehat{\mathbf{H}}$ and $\widehat{\mathbf{J}}$ and solve

$$\begin{aligned} \min_{\widehat{\mathbf{G}}} \ & \frac{1}{n}\sum_{i=1}^{n}\ell_i(\widehat{\mathbf{G}}) + \gamma\sum_{j=1}^{R_0}||\mathbf{H}_j||_0^2, \\ \text{s.t. } & \mathbf{G}_j \in \mathbf{S}_+, \widehat{\mathbf{G}} = \widehat{\mathbf{H}}, \widehat{\mathbf{G}} = \widehat{\mathbf{J}}, \widehat{\mathbf{H}} \in \{0,1\}^{NN\times R_0}, \\ & \operatorname{rank}(\mathbf{G}_j) \le 1, \ \ j = 1,...R_0, \mathbf{S}\widehat{\mathbf{J}}\mathbf{e}_{R_0} = \mathbf{e}_N. \end{aligned} \tag{15}$$

Using the notion of approximate augmented Lagrangian defined by [24] and applying it to problem (15), we obtain

$$\begin{aligned} \mathcal{L}_k = \ & \ell_k(\widehat{\mathbf{G}}^k) + \left\langle \ell'_{k+1}(\widehat{\mathbf{G}}^k), \widehat{\mathbf{G}}\right\rangle + \gamma\sum_{j=1}^{R_0}||\mathbf{H}_j||_0^2 + \frac{1}{2\mu}||\widehat{\mathbf{G}}-\widehat{\mathbf{H}}||_F^2 + \\ & \left\langle \mathbf{Y}_1, \widehat{\mathbf{G}}-\widehat{\mathbf{H}}\right\rangle + \frac{1}{2\mu}||\widehat{\mathbf{G}}-\widehat{\mathbf{J}}||_F^2 + \left\langle \mathbf{Y}_2, \widehat{\mathbf{G}}-\widehat{\mathbf{J}}\right\rangle + \\ & \frac{1}{2\mu}||\mathbf{S}\widehat{\mathbf{J}}\mathbf{e}_{R_0}-\mathbf{e}_N||_F^2 + \left\langle \mathbf{Y}_3, \mathbf{S}\widehat{\mathbf{J}}\mathbf{e}_{R_0}-\mathbf{e}_N\right\rangle + \frac{||\widehat{\mathbf{G}}-\widehat{\mathbf{G}}^k||_F^2}{2\eta_{k+1}} \\ \text{s.t. } & \widehat{\mathbf{H}} \in \{0,1\}^{NN\times R_0}, \mathbf{G}_j \in \mathbf{S}_+, \operatorname{rank}(\mathbf{G}_j) \le 1, \ \ j = 1,...R_0, \end{aligned} \tag{16}$$

where ℓ'_{k+1} is the subgradient of ℓ_{k+1} and $||\widehat{\mathbf{G}}-\widehat{\mathbf{G}}^k||$ is the proximal term which is scaled by a time-varying stepsize η_{k+1}. The subproblems we need to solve become:

Solving G.

$$\begin{aligned} \min_{\widehat{\mathbf{G}}} \ & \|\widehat{\mathbf{G}}-\widehat{\mathbf{P}}\|_F^2, \\ \text{s.t. } & \mathbf{G}_j \in \mathbf{S}_+, \operatorname{rank}(\mathbf{G}_j) \le 1, j = 1\dots R_0, \end{aligned} \tag{17}$$

where $\widehat{\mathbf{P}} = (\frac{1}{\mu}\widehat{\mathbf{H}}+\frac{1}{\mu}\widehat{\mathbf{J}}+\frac{1}{\eta_{k+1}}\widehat{\mathbf{G}}^k-\mathbf{Y}_1-\mathbf{Y}_2-\ell'_{k+1}(\widehat{\mathbf{G}}^k))/(\frac{2}{\mu}+\frac{1}{\eta_{k+1}})$.

Solving H.

$$\begin{aligned} \min_{\widehat{\mathbf{H}}} \ & \|\widehat{\mathbf{H}}-\widehat{\mathbf{Q}}\|_F^2 + 2\mu\gamma\sum_{j=1}^{R_0}||\mathbf{H}_j||_0^2, \\ \text{s.t. } & \widehat{\mathbf{H}} \in \{0,1\}^{NN\times R_0}, \end{aligned} \tag{18}$$

where $\widehat{\mathbf{Q}} = \widehat{\mathbf{G}} + \mu\mathbf{Y}_1$.

Solving J.

$$\begin{aligned} \min_{\widehat{\mathbf{J}}} \ & \left\langle \mathbf{Y}_2, \widehat{\mathbf{G}}-\widehat{\mathbf{J}}\right\rangle + \frac{1}{2\mu}\|\widehat{\mathbf{G}}-\widehat{\mathbf{J}}\|_F^2 + \\ & \left\langle \mathbf{Y}_3, \mathbf{S}\widehat{\mathbf{J}}\mathbf{e}_{R_0}-\mathbf{e}_N\right\rangle + \frac{1}{2\mu}\|\mathbf{S}\widehat{\mathbf{J}}\mathbf{e}_{R_0}-\mathbf{e}_N\|_F^2 \end{aligned} \tag{19}$$

Since subproblem (17) can be solved in a similar manner as (11) and (19) is linear in $\widehat{\mathbf{J}}$, the only difficulty lies in (18). Evidently, (18) is separable in the columns of $\mathbf{H}_j$. We solve each column $\mathbf{H}_j$ individually by the following theorem:

Theorem 2. *Let $\mathbf{a} \in \mathbb{R}^m$ be a given vector such that its elements $a_1 \ge a_2 \ge \dots \ge a_m$. Consider problem:*

$$\mathbf{x}^* = \arg\min \|\mathbf{x}-\mathbf{a}\| + \rho(\mathbf{e}^T\mathbf{x})^2, \text{ s.t. } \mathbf{x} \in \{0,1\}^m, \tag{20}$$

where $\mathbf{e}$ is an all-one vector. The components of $\mathbf{x}^*$ are also in a descending order and there exists an integer $0 \le s \le m$ such that $x_i^* = 1$ for $i \le s$ and $x_i^* = 0$ for $i > s$.

This theorem can be proved by contradiction. The detailed proof is provided in the supplementary material.

The stochastic ADMM is close to but not exactly the same as ADMM; details of the algorithm are provided in the

[2]Note that its convergence result is not applicable due to the non-convex problems in our formulation.

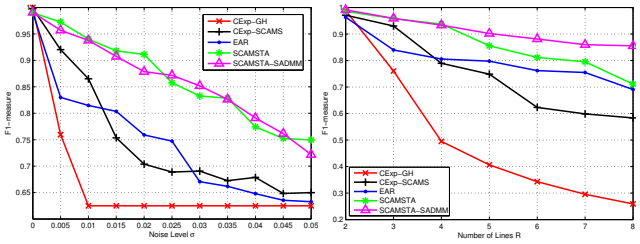
supplementary material. Since the problem is solved in an online fashion, we only need to randomly construct a slice in each iteration; thus this algorithm is memory-efficient.

5. Experiments

In this section, we compare our method with various model selection methods, including those based on pairwise affinity (acting upon projected 2D graph) and those based on higher order affinity. For the former category, we adopt the widely used clique expansion algorithm [36] to get the projected 2D graph. We then apply the basic gap heuristic (GH) [20] followed by normalized cut [30], the results of which are taken as baseline. We also choose SCAMS [18] as the representative of the state-of-the-art 2D graph methods for comparison. We denote these two methods as CExp+GH and CExp+SCAMS respectively. For the latter category, we choose the Ensembles of Affinity Relations (EAR) method [19], which is based on a generalization of the game-theoretic approach [6]. To obtain the final clusterings of [19], we need to further construct a 2D graph based on the probabilities output of this method and apply GH and normalized cut. We denote our methods as SCAMSTA and SCAMSTA-SADMM (the stochastic ADMM version). In all experiments, the upper bound R_0 is given as $R+2$, where R is the groundtruth group number. To evaluate the performance of the algorithms, we adopt the F1-measure.

5.1. Line clustering

We first investigate the performance of the various methods on the task of line clustering in the 2D space. In this task, we randomly generate lines within the region $[-0.5, 0.5]^2$, with all lines passing through the origin. To generate the affinities, we measure how well a triplet of points can be fitted to a line. Thus the order of the affinity tensor in this problem is three. To examine how noise can affect the performance of the various algorithms, we fix the number of lines to three and sample 10, 30 and 60 points from them respectively, resulting in a total of $N = 100$ points and three unbalanced clusters overlapping with each other to some extent. We perturb the sampled points by adding Gaussian noise $\mathcal{N}(0, \sigma)$, and consider 11 different noise levels: $\sigma = 0, 0.005, \cdots, 0.05$. Since larger noise leads to more ambiguous affinities, we should reduce the influence of the data term (the first term in (6)) as noise level increases. This is actually implemented by increasing the weightage of the sparsity term in (6). For this experiment, we set the parameters $\lambda = 2N$ and $\gamma = 2\sigma$ for SCAMSTA and $\gamma = 4\sigma \times 10^{-4}$ for SCAMSTA-SADMM. Similarly, we set $\lambda = 2$ and $\gamma = 2\sigma$ for SCAMS. For EAR, we set $\epsilon = \frac{1}{10}$, where the '10' is obtained from the number of points of the smallest cluster. The test runs 20 times and the average F1-measures are reported in Figure 1 (a). We also evaluate our method under varying number of lines. In this experiment, we fix $\sigma = 0.005$, and thus the parameter settings correspond to that noise level. We gradually increase the line number from 2 to 8, and the average F1-measures over 20 runs are reported in Figure 1 (b).



(a) (b)

Figure 1. Comparison on line clustering. Left: F1-measure on clustering the lines perturbed with increasing noise levels. Right: F1-measure on clustering varying number of lines.

As can be seen from Figure 1 (a), the performances of SCAMSTA and SCAMSTA-SADMM are close, and they are significantly better than those of the other methods at all noise levels. In Figure 1 (b), SCAMSTA and SCAMSTA-SADMM also consistently outperform the other methods. When we examined the estimated numbers of clusters produced by the different approaches, the performance gap is even more remarkable. We find that our approaches yield 75% correct number estimation among all test runs, while those of the other approaches are less than 30%. The reason for this performance gap is that the projection step required to produce the conventional 2D graph representation loses information. This is especially detrimental to performance when there are unbalanced clusters, as the genuine affinities in the small cluster are averaged out to small values by the clique expansion step.

5.2. Vanishing point estimation

We further evaluate our methods in dealing with real world problems. In this subsection, we tackle the vanishing point estimation problem [3] given multiple detected lines in the image plane. Since two lines always intersect at a point (perhaps at infinity), we need the third line to determine to what extent the triplet of lines intersect at a common point. Specifically, the affinity between three lines is computed as the sum of the distances from the three lines to the fitted intersection point, and the order of the affinity tensor in this problem is 3. Actually, this is a dual problem to the line clustering problem in the preceding experiment.

In this experiment, we evaluate the performance of the algorithms on the York Urban Dataset [12]. This dataset consists of 102 images with line segments hand-assigned to the vanishing points and pre-selected to conform to the Manhattan assumption [9]. It contains both indoor and outdoor scenes. Besides the other methods, we further compare with the J-Linkage clustering method [32], which is used in

[3] Under projective geometry, a set of parallel lines in 3D scene is projected onto a set of image lines meeting at a common point known as the vanishing point. Identifying these vanishing points provides strong 3D cues for inferring structure of the scene.

Table 1. Vanishing point estimation (F1-measure) on *York Urban Dataset*

Method	CExp-GH	CExp-SCAMS	EAR	J-Linkage	SCAMSTA	SCAMSTA-SADMM
Pre-selected	0.8487 ± 0.1849	0.8884 ± 0.1083	0.8434 ± 0.1515	0.7831 ± 0.1010	$\mathbf{0.9426 \pm 0.0739}$	0.9274 ± 0.0793
Detected	0.6602 ± 0.1617	0.7993 ± 0.1966	0.6570 ± 0.1837	$0.7936 \pm \mathbf{0.1504}$	$\mathbf{0.8273} \pm 0.1760$	0.8035 ± 0.1938

the state-of-the-art vanishing point estimation method [31]. However, J-Linkage usually needs a good sampling, and thus a fully computed affinity tensor (our input) – which means all possible model candidates are sampled – leads to very bad result (F1-measure < 0.6). Thus, the J-Linkage results reported here are instead obtained using the RCM sampling [25] to generate the input samples.

We first carry out our evaluation under this almost noiseless setting, for which we set parameters $\lambda = 2N$ and $\gamma = 0.05$ for SCAMSTA, $\gamma = 5 \times 10^{-5}$ for SCAMSTA-SADMM, $\lambda = 2$ and $\gamma = 0.08$ for SCAMS, the inlier threshold of J-Linkage is set to 0.05, and $\epsilon = \frac{1}{c}$ for EAR, where c is the number of points of the smallest cluster obtained from the groundtruth. The first row of Table 1 reports the F1-measures averaged over the 102 images. As can be seen, our methods significantly outperform the others, being the ones with mean F1-measure greater than 0.9 and standard deviation less than 0.1. Note that the stochastic version degrades slightly compared to the original one.

The above scenario is not very realistic as the line segments are pre-selected and hand-labeled. Thus, we next recreate a more realistic environment by having the Canny line detector generate the line segments in the images. This setting brings much more noise (e.g. orientations of lines might be perturbed) and outlier lines (e.g. lines belonging to additional vanishing points). To evaluate the algorithms, we match the detected line segments to the pre-selected ones and compute the F1-measure of only those line segments which can be matched (as only those have groundtruth). The last row of Table 1 reports the F1-measures averaged over the 102 images. The F1-measures of all methods drop due to the noise and outliers except J-Linkage, probably due to the RCM sampling. Nevertheless, both SCAMSTA and its stochastic variant perform consistently better than the other methods; SCAMSTA outperforms the best of other methods by 0.03.

5.3. Multiple homography fitting

In this subsection, we estimate multiple planar homographies on real images. Given the matched points in an image pair, at least 4 correspondences are needed to fit a homography model; a 5-th point is need to determine the goodness of the fitted model, meaning that the order in this problem is at least 5. With such an order and the attendant large number of unknowns, the ADMM version of SCAMSTA is impractical. Thus, we only compare the stochastic ADMM version with the others. Unfortunately, EAR cannot return a result in a reasonable time (*i.e.* 24 hours), and it often crashes with unknown reasons, so its result is also not listed. For a better comparison, we add the state-of-the-art multi-structure fitting method with label cost (LabelCost) [11], since multiple homography estimation is usually formulated as a multi-structure fitting problem.

Table 2. Multiple homography fitting (F1-measure) on *AdelaideRMF dataset*

Method	Label-Cost	CExp-GH	CExp-SCAMS	SCAMSTA-SADMM
mean $\pm$ variance	$0.7871 \pm$ 0.1183	$0.6980 \pm$ 0.1796	$0.7827 \pm$ 0.1925	$\mathbf{0.8418} \pm$ $\mathbf{0.1071}$

In this experiment, we evaluate the performances of the algorithms on the AdelaideRMF dataset [34]. This dataset consists of 17 image pairs with matched points available for multiple homography fitting. In each iteration of SCAMSTA-SADMM, we adopt the RCM sampling [25] to sample 3 correspondences, as $(5 - 2)$ samples are required to generate a slice. All the generated slices are also used to construct the projected 2D graph needed by other methods; using the same slices makes sure that the same amount of information are provided for each methods. We set the label cost to 7000 for all labels, parameters $\gamma = 10^{-5}$ for SCAMSTA-SADMM, and $\lambda = 2$ and $\gamma = 0.16$ for S-CAMS. The F1-measures averaged over 17 instances are reported in Table 2. It is observed that our method again outperforms the others significantly, being the only one with mean F1-measure greater than 0.8. An extra experiment on multiple fundamental matrix fitting is provided in the supplementary material, which shows similar results.

6. Conclusion

We propose a general and robust algorithm to perform simultaneous clustering and model selection given an affinity tensor as input. This is a more general setting for solving clustering problems and the proposed algorithm subsumes the previous SCAMS algorithm for the matrix case. To solve this problem, we impose the low-rank and sparsity conditions on the affinity tensor to reveal the underlying clusters hidden therein. With careful observation and design, we transform the tensor rank minimization and rank-1 sum constraint into matrix forms such that the problem becomes solvable. To tackle the scalability issue, we also provide an alternative slice formulation which lends itself to a stochastic ADMM solution. Experiments on different applications show its scalability to large problems and the advantage of our approach in dealing with unbalanced clusters and detecting small clusters.

Acknowledgements. This work was supported by Singapore PSF grant 1321202075.

References

[1] S. Agarwal, J. Lim, L. Zelnik-Manor, P. Perona, D. J. Kriegman, and S. Belongie. Beyond pairwise clustering. In *CVPR*, 2005. 2

[2] R. Agrawal, T. Imielinski, and A. N. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD*, 1993. 5

[3] N. Bansal, A. Blum, and S. Chawla. Correlation clustering. In *FOCS*, 2002. 3

[4] A. Barmpoutis, J. Ho, and B. C. Vemuri. Approximating symmetric positive semidefinite tensors of even order. *SIAM J. Imaging Sciences*, 5(1):434–464, 2012. 2

[5] 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. 4

[6] S. R. Bulò and M. Pelillo. A game-theoretic approach to hypergraph clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(6):1312–1327, 2013. 7

[7] J. Carroll and J.-J. Chang. Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition. *Psychometrika*, 35(3):283–319, 1970. 3

[8] V. Chandrasekaran, B. Recht, P. A. Parrilo, and A. S. Willsky. The convex geometry of linear inverse problems. *Foundations of Computational Mathematics*, 12(6):805–849, 2012. 2

[9] J. M. Coughlan and A. L. Yuille. Manhattan world: Orientation and outlier detection by bayesian inference. *Neural Computation*, 15(5):1063–1088, 2003. 7

[10] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1253–1278, 2000. 3

[11] A. Delong, A. Osokin, H. N. Isack, and Y. Boykov. Fast approximate energy minimization with label costs. *IJCV*, 96(1):1–27, 2012. 8

[12] P. Denis, J. H. Elder, and F. J. Estrada. Efficient edge-based methods for estimating manhattan frames in urban imagery. In *ECCV*, 2008. 7

[13] K. Kanatani. Geometric information criterion for model selection. *International Journal of Computer Vision*, 26(3):171–189, 1998. 3

[14] S. Kim, C. D. Yoo, S. Nowozin, and P. Kohli. Image segmentation using higher-order correlation clustering. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(9):1761–1774, 2014. 1, 3

[15] T. G. Kolda. Multilinear operators for higher-order decompositions. Technical Report SAND2006-2081, Sandia National Laboratories, 2006. 4

[16] T. G. Kolda and B. W. Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009. 2, 3

[17] M. Leordeanu and C. Sminchisescu. Efficient hypergraph clustering. In *AISTATS*, 2012. 2

[18] Z. Li, L. Cheong, and S. Z. Zhou. SCAMS: simultaneous clustering and model selection. In *CVPR*, 2014. 1, 3, 4, 5, 7

[19] H. Liu, L. J. Latecki, and S. Yan. Robust clustering as ensembles of affinity relations. In *NIPS*, 2010. 2, 7

[20] U. Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007. 7

[21] 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. 5

[22] B. Nadler and M. Galun. Fundamental limitations of spectral clustering. In *NIPS*, 2006. 3

[23] A. Y. Ng, M. I. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In *NIPS*, 2001. 2

[24] H. Ouyang, N. He, L. Tran, and A. G. Gray. Stochastic alternating direction method of multipliers. In *ICML*, 2013. 6

[25] P. Purkait, T. Chin, H. Ackermann, and D. Suter. Clustering with hypergraphs: The case for large hyperedges. In *ECCV*, 2014. 8

[26] C. Rother. A new approach to vanishing point detection in architectural environments. *Image Vision Comput.*, 20(9-10):647–655, 2002. 1

[27] P. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Computational and Applied Math*, 20(1):53–65, 1987. 3

[28] A. Shashua, R. Zass, and T. Hazan. Multi-way clustering using super-symmetric non-negative tensor factorization. In *ECCV*, 2006. 2

[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. 4

[30] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):888–905, 2000. 7

[31] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *ICCV*, 2009. 8

[32] R. Toldo and A. Fusiello. Robust multiple structures estimation with j-linkage. In *ECCV*, 2008. 7

[33] L. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966. 3

[34] H. S. Wong, T. Chin, J. Yu, and D. Suter. Dynamic and hierarchical multi-structure geometric model fitting. In *ICCV*, 2011. 1, 8

[35] D. Zhou, J. Huang, and B. Schölkopf. Learning with hypergraphs: Clustering, classification, and embedding. In *NIPS*, 2006. 2

[36] J. Y. Zien, M. D. F. Schlag, and P. K. Chan. Multilevel spectral hypergraph partitioning with arbitrary vertex sizes. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 18(9):1389–1399, 1999. 2, 7