A BI-CLUSTERING FRAMEWORK FOR CONSENSUS PROBLEMS*

MARIANO TEPPER† AND GUILLERMO SAPIRO†

Abstract. We consider grouping as a general characterization for problems such as clustering, community detection in networks, and multiple parametric model estimation. We are interested in merging solutions from different grouping algorithms, distilling all their good qualities into a *consensus* solution. In this paper, we propose a bi-clustering framework and perspective for reaching consensus in such grouping problems. In particular, this is the first time that the task of finding/fitting multiple parametric models to a dataset is formally posed as a consensus problem. We highlight the equivalence of these tasks and establish the connection with the computational Gestalt program, that seeks to provide a psychologically-inspired detection theory for visual events. We also present a simple but powerful bi-clustering algorithm, specially tuned to the nature of the problem we address, though general enough to handle many different instances inscribed within our characterization. The presentation is accompanied with diverse and extensive experimental results in clustering, community detection, and multiple parametric model estimation in image processing applications.

Key words. Consensus clustering, community detection, parametric model estimation, bi-clustering, matrix factorization

1. Introduction.

This paper addresses the problem of grouping data corrupted by noise and outliers. In our context, the data is a set $\mathcal{X}$ of m elements, described by

$$\mathcal{X} = \left(\bigcup_i \mathcal{X}_i \right) \cup \mathcal{O} \quad \text{with} \quad (\forall i)\mathcal{X}_i \cap \mathcal{O} = \emptyset, \tag{1.1}$$

where each $\mathcal{X}_i$ is a group (the number of groups is unknown), and $\mathcal{O}$ contains the outliers. Depending on the application context, $\mathcal{O}$ might be empty. Generically, we consider that a grouping algorithm provides a set $\mathcal{C}$ of candidate groups ($\mathcal{C} \subset \mathbb{P}(\mathcal{X})$, where $\mathbb{P}(\mathcal{X})$ is the power set of $\mathcal{X}$). The groups in $\mathcal{C}$ do not need to form a partition of $\mathcal{X}$ nor to cover $\mathcal{X}$. This general characterization subsumes many different grouping problems. For example, in this paper we address the following problems:

- If $\mathcal{C}$ is a partition of $\mathcal{X}$, then we are considering **clustering** algorithms. Traditionally, these algorithms assume $\mathcal{O} = \emptyset$. For a broad perspective on clustering techniques, we refer the reader to the overview reported in [22].
- If our dataset is a network, i.e., $\mathcal{X}$ are its vertices, then we have a **community detection** problem (notice that communities are not required to form a partition of $\mathcal{X}$). See [38, and references therein] for a comprehensive survey on community detection.
- If $\mathcal{C}$ contains a single group that was obtained by fitting a parametric model, we are dealing with a **parametric model estimation** problem.

Of course, many other problems fit into this characterization, such as image and video segmentation, or object detection in images and videos, to name just a few. Let us consider that we are provided with a pool (universe) $\{\mathcal{C}_k\}_{k=1}^c$ of c such candidate sets. These candidates might come from:

- running different grouping algorithms;
- running one algorithm with different parameters;
- running a grouping algorithm on different modalities of the same data;
- all of the above simultaneously.

*This work was partially supported by NSF, ONR, NGA, ARO, and NSSEFF.

†Department of Electrical and Computer Engineering, Duke University, Durham, NC 27708 (mariano.tepper@duke.edu, guillermo.sapiro@duke.edu)

Given the pool $\{\mathcal{C}_k\}_{k=1}^c$, we ask: which is the set of candidates $\mathcal{C}_k$ most representative of the actual grouping structure of $\mathcal{X}$? For this, one would need a criterion to select a specific set and discard all others. This has proven a rather difficult task, where even standard measures, such as modularity or normalized cuts, have known critical shortcomings [24, 27]. But why do we have to settle with selecting one solution from the pool? We argue that a better option is to combine the information of the different results in the pool into a new result consistent with them.

Recently, there have been interesting attempts to formulate this consensus problem in a sound way. Let us assume that we have a quality measure ω for each group in $\bigcup_{k=1}^c \mathcal{C}_k$. We establish a relation that says if two groups $C \in \mathcal{C}_k, C' \in \mathcal{C}_{k'}$ are mutually consistent (e.g., they do not overlap, $C \cap C' = \emptyset$). We then browse through the pool $\bigcup_{k=1}^c \mathcal{C}_k$ and build a new solution by combining the subset of mutually consistent groups with higher quality. This can be posed as a maximum-weight clique problem. This type of formulation was simultaneously introduced in [7] and [21] for image segmentation and extended for clustering [32] and community detection [45].

These two approaches, picking one candidate set $\mathcal{C}_k$ from the pool or combining all candidate sets to form a new solution, share a common issue: we need a sound and *general* quality measure (in the sense that it should allow to compare algorithms based on different principles, applied to the same data). In the clustering field, a plethora of methods to assess or classify clustering algorithms have been developed, some of them with very interesting results, e.g., [5, 10, 23, 24]. Unfortunately, the lack of a general definition (i.e., algorithm independent) makes difficult to find a unifying clustering theory and/or measure. In community detection for example, the best way to establish the community structure of a network is also an elusive task and still disputed, e.g., [27].

An alternative approach is to find and exploit the consistencies between the different grouping candidates in $\{\mathcal{C}_k\}_{k=1}^c$. Consensus/ensemble clustering is a well known family of techniques used in data analysis to solve this type of problems. Typically, the goal is to search for the so-called "mean" (or consensus) partition, i.e., the partition that is most similar, on average, to all the input partitions.

The most common form of consensus clustering involves creating an $m \times m$ co-occurrence matrix (recall m is the number of data elements), defined as

$$\mathbf{B} = \tfrac{1}{c}\sum_{k=1}^{c} \mathbf{B}_k, \quad \text{where} \quad (\mathbf{B}_k)_{ij} = \begin{cases} 1 & \text{if } (\exists C \in \mathcal{C}_k)\ i,j \in C; \\ 0 & \text{otherwise.} \end{cases} \tag{1.2}$$

There are many algorithms for analyzing $\mathbf{B}$, from simple techniques such as applying a clustering algorithm to it (e.g., k-means, hierarchical or spectral clustering), to more complex techniques. See [51] for a thorough survey of the area. In one way or another, all these techniques try to find a binary low-rank decomposition of $\mathbf{B}$ such that

$$\mathbf{B}^* = \operatorname*{argmin}_{\tilde{\mathbf{B}} \in \{0,1\}^{m\times m}} \sum_{k=1}^{c} \left\| \mathbf{B}_k - \tilde{\mathbf{B}} \right\|_F^2 = \operatorname*{argmin}_{\tilde{\mathbf{B}} \in \{0,1\}^{m\times m}} \left\| \mathbf{B} - \tilde{\mathbf{B}} \right\|_F^2 . \tag{1.3}$$

In [33], for example, the task is relaxed into a symmetric non-negative matrix factorization problem, replacing the binary constraint by non-negativity.

In the context of community detection, two works have explicitly addressed consensus within the standard framework just described. In [9], the matrix $\mathbf{B}$ is simply thresholded, and its connected components give the final result. In [28], $\mathbf{B}$ is considered as the adjacency matrix of a new weighted network. Then the following steps

are iteratively applied: (1) a unique non-deterministic algorithm is applied c times, (2) form $\mathbf{B}$ and threshold it to make it sparse, (3) stop if $\mathbf{B}$ is block diagonal, and (4) build a new network from $\mathbf{B}$ and go to (1).

The mentioned aggregation process used to build $\mathbf{B}$ involves loosing information contained in the individual matrices $\mathbf{B}_k$. In particular, only pairwise relations are conserved, while relations involving larger groups of nodes might be lost. In addition, using the average of several partitions might not be robust if some of them are of poor quality. All these methods involve working with an $m \times m$ matrix, which is highly prohibitive when the number of nodes m in the network becomes large.

Contributions. We propose a novel framework and perspective for reaching consensus in grouping problems by posing them as a bi-clustering problem.[1] The proposed approach has two main advantages: (1) all relations are conserved (instead of only keeping pairwise relations) and contribute to the consensus search, and (2) we use a much smaller matrix, rendering the problem tractable for large datasets. We also propose a new *parameterless* bi-clustering algorithm, fit for the type of matrices we analyze. We stress that our goal is not finding a better optimum for the objective function of a given grouping method, but obtaining an overall good solution via consensus search.

In addition, this is the first time that the task of finding/fitting multiple parametric models to a dataset is formally posed as a consensus/bi-clustering problem. The equivalence of these tasks is highlighted by the proposed framework and we devote special attention to explain the rationale behind this new characterization.

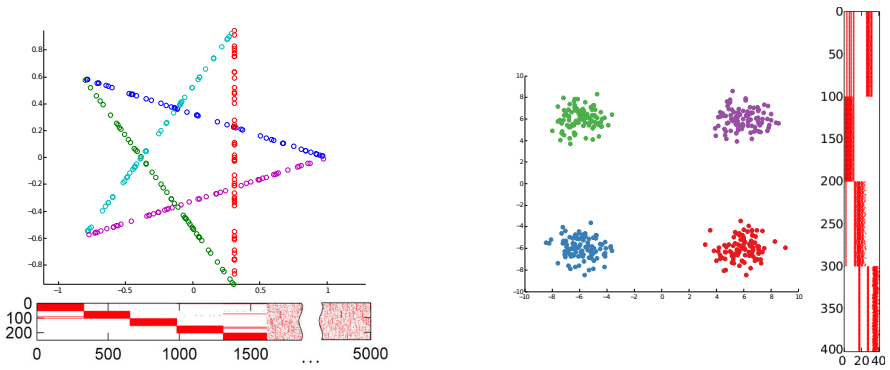
Finally, we make a formal connection with the computational Gestalt program [13], that seeks to provide a psychologically-inspired detection theory for visual events. The proposed framework allows to consider this theory from a new perspective both from the statistical and algorithmic viewpoints. We provide some insights that show the suitability of our approach as a new research direction in this field.

Organization. The remainder of this paper is organized as follows. In Section 2 we present the proposed approach for our general grouping framework. In sections 3, 4, and 5 we show how this framework applies to the problems of clustering, community detection, and multiple model parametric estimation, respectively. Each of these sections is accompanied with diverse and extensive experimental results. In Section 6 we discuss the links with the computational Gestalt theory. Finally, we provide some closing remarks in Section 7.

2. Reaching consensus by solving a bi-clustering problem. The input of the consensus algorithm is a pool $\{\mathcal{C}_k\}_{k=1}^{c}$ of candidates, that defines the universe of candidates $\mathcal{U} = \bigcup_{k=1}^{c} \mathcal{C}_k$. We also assign a weight $w_j \in \mathbb{R}^+$ to each group candidate $C_j \in \mathcal{U}$. From the data $\mathcal{X}$ and $\mathcal{U}$, we define an $m \times n$ matrix $\mathbf{A}$, whose rows and columns represent the $m = |\mathcal{X}|$ data elements and the $n = |\mathcal{U}|$ candidates, respectively; the element $(\mathbf{A})_{ij} = w_j$ if the i-th element belongs to the j-th group, and 0 otherwise. We call $\mathbf{A}$ a preference matrix. Fig. 1 presents two simplified examples.

The group weights indicate the importance assigned to each candidate and can take any form. The simplest form uses uniform weights $(\forall j)\ w_j = 1$, in which case $\mathbf{A}$ becomes a binary matrix. In this case, no prior information is used about the quality of the input group candidates. If we have such information, it can be simply incorporated in these weights.

[1] A preliminary version of this work, restricted to community detection in networks was published in the conference proceedings [46].



(a) Parametric model estimation (b) Clustering

Fig. 1: Two examples of preference matrix. (a) The data (objects) consist of 250 points on five segments (models) forming a star. The groups are potential parametric models. (b) The data (objects) consist of 400 points on four clusters. In both cases the preference matrix $\mathbf{A}$ was reordered (permuted) by group for improved visualization.

Although a clustering of elements/objects, using as features the set of sampled models they belong to, might lead to good results in certain applications, it does not fully address the problem at hand. The relationship of the objects with the sampled models is the actual focus of interest (notice the block structure of $\mathbf{A}$ in Fig. 1). A pattern-discovery algorithm is needed in this relationship space. We are thus interested in finding clusters in the product set $\mathcal{X} \times \mathcal{U}$. Such a problem is known in the literature as bi-clustering [37, and references therein], and we are therefore formally connecting it here for the first time with consensus algorithms for grouping problems.

The main contribution of this work is therefore to address the problem of grouping by bi-clustering the preference matrix $\mathbf{A}$. This provides a very intuitive rationale since, for each bi-cluster, we are jointly selecting a subset of elements and a subset of groups such that the former belongs to the latter. By directly analyzing $\mathbf{A}$ we keep all the information contained in $\mathcal{U}$ ($\mathbf{A}$ is a complete representation of $\mathcal{U}$). More classical consensus algorithms analyze an $m \times m$ matrix, see definition (1.2), while we, in contrast, work with a much smaller matrix since for common grouping problems $m \gg n$. In multiple parametric model estimation, commonly $m \ll n$ and we will show how to prune $\mathcal{U}$ to go back to the general scenario. Another important feature for analyzing very large datasets is that each base algorithm does not need to see the complete dataset. We can split the network in several (preferably overlapping) chunks, run one or more algorithms in each chunk, and let our bi-clustering formulation perform the stitching via consensus.

Notice of course that if all the base algorithms consistently make the same mistakes, these will be translated to the consensus solution, a characteristic common to all consensus algorithms, since there is no natural way to detect such consistent mistake.

2.1. Solving the bi-clustering problem. Among many tools for bi-clustering, see [37] and references therein, the Penalized Matrix Decomposition (PMD) [52] and

the Sparse Singular Value Decomposition (SSVD) [30] have shown great promise, mainly due to their conceptual and algorithmic simplicity. We are looking for interpretable row-column associations within $\mathbf{A}$. Both algorithms iterate two steps until some stopping criterion is met: (1) find one bi-cluster $\{\mathbf{u}, \mathbf{v}, s\}$, where $\mathbf{u} \in \mathbb{R}^m$, $\mathbf{v} \in \mathbb{R}^n$, $s \in \mathbb{R}^+$; (2) set $\mathbf{A} = \mathbf{A} - s\mathbf{u}\mathbf{v}^{\mathrm{T}}$. For step (1), these algorithms solve

$$\text{(PMD)} \qquad \min_{\mathbf{u},\mathbf{v},s} \left\|\mathbf{A} - s\mathbf{u}\mathbf{v}^{\mathrm{T}}\right\|_F^2 \quad \text{s.t.} \quad \begin{array}{c} \|\mathbf{u}\|_2 = 1, \|\mathbf{v}\|_2 = 1, \\ \|\mathbf{u}\|_1 \leq c_1, \|\mathbf{v}\|_1 \leq c_2, \end{array} \tag{2.1}$$

$$\text{(SSVD)} \qquad \min_{\mathbf{u},\mathbf{v},s} \left\|\mathbf{A} - s\mathbf{u}\mathbf{v}^{\mathrm{T}}\right\|_F^2 + \lambda_1 \|\mathbf{u}\|_1 + \lambda_2 \|\mathbf{v}\|_1 . \tag{2.2}$$

The sparsity-inducing constraints on $\mathbf{u}$, $\mathbf{v}$ lead to approximating $\mathbf{A}$ with only a few rows and columns of $\mathbf{u}\mathbf{v}^{\mathrm{T}}$. Let $\mathcal{R}, \mathcal{Q}$ be the active sets (the sets of nonzero elements) of $\mathbf{u}$, $\mathbf{v}$, respectively. $\mathcal{R}, \mathcal{Q}$ act as indicators of the presence of a rank-one submatrix in $\mathbf{A}$: $\mathcal{R}$ selects rows (elements), while $\mathcal{Q}$ selects columns (groups). This behavior makes both PMD and SSVD very suitable for bi-clustering.

Correctly setting the parameters $c_1, c_2, \lambda_1, \lambda_2$ is crucial, since they determine de size of the bi-clusters (via the sparsity of $\mathbf{u}, \mathbf{v}$). PMD sets c_1, c_2 via cross-validation, while SSVD uses the Bayesian information criterion. In [40] a minimum description length criterion is used to set λ_1, λ_2 and the number of iterations for SSVD. In our experiments, finding the correct values for these parameters has proven extremely challenging, since each experiment needs specifically tuned set of values. This motivates in part the development of the algorithm described next.

We propose to follow a different path for solving the bi-clustering problem at hand. Let us first notice that non-negative matrix factorization (NMF) [36] pursues a similar objective. For $1 \leq q \leq \min\{m, n\}$, NMF solves the problem

$$\min_{\mathbf{X} \in \mathbb{R}^{m \times q}, \mathbf{Y} \in \mathbb{R}^{q \times n}} \|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_F^2 \quad \text{s.t.} \quad \mathbf{X}, \mathbf{Y} \geq 0. \tag{2.3}$$

$\mathbf{A}$ is, in our application, a sparse non-negative matrix. Notice that the positivity constraints on $\mathbf{X}, \mathbf{Y}$ have a sparsifying effect on them. The intuition behind this is that when approximating a sparse non-negative matrix, the non-negative factors will only create a sparse approximation if they are themselves sparse. We thus obtain sparse factors $\mathbf{X}, \mathbf{Y}$ as in PMD and SSVD without introducing any (difficult to set) parameters. Another consequence of the sparsity of $\mathbf{A}$ is that the Frobenius norm is not entirely well suited for analyzing it. It is more appropriate to use instead an L1 fitting term,

$$\min_{\mathbf{X} \in \mathbb{R}^{m \times q}, \mathbf{Y} \in \mathbb{R}^{q \times n}} \|\mathbf{A} - \mathbf{X}\mathbf{Y}\|_1 \quad \text{s.t.} \qquad \mathbf{X}, \mathbf{Y} \geq 0. \tag{2.4}$$

With this change, we are now aiming at obtaining a "median" type of result instead of the mean, providing robustness to poor group candidates present in the pool (i.e., columns of $\mathbf{A}$ in our representation).

Any standard NMF algorithm can be adapted to use the L1 norm and solve (2.4); in this work, we modify the method in [54], that has shown good performance in practice. The algorithmic details are provided in Appendix A.

A challenge with NMF is that q is not an easy parameter to set. To avoid a cumbersome decision process, we propose to set $q = 1$ and inscribe the L1-NMF approach in an iterative loop, as with PMD and SSVD. The rank-one factorization $\mathbf{X}\mathbf{Y}$ will thus approximate a subset of $\mathbf{A}$ (because of the sparsity-inducing L1-norm),

correctly detecting a single bi-cluster. Note that this is related to a common problem in clustering known as masking, where a conceptually similar iterative procedure can address the issue [44].

2.2. Algorithmic decisions and parameters. Algorithm 1 summarizes the proposed non-negative bi-clustering approach. Notice that instead of subtracting the product $\mathbf{XY}$ from $\mathbf{A}$ as in PMD and SSVD, we set the corresponding rows and columns to zero, enforcing disjoint active sets between the successive $\mathbf{X}_t$ and $\mathbf{Y}_t$, and hence orthogonality. This also ensures that non-negativity is maintained throughout the iterations. If the bi-clusters are allowed to share elements, we do not change the rows of $\mathbf{A}$. The proposed algorithm is very efficient, simple to code, and demonstrated to work well in the experimental results that we will present later.

The iterations should stop (1) when $\mathbf{A}$ is empty (line 2), or (2) when $\mathbf{A}$ only contains noise (no structured patterns). The second case is controlled by the parameters τ_{R} and τ_{C} (line 7). These parameters determine the minimum bi-cluster size, τ_{R} rows and τ_{C} columns, necessary for being considered a structured pattern. Note that in contrast with the parameters discussed before, these are intuitive, related to the physics of the problem, and easier to set: τ_{R} encodes the minimum number of elements that a bi-cluster should contain, while τ_{C} encodes the minimum number of candidate groups that need to be in agreement to form a bi-cluster.

In theory, τ_{R} and τ_{C} depend on m, n, and the probability of having a non-zero entry in the preference matrix (the probability of success in a Bernoulli trial). It would be interesting to further explore these dependencies from a theoretical point of view. In all experiments in this paper, we set $\tau_{\mathrm{C}} = 3$. For clustering and community detection, we use $\tau_{\mathrm{R}} = 6$ for all experiments. The number and the diversity of the experiments in which these values work show that, in practice, they do not need to be carefully tuned for each case. This is due in part to the fact that in the clustering and community detection experiments, the bi-clusters are not allowed to intersect, enabling the proposed algorithm to quickly eliminate spurious entries from the preference matrix.

In the case of multiple parametric model estimation, where the bi-clusters do intersect, we set $\tau_{\mathrm{R}} = 1$ and enforce a posteriori a more strict (higher) value for it using the computational Gestalt theory. This adjustment can be done during the bi-clustering process but we decided to do it a posteriori to improve clarity and homogeneity between the different applications.

Notice that, as an alternative, we could consider the proposed bi-clustering algorithm as a lossy compression encoder; set $\tau_{\mathrm{R}} = \tau_{\mathrm{C}} = 1$, and select the first T bi-clusters that yield maximum compression (low T) with minimal loss (low fitting error).

Note. A seemingly related approach [33] involves using symmetric non-negative matrix factorization (SNMF) to cluster the co-occurrence matrix, defined in Equation (1.2), into q clusters. This is achieved by approximating problem (1.3) with

$$\min_{\substack{\mathbf{H}\in\mathbb{R}^{m\times q}\\ \mathbf{D}\in\mathbb{R}^{q\times q}}} \left\|\mathbf{B}-\mathbf{H}\mathbf{D}\mathbf{H}^{\mathrm{T}}\right\|_F^2 \quad \text{s.t.} \quad \begin{array}{l}\mathbf{H},\mathbf{D}\geq 0,\\ \mathbf{H}^{\mathrm{T}}\mathbf{H}=\mathbf{I},\end{array} \tag{2.5}$$

where $\mathbf{H}\mathbf{D}\mathbf{H}^{\mathrm{T}}$ provides a non-negative low-rank approximation of $\mathbf{B}$, and $\mathbf{D}$ is a matrix that encodes the sizes of the q obtained clusters. As in all matrix decomposition methods, the number of clusters q is a critical parameter and it is in fact one of the parameters we are interested in discovering!

Algorithm 1: Bi-clustering algorithm.

input : Preference matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, stopping parameters $\tau_{\mathrm{R}}, \tau_{\mathrm{C}} \in \mathbb{N}$
output : Bi-clusters $\{(\mathcal{R}_t, \mathcal{Q}_t)\}_{t=1}^{T}$

$T \leftarrow 0$;
while $\mathbf{A} \neq \mathbf{0}$ **do**
  $T \leftarrow T + 1$;
  Solve Problem (2.4) for $\mathbf{X}, \mathbf{Y}$ with $q = 1$;
  $\mathcal{R}_T \leftarrow \{i,\ 1 \leq i \leq m,\ (\mathbf{X})_{i,1} \neq 0\}$;
  $\mathcal{Q}_T \leftarrow \{j,\ 1 \leq j \leq n,\ (\mathbf{Y})_{1,j} \neq 0\}$;
  if $|\mathcal{R}_T| \leq \tau_R \vee |\mathcal{Q}_T| \leq \tau_C$ **then**
    $T \leftarrow T - 1$;
    break;
  if *different bi-clusters are not allowed to share rows* **then**
    $(\forall i, j)\ i \in \mathcal{R}_T, 1 \leq j \leq n,\ (\mathbf{A})_{ij} \leftarrow 0$;
  $(\forall i, j)\ 1 \leq i \leq m, j \in \mathcal{Q}_T,\ (\mathbf{A})_{ij} \leftarrow 0$;
return $\{\mathcal{R}_t\}_{t=1}^{T}$, $\{\mathcal{Q}_t\}_{t=1}^{T}$

Digging deeper, we point out that

$$\sum_{i=1}^{n} (\mathbf{A})_i (\mathbf{A})_i^{\mathrm{T}} = \mathbf{B}, \tag{2.6}$$

where $(\mathbf{A})_i$ is a column of $\mathbf{A}$ and, when bi-clusters are not allowed to share rows, by construction we have $(\forall t, t'), t \neq t', \mathcal{R}_t \cap \mathcal{R}_{t'} = \emptyset$. We thus have the same orthogonality constraints as in Equation (2.5). The proposed formulation presents all the benefits of Equation (2.5), but with increased robustness. A double averaging is present in the original problem (1.3) and thus in problem (2.5). First, Equation (2.6) acts like a pooling operator in $\mathbf{A}$, loosing critical information. Second, the Frobenius norm is known to be non-resilient to outliers. On the other hand, our formulation computes a robust median approximation to the preference matrix, which carries all needed information.

2.3. On the input pool of group candidates. There is a key ingredient in every consensus problem: the quality of the input group candidates. Performing the consensus of many extremely poor groups will not yield a good consensus solution. Among the candidates fed to any consensus algorithm, there needs to be a certain number of reasonably good groups and, at the same time, not too many bad groups. Otherwise, if the number of bad groups overwhelms the number of good groups, a masking phenomenon will occur and we will be facing an extremely hard pattern-discovery problem.

We will thus assume that, in the general case, we have a set of reasonably good candidate groups, contaminated with a few bad groups (by few we mean a number not overwhelmingly large). In the particular case of parametric model estimation, we have at our disposal more information about the nature of the groups (they are parametric) and thus employ a simple but powerful testing procedure to eliminate the vast majority of bad candidate groups. We are currently investigating how to perform similar tests in non-parametric scenarios.

The nature of the group candidates plays a non-negligible role in the consensus grouping: we would like the mistakes of the grouping algorithms to be as uncorrelated

as possible (this is a general observation that applies to all consensus approaches). The algorithms should ideally fit the data well, with their "mistakes" being caused by different factors in the data and/or different algorithmic decisions/artifacts and hence not systematically appearing over and over. This a key observation that is not trivially enforced in practice; often it is enough to use a pool of candidates that exhibit some consistency in the results while keeping some variety.

Following the presentation of our general consensus grouping framework and procedure, we now proceed to show different application contexts. We focus on clustering, community detection in networks, and multiple parametric model estimation.

3. Consensus clustering. Clustering seeks to group observations into subsets (called clusters) so that, in some sense, intra-cluster observations are more similar than inter-cluster ones. It is one of the key components of exploratory data mining and a common technique for statistical data analysis, used in such diverse fields as machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. See [22] for a broad overview of the field.

Consensus clustering is the most straightforward application of our grouping framework. Each set $\mathcal{C}_k$ of candidate groups in the pool $\{\mathcal{C}_k\}_{k=1}^{c}$ is the result of an instance of a clustering algorithm, and the preference matrix $\mathbf{A}$ is therefore constructed in a straightforward fashion.

3.1. Experimental results. In these experiments, we use uniform weights in $\mathbf{A}$. For assessing the quality of a solution when ground truth is available, we use the standard normalized mutual information (NMI) and F-measure to evaluate the results. Let $\mathcal{G}, \mathcal{C}$ be the ground truth and the evaluated solution, respectively. The normalized mutual information is defined as

$$\mathrm{NMI}(\mathcal{G}, \mathcal{C}) = 2\frac{\mathrm{H}(\mathcal{G}) + \mathrm{H}(\mathcal{C}) - \mathrm{H}(\mathcal{G}, \mathcal{C})}{\mathrm{H}(\mathcal{G}) + \mathrm{H}(\mathcal{C})}, \tag{3.1}$$

where $\mathrm{H}(\mathcal{G}), \mathrm{H}(\mathcal{C})$ are marginal entropies, and $\mathrm{H}(\mathcal{G}, \mathcal{C})$ is the joint entropy of $\mathcal{G}$ and $\mathcal{C}$. The F-measure is defined as

$$\mathrm{F}_\beta(\mathcal{G}, \mathcal{C}) = \frac{(\beta^2 + 1) \cdot \mathrm{P}_{\mathcal{G}}(\mathcal{C}) \cdot \mathrm{R}_{\mathcal{G}}(\mathcal{C})}{\beta^2 \cdot \mathrm{P}_{\mathcal{G}}(\mathcal{C}) + \mathrm{R}_{\mathcal{G}}(\mathcal{C})}, \tag{3.2}$$

where $\mathrm{P}_{\mathcal{G}}(\mathcal{C}), \mathrm{R}_{\mathcal{G}}(\mathcal{C})$ are the precision and recall rates of $\mathcal{C}$ with respect to $\mathcal{G}$. In these experiments we use $\beta = 1$, and express the F-measure as a percentage.

In figures 2 and 3 we observe two synthetic examples, where the datasets are different mixtures of Gaussians. In Figure 2 we run several instances of a Gaussian mixture model (GMM) clustering algorithm and K-means, where each instance has a different number of pre-specified clusters. In Figure 3 we run several instances of the mean-shift algorithm, each one with a different kernel size. We show the bi-clusters obtained with the proposed consensus algorithm. In both cases and without tuning any parameters, the quality of the solution approximates quite well the ground truth both qualitatively (by visual inspection) and quantitatively. In Figure 2, the consensus solution ranks second best when comparing with the base algorithms in terms of NMI and F-measure; in Figure 3, it ranks first.

It is interesting to visualize the differences between the more classical NMF, see problem (2.3), and L1-NMF, see problem (2.4). In Figure 4 we compare the active-sets of the rank-one factors $\mathbf{X}, \mathbf{Y}$ obtained with both formulations at the first iteration of Algorithm 1 for the example of Figure 2. NMF approximates in average the preference

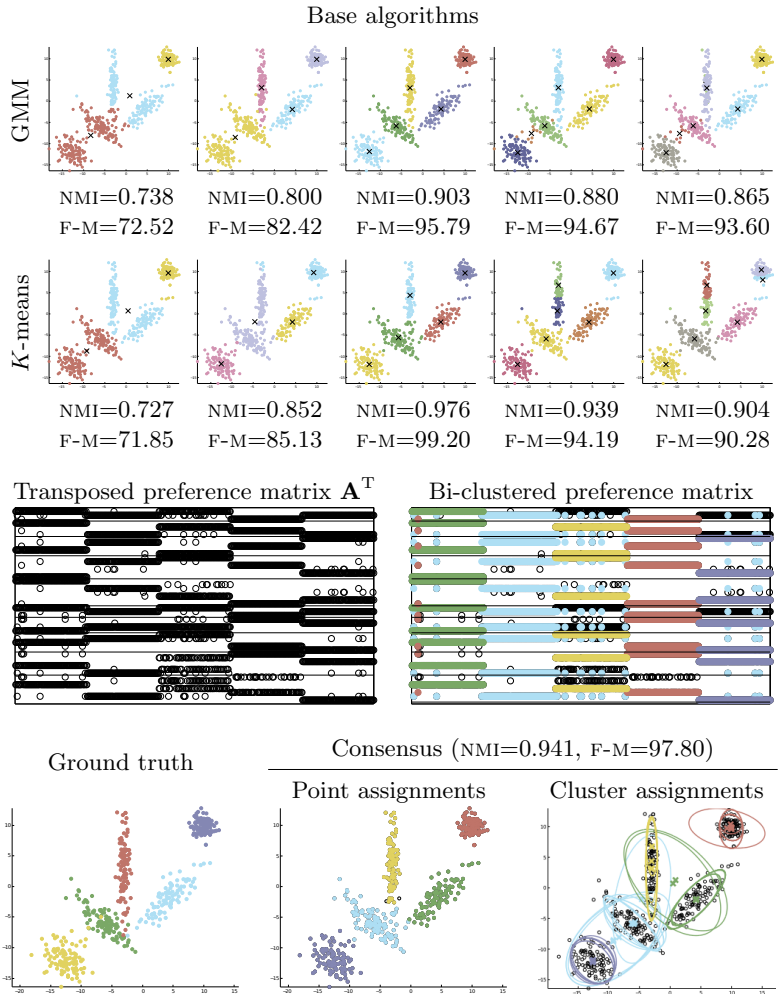


Fig. 2: Synthetic example of the proposed bi-clustering consensus computed from different instances of GMM and K-means, varying the number of classes. Each bi-cluster corresponds to jointly selected point and cluster assignments (see the bi-clustered preference matrix), we thus show with the same color its points and the covariance matrices of its constituting clusters, represented by ellipses. NMI and F-M stand for normalized mutual information and F-measure, respectively.

matrix as a whole; thus, the active-set of the analyzed bi-cluster merges information from almost all the candidate groups (i.e., its $\mathbf{Y}$ factor is non-sparse), trying to approximate more than a single group (i.e., its $\mathbf{X}$ factor is non-sparse). Contrarily, the proposed L1-NMF formulation robustly fits a subset of the preference matrix entries; it thus selects a sparse number of candidate groups and closely fits a single ground-truth group.

We compare in Figure 5 our bi-clustering approach with clustering-based consen-

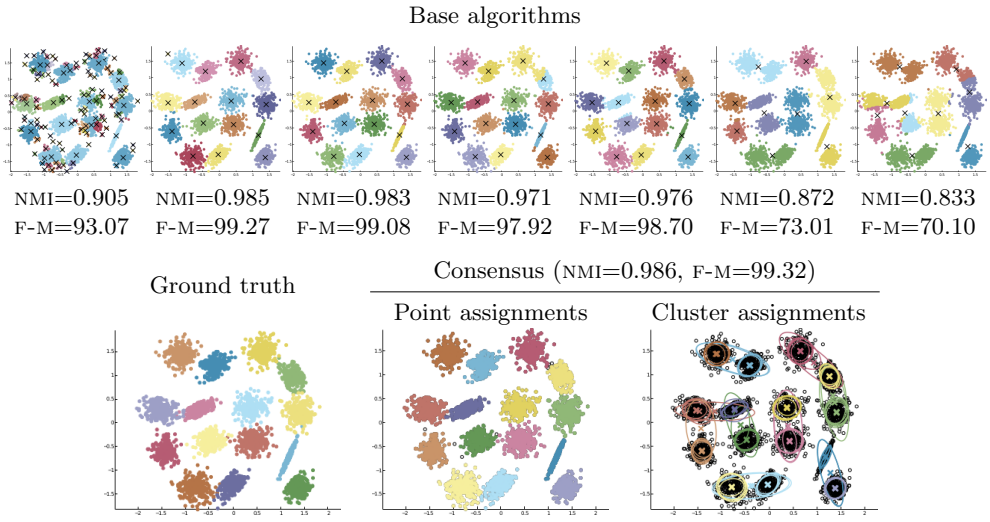


Fig. 3: Synthetic example of the proposed bi-clustering consensus computed from different mean-shift instances, varying the kernel size. The bi-clusters correspond to jointly selected point and cluster assignments (see the bi-clustered preference matrix), we thus show with the same color its points and the corresponding covariance matrix, represented by an ellipse. NMI and F-M stand for normalized mutual information and F-measure, respectively.

sus methods on a standard synthetic dataset[1]. Our first observation is that spectral clustering can obtain good results as a consensus algorithm, as long as the correct number of classes is used. Notice that this is sort of a self-defeating argument because we are introducing new (and critical) parameters for getting rid of other parameters. The same consideration is valid for SNMF [33], although in this case we obtain a very poor solution even if the correct number of classes is used (matrix $\mathbf{B}$ is not really clustered, a thresholding of $\mathbf{HDH}^{\mathrm{T}}$ being still needed, see problem (2.5)). Another clustering algorithm, J-linkage [48], popular in parametric model estimation, also yields a poor solution. Our algorithm, without tuning any parameters yields a good solution and only misclassifies a single point (this occurs because 5 out of the 8 algorithms wrongly classify that specific point).

We also present experiments on standard real datasets from the UCI repository [4] in Figure 6, where we compute the consensus solution of several instances of K-means and spectral clustering. In all four examples, we obtained results competitive with the best solutions in the pool (the best being always different for different datasets, our proposed framework is universally good). In the Iris dataset, our consensus solution is better than all of the base solutions. When the dataset contains only two classes, e.g., the Breast dataset, it becomes much harder to optimize for the number of classes since in the pool of solutions there is no balance between under and over-clustered solutions: all of them either have the right number of clusters or are over-clustered. In this case, we obtained good results by only employing algorithms that yield two or three clusters.

[1] http://www.vision.caltech.edu/lihi/Demos/SelfTuningClustering.html

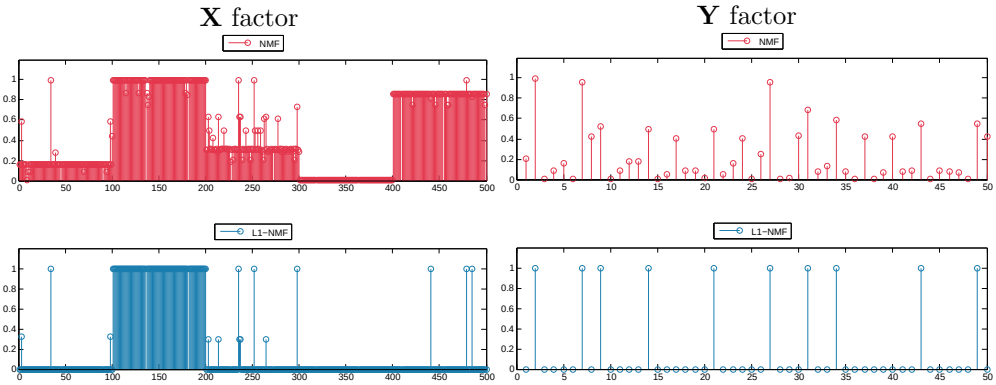


Fig. 4: Bi-cluster extraction using NMF (problem (2.3)) versus L1-NMF (problem (2.4)) in Algorithm 1. We show the active set of the first extracted bi-cluster in the example of Figure 2 (depicted there in light blue). Sparsely fitting the preference matrix, by using the L1-norm, helps to detect a bi-cluster that corresponds to a single ground-truth group, without tuning active-set thresholds.

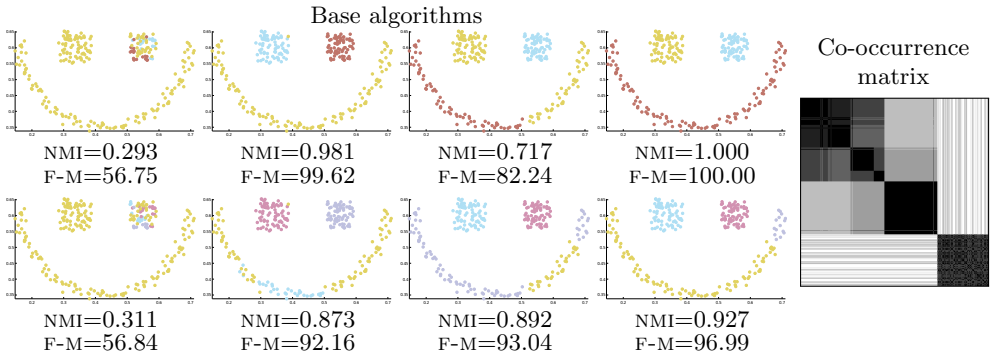
Subspace clustering. In many problems involving high-dimensional data, each class or category spans a low-dimensional subspace of the high-dimensional ambient space. The data are a set $\mathcal{X}$ of m d-dimensional data points $\{\mathbf{x}_i \in \mathbb{R}^d\}_{i=1}^m$ that lie approximately in a union of low-dimensional subspaces. Sparse subspace clustering aims at clustering these data points, finding the correct subspaces. The data are represented as a matrix $\mathbf{D} \in \mathbb{R}^{m\times d}$, whose columns are the vectors $\mathbf{x}_i$. In the case of noisy data, the underlying algorithm is defined as [14]:

1. Solve the sparse optimization problem

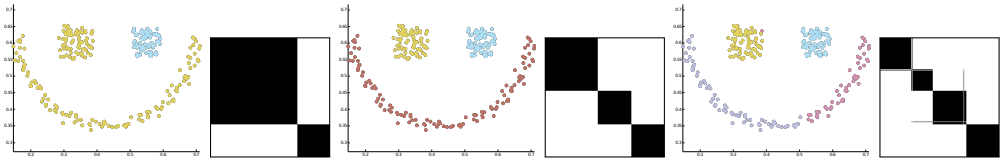
$$\min_{\mathbf{C},\mathbf{Z}} \|\mathbf{C}\|_1 + \tfrac{\kappa}{2\mu_{\mathbf{D}}} \|\mathbf{Z}\|_F^2 \quad \text{s.t.} \quad \begin{matrix} \mathbf{D} = \mathbf{D}\mathbf{C} + \mathbf{Z}, \\ \operatorname{diag}(\mathbf{C}) = \mathbf{0}, \end{matrix} \tag{3.3}$$

   where $\mu_{\mathbf{D}} \in \mathbb{R}^+$ is a normalization coefficient dependent on $\mathbf{D}$, and $\kappa \in \mathbb{R}^+$ is a parameter of the algorithm that controls how much denoising we wish to apply.
2. Normalize the columns $(\mathbf{C})_i$ of $\mathbf{C}$ as $(\mathbf{C})_i \leftarrow (\mathbf{C})_i / \|(\mathbf{C})_i\|_\infty$.
3. Form a similarity graph with m nodes representing the data points, set the affinity matrix $\mathbf{W}$ of the graph to $\mathbf{W} = |\mathbf{C}| + |\mathbf{C}^{\mathrm{T}}|$ (this is the element-wise absolute value operator, i.e., $(|\mathbf{B}|)_{ij} = |(\mathbf{B})_{ij}|$).
4. Apply spectral clustering [35] to the similarity graph, the number K of desired clusters/subspaces being a parameter of the algorithm.

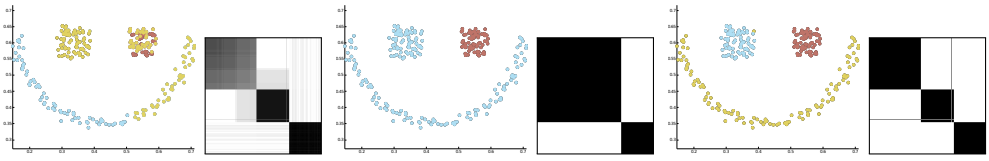
We use subspace clustering for motion segmentation in videos: given feature points on multiple rigidly moving objects tracked in multiple frames of a video, the goal is to separate the feature trajectories according to the object's motions [14]. We treat the case of subspace clustering as yet another grouping instance, therefore addressed by our proposed bi-clustering consensus approach. We run several instances of this algorithm with different values for the regularization parameter κ and compute the consensus solution from them. (We also experimented with different numbers of classes, but sparse subspace clustering is very sensitive to using the wrong number of classes.) The results of this experiment are shown in Figure 7. In this case,



(a) The base algorithms are different spectral clustering instances, obtained by varying the kernel weighting $\sigma \in \{0.005, 0.01, 0.015, 0.02\}$ for producing the similarity matrix (from left to right), and the desired number of classes $K \in \{3, 4\}$ (from top to bottom).



(b) Spectral clustering [35] (from left to right, $K = 2, 3, 4$)



(c) SNMF [33] ($q = 3$) (d) J-linkage [48] (e) Bi-clustering

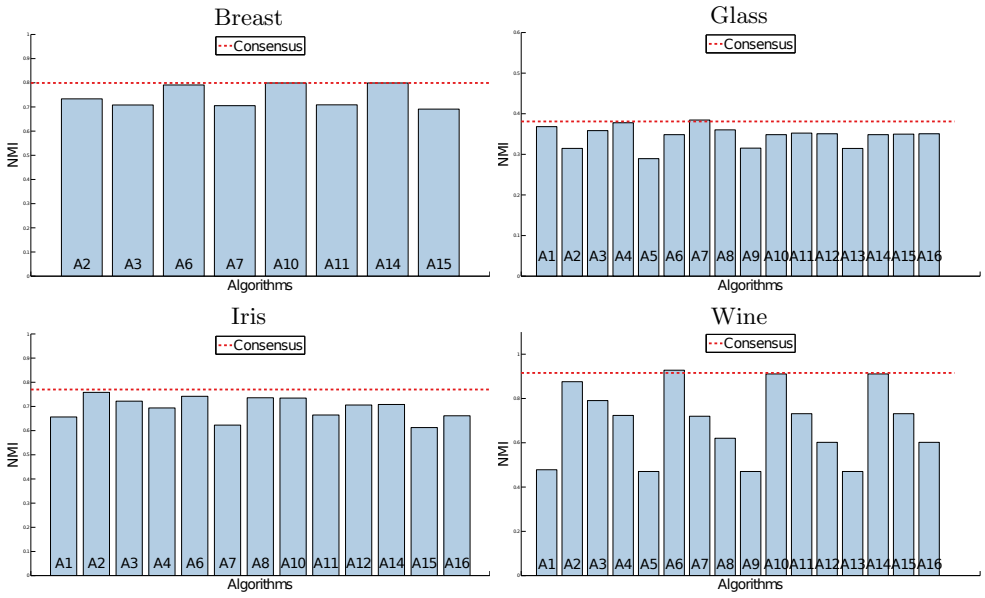
Fig. 5: Solving the consensus problem on a standard synthetic dataset [55]. (a) Results obtained with the base algorithms. (b) Clustering the co-occurrence matrix might yield good results but introduces new parameters, depending on the employed clustering algorithm. (c) SNMF yields a poor result, even when the correct number of classes is specified. (d) J-linkage yields poor results. (e) A single point is misclassified with the proposed bi-clustering approach. NMI and F-M stand for normalized mutual information and F-measure, respectively.

following the methodology in [14], we use misclassification error to analyze our results. Our first observation is that there is a range of κ values that give correct results, thus enabling the use of consensus algorithms. The next observation is that very low misclassification errors are obtained with the consensus solution, sometimes even outperforming the best individual result.

On the parameters $\tau_{\mathbf{R}}, \tau_{\mathbf{C}}$. One of the stoping criteria for Algorithm 1 is the size of the extracted bi-clusters. Each bi-cluster must have at least τ_{R} rows and τ_{C}

		Breast		Glass		Iris		Wine	
		NMI	F-M	NMI	F-M	NMI	F-M	NMI	F-M
A1	$(K_{\mathrm{GT}}-1)$-means	–	–	0.368	54.74	0.657	76.35	0.478	66.59
A2	(K_{GT})-means	0.733	95.73	0.315	49.51	0.758	89.18	0.876	96.60
A3	$(K_{\mathrm{GT}}+1)$-means	0.708	93.98	0.358	50.21	0.722	82.43	0.791	89.34
A4	$(K_{\mathrm{GT}}+2)$-means	–	–	0.378	50.30	0.694	76.29	0.724	79.84
A5	SC $(1, K_{\mathrm{GT}}-1)$	–	–	0.289	46.99	–	–	0.470	64.94
A6	SC $(1, K_{\mathrm{GT}})$	0.791	96.93	0.348	47.22	0.742	88.53	**0.928**	**98.31**
A7	SC $(1, K_{\mathrm{GT}}+1)$	0.705	93.02	**0.385**	**49.19**	0.623	77.81	0.720	82.53
A8	SC $(1, K_{\mathrm{GT}}+2)$	–	–	0.360	43.23	0.736	79.11	0.620	74.50
A9	SC $(1, K_{\mathrm{GT}}-1)$	–	–	0.315	47.48	–	–	0.470	64.94
A10	SC $(1, K_{\mathrm{GT}})$	**0.799**	**97.08**	0.348	47.22	0.735	87.82	0.911	97.74
A11	SC $(1, K_{\mathrm{GT}}+1)$	0.709	92.46	0.352	45.25	0.665	79.16	0.731	83.08
A12	SC $(1, K_{\mathrm{GT}}+2)$	–	–	0.351	42.66	0.706	78.17	0.602	75.56
A13	SC $(1, K_{\mathrm{GT}}-1)$	–	–	0.315	46.86	–	–	0.470	64.94
A14	SC $(1, K_{\mathrm{GT}})$	**0.799**	**97.08**	0.348	47.22	0.708	86.53	0.911	97.74
A15	SC $(1, K_{\mathrm{GT}}+1)$	0.691	91.97	0.350	44.75	0.613	78.20	0.731	83.08
A16	SC $(1, K_{\mathrm{GT}}+2)$	–	–	0.351	42.66	0.661	74.59	0.602	75.56
Consensus		**0.799**	**97.08**	0.381	46.89	**0.770**	**89.12**	0.915	98.01

(a) Normalized mutual information (NMI) and F-measure (F-M) values for each individual algorithm and for the proposed consensus solution. SC (σ, K) stands for spectral clustering with kernel σ and K clusters. For K-means, the value between parenthesis indicates the actual value of K. K_{GT} indicates the ground truth number of classes.

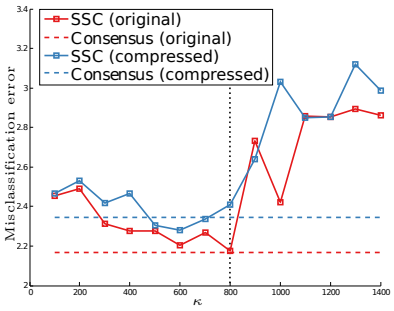


(b) Graph representation of Table (a), highlighting the quality of the proposed consensus solution.

Fig. 6: Consensus experiments on datasets from the UCI repository [4]. The proposed bi-clustering solution provides normalized mutual information (NMI) values that are universally on par with the best algorithms from the pool (which is dataset dependent and with carefully tuned parameters), without any information about the nature of the algorithms or their parameters.

	Original features			Compressed features		
κ	2	3	All	2	3	All
100	**1.46**	5.86	2.45	**1.48**	5.85	2.47
200	1.61	5.50	2.49	1.62	5.65	2.53
300	1.60	4.78	2.31	1.58	5.30	2.42
400	1.53	4.84	2.27	1.62	5.37	2.46
500	1.61	4.55	2.27	1.66	4.52	**2.30**
600	1.54	4.50	2.20	1.62	4.54	2.28
700	1.66	4.36	2.27	1.73	4.41	2.34
800	1.53	4.40	2.18	1.83	4.40	2.41
900	1.95	5.43	2.73	1.79	5.56	2.64
1000	1.79	4.58	2.42	2.36	5.34	3.03
1100	2.16	5.25	2.86	2.15	5.24	2.85
1200	2.14	5.28	2.85	2.17	5.19	2.85
1300	2.20	5.27	2.90	2.53	5.12	3.12
1400	2.19	5.15	2.86	2.33	5.24	2.99
Cons.	1.56	**4.25**	**2.17**	1.78	**4.30**	2.35

(a) Misclassification errors. In orange we observe the value of κ hand-selected in [14]. In green, we observe the result of computing the consensus solution of all the subspace clustering instances.



(b) Graph representation of Table (a) considering all instances in the dataset. The value $\kappa = 800$ is marked with a vertical dotted line. Very competitive misclassification errors are obtained with our consensus solution, without any information about the nature of the algorithms or their parameters.

Fig. 7: Subspace clustering example for the Hopkins 155 dataset [50], which consists of 155 video sequences of 2 or 3 motions, each corresponding to a low-dimensional subspace in each video. We use the original trajectories and the data projected into a lower-dimensional space using PCA. The base algorithms are different subspace clustering instances [14], varying the regularization parameter κ in (3.3). We show the mean misclassification error (%) for each of them and for our consensus solution.

Table 1: Bi-cluster sizes for several clustering experiments. Algorithm 1 returns T bi-clusters. We compare the sizes of the T-th and $(T+1)$-th bi-clusters, i.e., the last returned and the first discarded, respectively. In the cases with missing numbers, the algorithm terminated because $\mathbf{A} = \emptyset$. These sizes help to understand why there is no need to carefully tune the thresholds $\tau_{\mathrm{R}}, \tau_{\mathrm{C}}$, considering the size gap between the T-th and $(T+1)$-th bi-clusters.

	$\#\{\mathcal{R}_T\}$	$\#\{\mathcal{C}_T\}$	$\#\{\mathcal{R}_{T+1}\}$	$\#\{\mathcal{C}_{T+1}\}$
Figure 2	90	7	2	2
Figure 3	295	5	2	1
Figure 5	74	6	-	-
Figure 6 (Breast)	243	8	-	-
Figure 6 (Glass)	12	6	2	10
Figure 6 (Iris)	36	12	-	-
Figure 6 (Wine)	67	13	1	5

columns. As mentioned above, for every clustering experiment we set $\tau_{\mathrm{R}} = 6, \tau_{\mathrm{C}} = 3$. Table 1 compares the size of the T-th and $(T+1)$-th bi-clusters (recall that Algorithm 1 only returns T bi-clusters, hence discarding the $(T+1)$-th bi-cluster) for several clustering experiments. The size drop is in all cases so dramatic that $\tau_{\mathrm{R}}, \tau_{\mathrm{C}}$ become extremely easy to set.

4. Consensus community detection in networks. Networks are frequently used to describe many real-life scenarios were units interact with each other (e.g., see [1,34] and references therein). A seemingly common property to many networks is the *community structure*: networks can be divided into (in general non-overlapping) groups such that intra-group connections are denser than inter-group ones. Finding and analyzing these communities sheds light on important characteristics of the networks and the data they represent. However, the best way to establish the community structure is still disputed. Addressing this is the topic of this section.

Let $G = (V, E, \psi)$ be the graph to analyze, where V is the set of m nodes, E is the set of edges, and $\psi : E \to \mathbb{R}^+$ is a weighting function on the edges (in the following we indistinguishably use the terms graph and network). Generically, we consider that a community-detection algorithm provides a set $\mathcal{C}$ of candidate communities ($\mathcal{C} \subset \mathbb{P}(V)$, where $\mathbb{P}(V)$ is the power set of V).

4.1. Experimental Results. For the experiments we use the following base algorithms for community detection: Louvain [6], Infomap [42], and Spectral Clustering (SC-(K)) [35], where K is the number of detected clusters/communities. For assessing the quality of a solution when ground truth is available, we again use normalized mutual information (NMI). Unless specified, we use uniform weights in $\mathbf{A}$.

Several algorithms, same network. The most classical consensus scenario is when we have the result of several detection algorithms and wish to combine them into a better result.

In Figure 8, we first explore the differences between the proposed iterative rank-one L1-NMF approach and L1-NMF with $q \neq 1$. We use the Aegean34 network [15], its small size makes easy to visualize the preference matrix $\mathbf{A}$. The network models the interactions between Middle Bronze Age (MBA) Aegean archaeological sites. Our algorithm recovers the correct number of bi-clusters, a critical parameter in classical NMF approaches.

In Figure 9 we observe in detail how the bi-clustering algorithm selects entries of $\mathbf{A}$ to create a new solution, preferring regularities in the matrix, while disregarding peculiarities of individual solutions ("inpainted" nodes do not have a black circle around them in the bottom left graph in Figure 9). An important feature is that the proposed algorithm does not blindly select the best solution, but composes a consensual solution from the provided candidates.

In Table 2 we show our results using a generator of synthetic networks [29]. For this experiment, we compute the modularity $\mathrm{Mod}(\mathcal{C}_k)$ [34] of the solution $\mathcal{C}_k$ provided by the k-th base algorithm, and use a smooth increasing nonlinear function of $\mathrm{Mod}(\mathcal{C}_k)$ as the weight for each community $C \in \mathcal{C}_k$. In general, there is no community detection algorithm that "rules them all" for every network; however our algorithm consistently performs well in all examples.

Using seeds. What happens when there is not enough information in the network to recover the "correct" structure? The College football network [17] presents a very interesting such example. The network represents the matches played between teams in a season. The teams are organized in divisions, and teams should play more matches with teams from the same division than from different ones. Hence, divisions

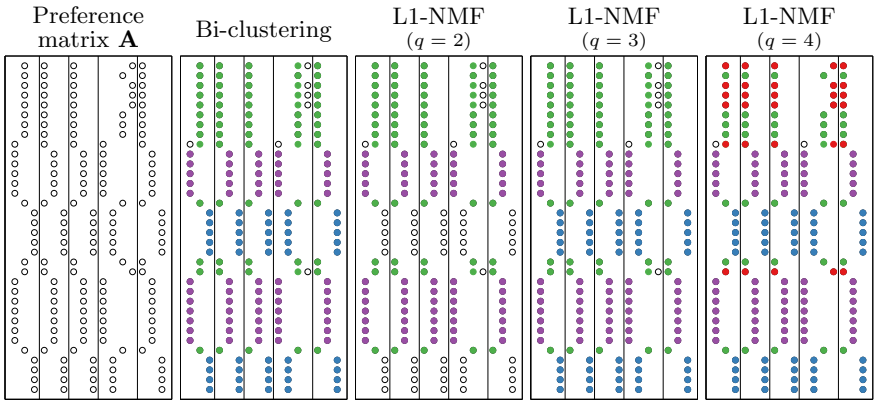


Fig. 8: The proposed iterative bi-clustering approach finds the correct number of bi-clusters (3) on the Aegean34 network [15], with 34 nodes (a different color is assigned to each pair $(\mathcal{R}_t; \mathcal{Q}_t)$, see Algorithm 1). L1-NMF, see problem (2.4), with $q = 2$ undersegments $\mathbf{A}$ (some nodes are not assigned to any community, i.e., no color, despite a lot of consistency between the base algorithms), and with $q = 4$ oversegments $\mathbf{A}$ (a single algorithm splitting a community is enough to create a new "artificial" consensus community, see the red entries).

Table 2: Results with synthetic networks (m is the number of nodes and δ the average node degree), produced with a standard benchmark generator [29]. There is no single algorithm that produces the best solution for every network; however, the consensus solution is always competitive with the best base solution (in bold). To "help" spectral clustering, K was set to the number of ground truth communities. NMI and Mod. stand for normalized mutual information and modularity, respectively.

		Louvain	Infomap	Spectral clustering				Consensus
				K	$K-1$	$K+1$	$K+2$	
$m = 10^2$	Mod.	**0.494**	0.481	0.335	0.307	0.429	0.361	0.476
$\delta = 5$	NMI	0.565	0.514	0.333	0.350	0.517	0.379	**0.584**
$m = 10^2$	Mod.	**0.436**	**0.436**	**0.436**	0.286	0.378	0.321	**0.436**
$\delta = 15$	NMI	**1.000**	**1.000**	**1.000**	0.612	0.851	0.774	**1.000**
$m = 10^3$	Mod.	**0.757**	0.753	0.612	0.651	0.618	0.601	0.750
$\delta = 5$	NMI	0.867	**0.886**	0.782	0.813	0.806	0.793	0.885
$m = 10^3$	Mod.	**0.759**	0.759	0.665	0.662	0.662	0.634	0.759
$\delta = 10$	NMI	0.995	**1.000**	0.907	0.892	0.910	0.890	**1.000**
$m = 10^4$	Mod.	**0.765**	**0.765**	0.753	0.760	0.721	0.699	**0.765**
$\delta = 300$	NMI	**1.000**	**1.000**	0.955	0.980	0.958	0.960	**1.000**
$m = 10^4$	Mod.	**0.794**	**0.794**	0.783	0.785	0.791	0.784	0.794
$\delta = 50$	NMI	**1.000**	**1.000**	0.978	0.973	0.969	0.981	0.994
Average NMI		0.904	0.900	0.826	0.770	0.835	0.796	**0.911**

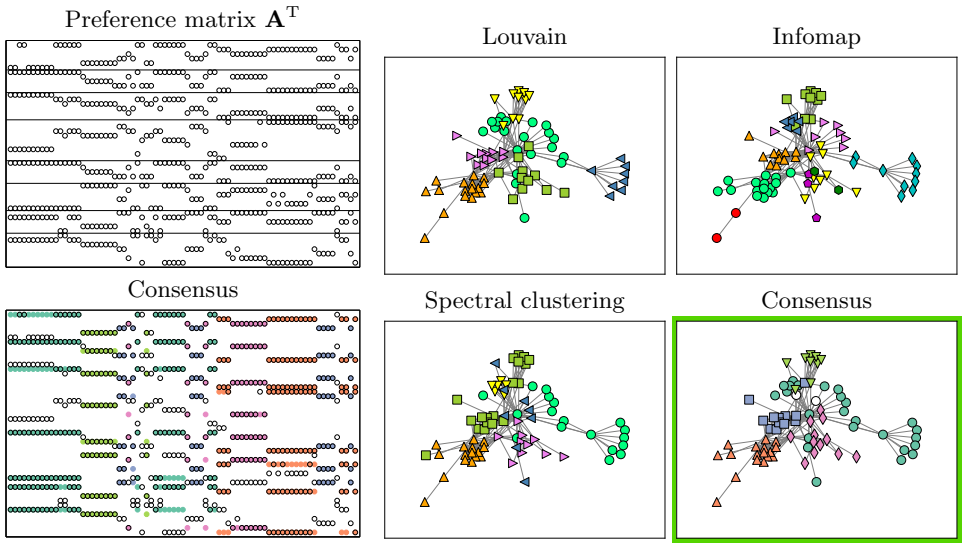
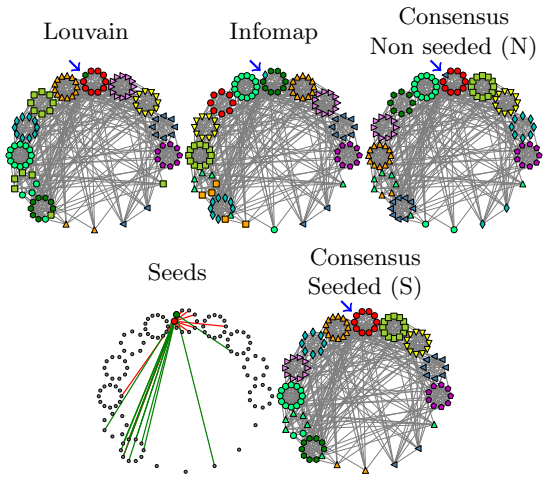


Fig. 9: Running different standard algorithms on the "Les Misérables" network with 77 nodes [25]. We show three individual results and the consensus solution. The bi-clustering result does not carbon-copy any of the individual solutions (the horizontal bands in $\mathbf{A}^{\mathrm{T}}$), but creates a new one. Also notice on the colored preference matrix (bottom left) how the algorithm "corrects" individual solutions (a different color is assigned to each pair $(\mathcal{R}_t, \mathcal{Q}_t)$, see Algorithm 1). The proposed consensus algorithm detects two nodes as singletons (in white in the bottom right graph); it is interesting to notice that the base algorithms all differ on how to treat these nodes and assign them to different communities.

are considered as ground truth communities for this network. When we run different community detection algorithms on this network, we can observe that one of the divisions is not well recovered by any of them, one of its teams being assigned to a different community, see the blue arrows in Figure 10. But in fact, when we observe this team's matches, it did not play against any of the teams in his division! We can add a tiny bit of a priori information by manually adding seeds to $\mathbf{A}$, i.e., by forcing some nodes (in Figure 10 the red and green nodes in the 4th graph) to be on the same community. For this, we just modify the corresponding rows of $\mathbf{A}$ by replacing them by their disjunction (logical or). This simple seeding mechanism is able to correct the "original mistake."

One algorithm, different networks. The proposed approach also allows to combine the results of community structure algorithms that analyze different aspects of a given network (e.g., a network with different modalities or evolving over time).

The Facebook 100 dataset presents such an example. The edges represent Facebook friendship but we can also observe several node attributes (e.g., gender, major, minor, dorm, year of graduation). In our particular example, we focus on the 2006 Duke graduates. We build two modalities of this network, by assigning different weights to the edges. In the first one, we use gender information, assigning a weight of 1 if an edge links students of different sex and of 2 otherwise. In the second one,



Algorithm	Mod.	NMI
Louvain	**0.6046**	0.8964
Infomap	0.6005	0.9345
SC-(11)	0.5859	0.9054
SC-(10)	0.5928	0.8882
SC-(12)	0.5391	0.8872
Cons. (N)	0.5999	0.9309
Cons. (S)	0.5869	**0.9425**

Fig. 10: College football network with 115 nodes [17], representing the matches between different teams. All base methods separate the node (marked with a blue arrow) from its division. By looking at its edges (in green in the 4th graph), we see that this can indeed be correct given the network. Using a little extra information, i.e., forcing the red and green nodes in the 4th graph to be in the same community, corrects this effect. As before, NMI and Mod. stand for normalized mutual information and modularity, respectively.

we use study field information, assigning a weight of 1 if the students do not share major nor minor, of 2 if they share major or minor, and of 3 if they share major and minor. We run the Louvain algorithm independently on these two networks and obviously obtain two different community structures, see Figure 11. By running our consensus algorithm on these two results, we produce a solution that aggregates information from both modalities. The proposed approach allows to perform a coherent cross-modality analysis, something not possible with traditional community detection algorithms. We can therefore expand the analysis to multimodal networks, using standard algorithms (such as Louvain) and without the need to develop new algorithms.

Another interesting example occurs when the network connectivity changes over time or when different modalities exhibit different edge sets. This is important for example when the graph is obtained through inference, because differences and/or errors in the inference process might yield different connectivities. We simulate such an example by taking a network and building 10 perturbed copies, randomly reassigning a subset of its edges. We then run Infomap on each copy and compare the result in terms of NMI with the Infomap result on the original network (Table 3). When a small portion of edges is perturbed, the best individual solution is still good, because there is a non-negligible chance that one of the perturbations does not alter the community structure of the original network. This is no longer true as the number of perturbed edges increases. Our algorithm is able to balance out the peculiarities of the perturbed solutions, obtaining a solution much closer to the original one and hence more resilient to perturbations.

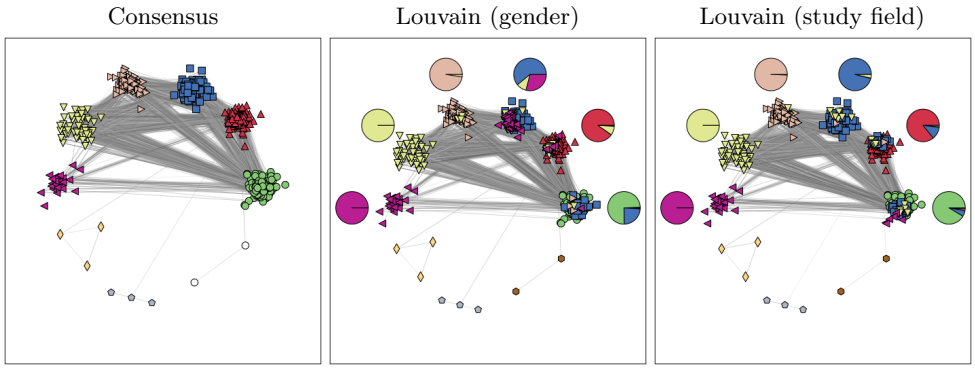


Fig. 11: Network of Facebook links between 2006 Duke graduates with 1424 nodes (part of the Facebook 100 dataset [49]). We build two different modalities by assigning weights to the edges according to different node (student) features: field of study and gender. We find a consensus between the two different results of the Louvain algorithm. The pie charts represent the distribution of the nodes of each modality's communities with respect to the consensus.

Table 3: Results of perturbing the edge set of the US politics books network with 105 nodes (`http://www.orgnet.com/divided.html`): $\rho|E|$ edges are exchanged at random. We provide average NMI values across 1000 trials, using 10 perturbed networks per trial. The result of Infomap in the original network is considered the ground truth. The consensus solution outperforms all individual ones, with the performance gap increasing as more edges are perturbed.

ρ	Best	Median	Consensus
0.1	0.9224	0.8388	**0.9258**
0.25	0.8004	0.6944	**0.8551**
0.3	0.7428	0.6315	**0.8244**
0.35	0.6801	0.5581	**0.7933**

5. Multiple parametric model estimation. This section addresses the problem of fitting multiple instances of a parametric model to data corrupted by noise and outliers, formally connecting it with bi-clustering and consensus. In our context, the data is a set $\mathcal{X}$ of m geometric objects, described by $\mathcal{X} = (\bigcup_i \mathcal{X}_i) \cup \mathcal{O}$, with $(\forall i)\mathcal{X}_i \cap \mathcal{O} = \emptyset$, and $\mathcal{O}$ being once again outliers as detailed next. The objects in each subset $\mathcal{X}_i$, which might intersect, are (noisy) measurements that can be described with a parametric model $\mu(\theta^{(i)})$, where $\theta^{(i)}$ is a parameter vector. In the following, we say that the objects in $\mathcal{X}_i$ are inliers to the model $\mu(\theta^{(i)})$. We also generally refer to a set of objects as inliers, in the sense that it exists a statistically meaningful model that describes them. The objects in $\mathcal{O}$ cannot be described with any of the models $\mu(\theta^{(i)})$, and we refer to them as outliers.

Let us define more formally these intuitive concepts. A model μ is defined as the zero level set of a smooth parametric function $f_\mu(x;\theta)$,

$$\mu(\theta) = \{\mathbf{x} \in \mathbb{R}^d,\ f_\mu(\mathbf{x};\theta) = 0\}, \tag{5.1}$$

where θ is a parameter vector. We define the error associated with the datum $\mathbf{x} \in \mathcal{X}$ with respect to the model $\mu(\theta)$ as

$$\mathrm{e}_\mu(\mathbf{x}, \theta) = \min_{\mathbf{x}' \in \mu(\theta)} \mathrm{dist}(\mathbf{x}, \mathbf{x}'), \tag{5.2}$$

where dist is an appropriate distance function. Using this error metric, we define the Consensus Set (CS) of a model as

$$\mathcal{C}(\theta) = \{\mathbf{x} \in \mathcal{X},\ \mathrm{e}_\mu(\mathbf{x}, \theta) \leq \delta\}, \tag{5.3}$$

where δ is a threshold that accounts for the measurement noise.

The goal of this section is to find the set of inliers-model pairs $\{(\mathcal{X}_i, \theta^{(i)})\}$ from the observed $\mathcal{X}$ such that $\mathcal{X}_i = \mathcal{C}(\theta^{(i)})$. This problem is, by itself, ill-posed. It is standard in the literature to implicitly or explicitly impose a penalty on the number of recovered pairs to render it tractable. We also adopt such a design choice. Notice that once $\mathcal{X}_i$ is found, the corresponding $\theta^{(i)}$ can be estimated for example by simple least-squares regression, i.e.,

$$\hat{\theta}^{(i)} = \operatorname*{argmin}_\theta \sum_{\mathbf{x} \in \mathcal{X}_i} \left[\mathrm{e}_\mu(\mathbf{x}, \theta)\right]^2. \tag{5.4}$$

This is an important but difficult problem, as standard robust estimators, like RANSAC (RANdom SAmple Consensus) [11,16], are designed to extract a single model. Let us then begin by formally explaining how does the RANSAC machinery work, to further illustrate the value and perspective of our proposed multi-model formulation.

Let us denote by b the minimum number of elements necessary to uniquely characterize a given parametric model, e.g., $b = 2$ for lines and $b = 3$ for circles. For example, if we want to discover alignments in a 2D point cloud, the elements are 2D points, models μ are lines, and $b = 2$, since a line is defined by two points. A set of b objects is called a minimal sample set (MSS). RANSAC randomly samples n MSSs, each generating a model hypothesis. The number n is an overestimation of the number of trials needed to obtain a certain number of "good" models [16, 48, 58]. Then RANSAC computes the CS of each model hypothesis using Equation (5.3). Algorithm 2 describes the standard RANSAC procedure.

Applying RANSAC sequentially, removing the inliers from the dataset as each model instance is detected, has been proposed as a solution for multi-model estimation, e.g., [39]. However, this approach is known to be suboptimal [58]. The multi-RANSAC algorithm [58] provides a more effective alternative, although the number of models must be known a priori, imposing a very limiting constraint in many applications. An alternative approach consists of finding modes in a parameter space. The overall idea is that we can map the data into the parameter space through random sampling, and then seek the modes of the distribution by discretizing the distribution, i.e., using the Randomized Hough Transform [53], or by using non-parametric density estimation techniques, like the mean-shift clustering algorithm [43]. These, however, are not intrinsically robust techniques, even if they can be robustified with outliers rejection heuristics [48]. Moreover, the choice of the parametrization is critical, among other important shortcomings [48]. The computational cost of these techniques can be very high as well.

All these techniques share a common high-level conceptual approach to model estimation: in order to solve the problem, objects are clustered. In this work, we propose an alternative formulation that involves as before *bi-clustering* the objects *and* the

Algorithm 2: RANSAC

input: set of objects $\mathcal{X}$, parametric function f_μ.

1 Set b to the minimum number of elements necessary to uniquely characterize model μ, see Equation (5.1);
2 **foreach** $k \in \{1 \dots n\}$ **do**
3 Sample at random a minimal sample set (MSS) $\mathcal{X}_{\text{MMS}}$ of size b from $\mathcal{X}$;
4 Estimate $\theta^{(k)}$ from $\mathcal{X}_{\text{MMS}}$ by solving $(\forall \mathbf{x} \in \mathcal{X}_{\text{MMS}})\ f_\mu(\mathbf{x};\theta) = 0$;
5 Check that $\mu(\theta^{(k)})$ is not a degenerate model, otherwise go to line 3;
6 Compute $\mathcal{C}\left(\theta^{(k)}\right)$;
7 $k_{\max} = \text{argmax}_k \left|\mathcal{C}\left(\theta^{(k)}\right)\right|$;
8 Estimate $\theta^{(k_{\max})}$ from $\mathcal{C}\left(\theta^{(k_{\max})}\right)$ using least-squares;
9 **return** $\left(\mathcal{C}\left(\theta^{(k_{\max})}\right), \theta^{(k_{\max})}\right)$;

models generated with, e.g., RANSAC. First of all, the proposed modeling does not impose non-intersecting subsets $\mathcal{X}_i$. Secondly, it exploits consistencies that naturally arise during the RANSAC execution, while explicitly avoiding spurious inconsistencies. This new formulation conceptually changes the way that the data produced by the popular RANSAC, or related model-candidate generation techniques, is analyzed.

5.1. Multiple model estimation by analyzing the preference matrix. The information generated throughout the RANSAC iterations, i.e., the link between objects and model hypotheses, can be represented with the same $m \times n$ preference matrix $\mathbf{A}$ considered before, whose rows and columns represent objects and models, respectively; the element $(\mathbf{A})_{ij} = 1$ if the i-th object is in the CS of the j-th model, and 0 otherwise.

Traditionally and as in clustering, in algorithms like RANSAC, the preference matrix is (often implicitly) analyzed column-by-column or row-by-row. For example, Toldo and Fusiello [48] proposed to cluster the objects in $\mathcal{X}$ using the rows of $\mathbf{A}$ as feature vectors, obtaining a powerful state-of-the-art clustering-based technique for multiple model estimation, called J-linkage. For this, they use a tailored agglomerative clustering algorithm. As all agglomerative clustering algorithms, J-linkage proceeds in a bottom-up manner: starting from all singletons, each iteration of the algorithm merges the two clusters with the smallest distance. J-linkage uses the Jaccard distance [48], and the features are updated during the merging process. Each cluster's feature is computed as the intersection of the features of its objects (i.e., the logical conjunction of the corresponding rows of $\mathbf{A}$).

Although a clustering of objects, using as features the set of sampled models they belong to, might lead to good results in certain applications, it does not fully address the problem at hand. The relationship of the objects with the sampled models is the actual focus of interest (notice again the block-diagonal structure of $\mathbf{A}$ in Figure 1). A pattern-discovery algorithm is needed in this relationship space. We are thus interested in finding clusters in the product set $\mathcal{X} \times \mathcal{M}$, where $\mathcal{M} = \{\mathcal{C}(\theta^{(i)})\}_{1 \le i \le n}$ is the set of sampled models. As already mentioned in Section 2, such a problem is known in the literature as bi-clustering.

We thus propose to address the problem of model estimation by bi-clustering the matrix $\mathbf{A}$, using the algorithm presented in Section 2. A further conceptual advantage is that an object is allowed to belong to multiple bi-clusters, because we

are partitioning the product set $\mathcal{X} \times \mathcal{M}$ instead of $\mathcal{X}$. This kind of situation occurs very frequently, as observed in Figure 1: if the lines (models) intersect (left), then they share points (objects); this is translated as elements outside the block-diagonal structure of $\mathbf{A}$ (right). RANSAC-related techniques arbitrarily assign shared objects to a given model.

5.2. Cleansing the preference matrix. The standard random sampling approach to multiple model estimation generates many good model instances (composed of inliers), but also generates many bad models (composed mostly of outliers). In general, the number of bad models exceeds by far the number of good ones. It is not worth devoting computational effort in the analysis of these columns of $\mathbf{A}$. Any pattern-discovery technique, such as the bi-clustering approach presented in the previous section, would benefit from having a simple, efficient, and statistically meaningful method for discarding the bad models. These models will typically contain only a handful of objects. The question is how do we determine the minimum size of a good consensus set? This important computational contribution is addressed next, following the a contrario testing mechanism presented in depth in [13].

Let us assume that we have a set of m (random) objects. We are interested in computing the probability that a model $\mu(\theta)$ has an associated consensus set $\mathcal{C}(\theta)$ of at least $k = |\mathcal{C}(\theta)|$ objects. Under the simplifying assumption that all objects are i.i.d., the probability of such an event is $\mathcal{B}(m, k; p)$, where $\mathcal{B}$ is the binomial tail and $p = p(\mu(\theta), \delta)$ is the probability that a random object belongs to the consensus set $\mathcal{C}(\theta)$, built with noise level δ. We will later provide details on how to compute p for several different scenarios (Appendix B).

Let then $\mu(\theta)$ be a model and $\mathcal{C}(\theta)$ its associated consensus set, obtained with precision parameter δ. The model $\mu(\theta)$ is said to be ε-meaningful if (NFA stands for Number of False Alarms)

$$\text{NFA}(\mu(\theta)) = N_{\text{tests}}\, \mathcal{B}(m, |\mathcal{C}(\theta)|; p) < \varepsilon, \tag{5.5}$$

where as mentioned above $p = p(\mu(\theta), \delta)$ takes different forms for different models, and $N_{\text{tests}} = \binom{m}{b}$ represents the total number of considered models. Recall that b is the minimum number of elements necessary to uniquely characterize a given parametric model. It is easy to prove, by the linearity of expectation, that the expected number of ε-meaningful models in a finite set of random models is smaller than ε. Alternatively, N_{tests} can be empirically set by analyzing a training dataset [8], providing a tighter bound for the expectation.

Equation (5.5) provides a formal probabilistic method for testing if a model is likely to happen at random or not. From a statistical viewpoint, the method goes back to multiple hypothesis testing. Following an a contrario reasoning [13], we decide whether the event of interest has occurred if it has a very low probability of occurring by chance in the above defined random (background) model. In other words, a model $\mu(\theta)$ is ε-meaningful if $|\mathcal{C}(\theta)|$ is sufficiently large to have $\text{NFA}(\mu(\theta)) < \varepsilon$. Only ε-meaningful models are kept in $\mathbf{A}$.

Notice that we are in a sense using the a contrario validation procedure backwards: instead of using it to detect good models, we use it to eliminate bad models. We do not need a very sharp event-detection procedure in order to only keep good models; we only need a statistical test to eliminate the vast majority of clearly poor models. Hence, the value of ε is not critical, our model being inherently robust to poor models, as shown in Section 2. As any statistical test that controls false positives, our a contrario tests do not provide a good control of false negatives (i.e., missed detections).

We can thus easily relax the value of $\varepsilon = 1$, allowing for some quantity of poor models in our cleansed preference matrix, but being sure that we do not miss good ones.

As a result of this statistical validation procedure, the preference matrix $\mathbf{A}$ is considerably shrunk. Many unuseful columns are eliminated (be observed that only about 10% of the original columns are kept in our experiments). Due to this elimination, some rows might also become zero-valued and can also be eliminated. This shrunk preference matrix is fed to the bi-clustering algorithm (Algorithm 1), gaining in stability of the results as well as in speed.

5.3. An algorithm for multiple parametric model estimation. When we presented Algorithm 1, we claimed that tighter bounds could be computed for τ_R in the case of parametric models. The tests used for cleansing the preference matrix are readily available for this task. For each bi-cluster, we can compute its NFA, where the number of elements in the group if the number of rows in the bi-cluster. Finally, we only keep those bi-clusters that are ε-meaningful. The resulting complete algorithm is as follows:

1. Compute the preference matrix $\mathbf{A}$ by randomly sampling minimal sampled sets and computing their corresponding consensus sets (Section 5.1).
2. Cleanse $\mathbf{A}$ by discarding the columns that do not satisfy the statistical test presented in Section 5.2. For these tests, we relax the value of ε.
3. Bi-cluster the cleansed version of $\mathbf{A}$ using Algorithm 1.
4. Discard the bi-clusters that are not ε-meaningful, see Section 5.2. For these tests, we fix $\varepsilon = 1$. We also ask that each bi-cluster is meaningful when we only consider the elements that do not belong to other bi-clusters. This last step ensures that each bi-cluster contains some points that only belong to it and is a mild version of the exclusion principle [13].

We next present many experimental results and different applications that make use of this general algorithm for multiple model estimation.

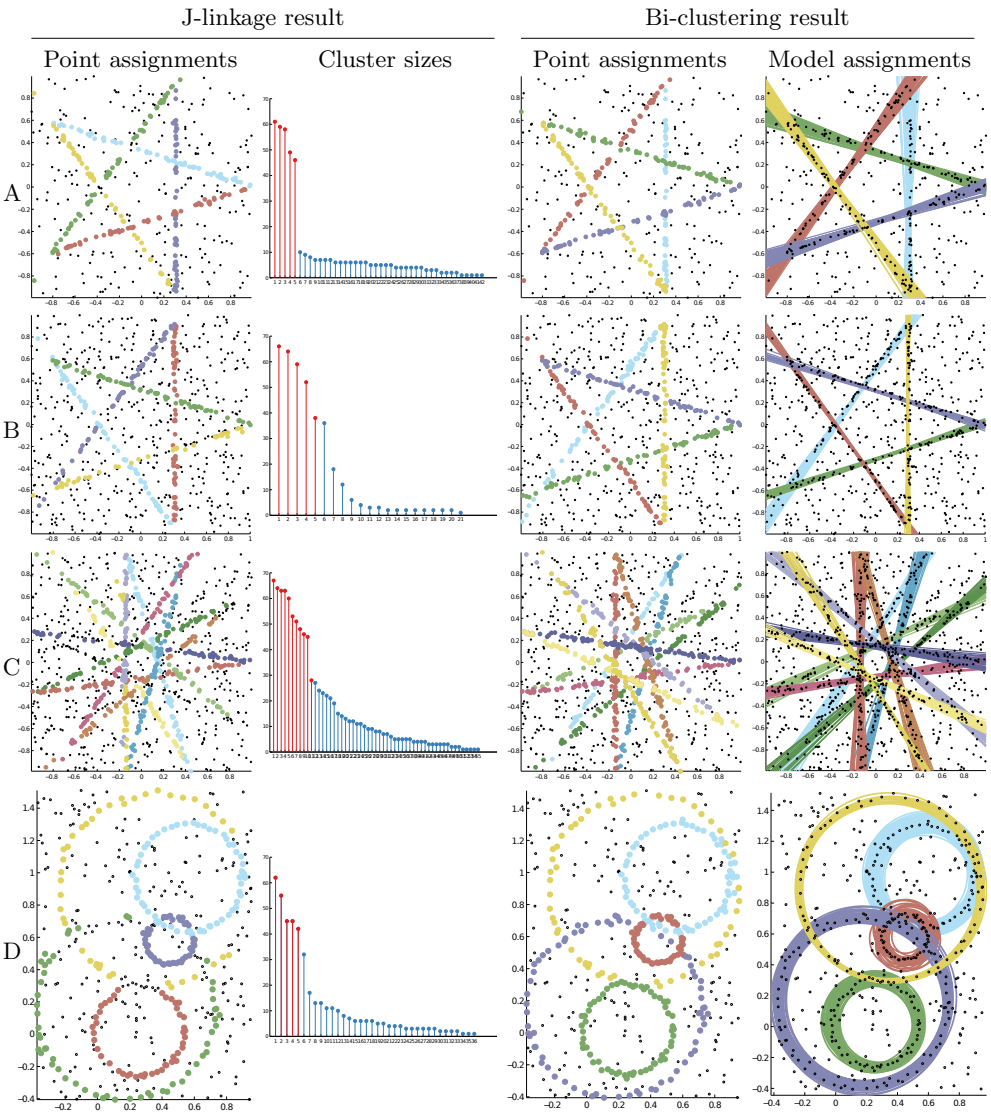
5.4. Experimental results. Each minimal sample set (MSS) is built using non-uniform random sampling [58], such that if an object $\mathbf{x}$ has already been chosen, $\mathbf{y} \neq \mathbf{x}$ is selected with probability

$$\Pr(\mathbf{y}|\mathbf{x}) = z^{-1} \exp\left(-\sigma^{-2} \operatorname{dist}(\mathbf{x}, \mathbf{y})^2\right), \tag{5.6}$$

where z is a normalization constant and σ is a parameter of the algorithm. Depending on the application at hand, the distance dist takes different forms (specified in Appendix B).

We provide several standard 2D examples [48] with lines and circles in Figure 12. For the line examples, we set $n = 5000$, $\sigma = 0.5$, $\delta = 0.03$. For circles, we set $n = 20000$, $\sigma = 0.5$, $\delta = 0.03$. In all examples, the proposed bi-clustering approach does a much better job at recovering the data structure than J-linkage and other popular techniques, see Figure 13. J-linkage, considered a state-of-the-art algorithm, has a general tendency to find clusters with fewer points than expected (there are many 'undercomplete' lines and circles). This is further emphasized by comparing the recall of both methods, see Table 12(b).

In all these examples, the bi-clustering algorithm automatically finds the number of clusters. J-linkage uses the size of the obtained clusters to decide whether to keep them or to discard them. In Figure 12(a), it can be easily seen that this method is not stable, since the decay in the size does not indicate the proper cut-point.



(a) J-linkage does not perform accurate point-model assignments, notice the missing points on the detected lines and circles. The size of the J-linkage clusters is not a robust criterion for selecting the final clusters. The proposed approach correctly retrieves the lines and circles.

Dataset	J-linkage	Bi-clustering
A	96	100.0
B	87.2	100.0
C	82.2	99.6
D	81.6	99.6

(b) Comparison of the recall (%) for the above examples (in these settings precision is distorted by the outliers and is not completely meaningful). This shows that the bi-cluster sizes are very close to the ground truth sizes (slightly larger, again because of the outliers).

Fig. 12: Several synthetic examples where the proposed approach yields considerable improvements over J-linkage [48], both on the quality and the stability of the results.

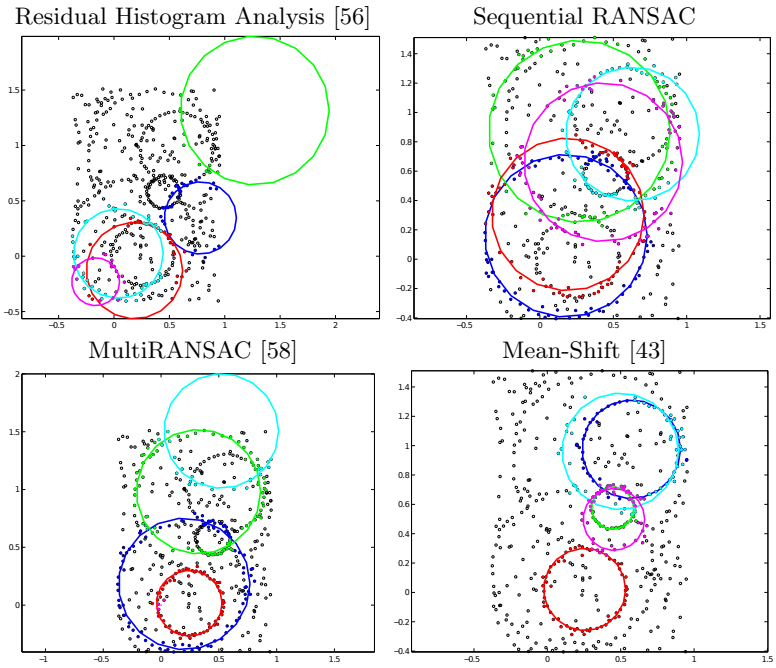


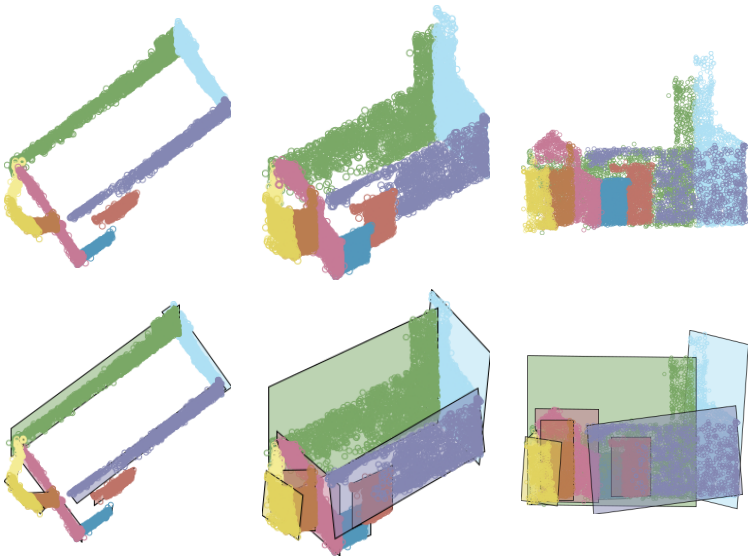
Fig. 13: Comparisons with several multiple model estimation algorithms on the example in Figure 12 (reproduced from [48]). Contrarily to the proposed approach, these techniques both miss models and detect false ones.

We also observe in Figure 12 that the proposed approach can correctly recover overlapping models. This is an intrinsic limitation of J-linkage and most multiple model estimation techniques, which are generally based on partitioning (clustering) the set of objects (data).

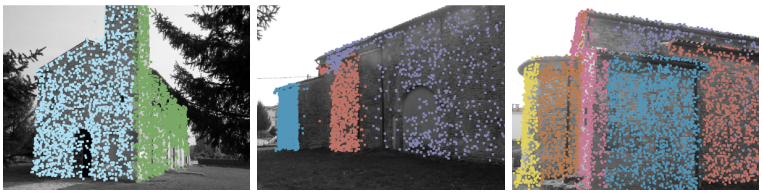
Figure 14 presents an example of a real application with 3D planes from the Pozzovegianni dataset.[2] The 3D points are obtained from different images of a building with a sparse multi-view 3D reconstruction algorithm. Our algorithm recovers the 3D planes in the scene to properly reconstruct the building structure (for this example, $n = 5000$, $\sigma = 0.5$, $\delta = 0.5$).

As an additional example, we developed a simple method for detecting cells in microscopy images, see Figure 15. This is achieved by using ellipses as models (during sampling, we discard ellipses that are too elongated). Instead of using edge points as the base elements, we use line segments, detected with LSD [19], providing more robust detections and making the process of estimating an ellipse from an MSS more stable. For this example, we set $n = 20000$, $\sigma = 30$, $\delta = 10$ and instead of using a zero-centered distance distribution, see Equation (5.6), we set the mean distance to 80 pixels. Our method retrieves most of the ellipses (i.e., cell membranes) in the image without further tuning our generic algorithm. Notice that we employ a generic a contrario test, that is not completely adapted to this scenario: should the lengths of the segments be taken into account in the NFA, see definition (5.5), the results would

[2] `http://www.diegm.uniud.it/fusiello/demo/samantha/#pozzo`



(a) Three 3D views of the 9 bi-clusters obtained with the proposed method. On the bottom row, we also display the fitted planes



(b) Three different views of the 3D points projected back to the original images.

Fig. 14: Example with 3D planes on the Pozzoveggiani dataset. The 3D points are obtained from a set of images, such as the ones in (b), using multi-view 3D reconstruction. We correctly recover the building structure by detecting 3D planes.

automatically be further refined and improved. This is noticeable for small ellipses (i.e., cell nuclei) that contain very few segments. Our goal in this work is to present the general detection framework, we leave this specific refinement for future work.

Finally, we present another application for the proposed framework: vanishing point detection in uncalibrated images. As with cells, in this application we use line segments as the base objects/elements of our method. The groups are formed by detecting the subset of lines that intersect at a given point in the image plane [47]. In this case, since we only need 2 segments to compute a MSS (i.e., an intersection point) [47], and the number m of segments in not too large, we compute all $m(m-1)/2$ MSSs, instead of sampling at random ($\sigma = 1$, $\delta = 10$ pixels). This helps to show that in the other cases the sampling procedure does not artificially boosts the performance, and only helps to speed up the algorithm. In Figure 16 we show some results from the York Urban database [12], where we can robustly and reliably detect vanishing points. As with ellipses, the results could be improved by considering the segment

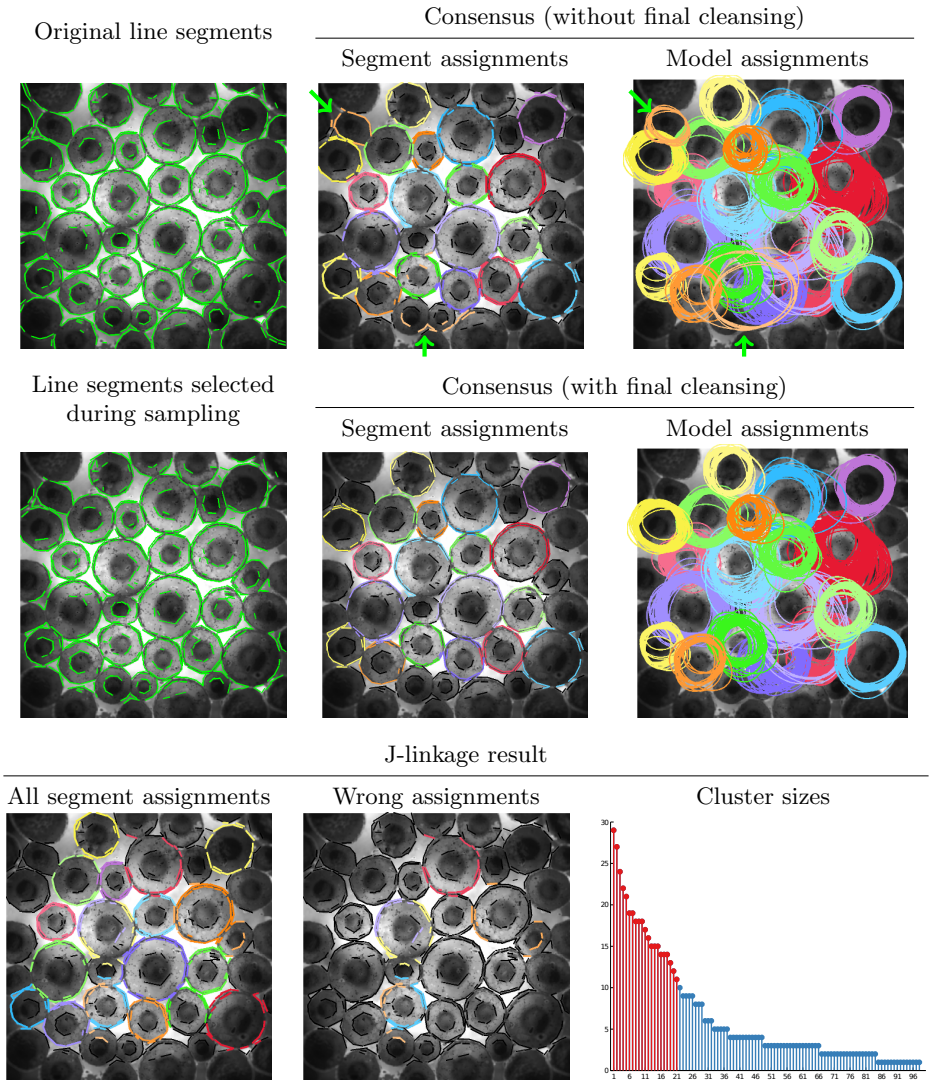


Fig. 15: Ellipse (cell) detection example. We use line segments [19] as base elements for our method. We are able to reliably detect most ellipses without a specific and highly tuned method. Notice that the final cleansing process eliminates two detected ellipses, one in the top left corner and one on the bottom of the image (indicated with green arrows in the first row). These removals make sense from a perceptual point of view. J-linkage (bottom row) does not yield good results, returning wrong clusters and exhibiting an arbitrary cut-off point (notice the smooth decay in the cluster sizes).

length in the a contrario test. Using more sophisticated and specific schemes to obtain the vanishing point candidates might also lead to further improvements [31].

In these experiments we can observe that the bi-clustering approach provides: (1) a good description for each detected model instance in terms of its objects; (2) a compact overall description, by considering the reduced number of detected bi-

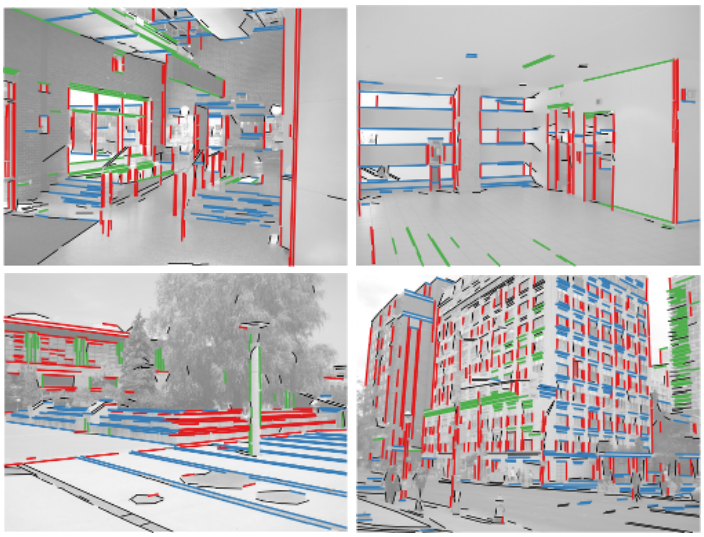


Fig. 16: Vanishing point detection example. Our simple approach correctly recovers the vanishing points in the images. We only show the line segments in each bi-cluster, because the vanishing points might lie far away from the image in the image plane. The detection of 3D parallelism from 2D images is an ill-posed problem, as some information is inherently lost during the projection. For a handful of segments (e.g., the one marked with a green arrow in the second image), there is no way of determining from the observed projections their true 3D orientation. Despite this indeterminacy, the overall problem is still solvable.

clusters; and (3) support for overlapping groups of objects. These features of the proposed general approach lead to a rich characterization of the data in terms of parametric models.

5.5. The boundaries of multiple parametric model estimation. The proposed approach for detecting parametric models assumes that observing an unusual concentration of elements around a parametric model instance is enough to produce a robust candidate. In most situations this assumption is valid and works well in practice, as seen in the numerous previous examples. However, there are situations where these element-model distances are not enough to fully characterize a given configuration of elements. Let us illustrate this with a simple example, see Figure 17. Using the classical definition of a consensus set, see (5.3), every line that passes through the central cluster will have a large consensus set. This artificially fires many candidate detections, that end up creating a bi-cluster. Notice that (1) this result is completely consistent with our theoretical formulation, and (2) this is not a bi-product of using lines since a similar effect will occur if we use circles, for example. Neither RANSAC nor the Hough transform can discern between points lying along a line and points concentrated in a small cluster. In fact, the setup is missing a dispersion constraint along the model. This has been addressed for a particular case in [31], but many formulations could be employed to achieve the desired effect, such as Ripley's K and L functions [41]. Additionally, this improved validation would greatly simplify the validation step of the proposed method.

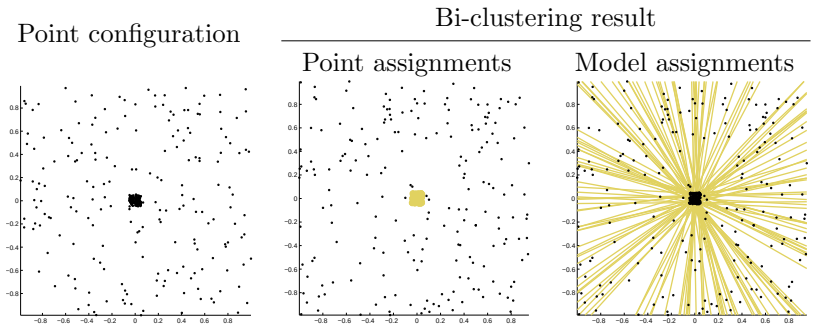


Fig. 17: An aberrant example, containing uniformly distributed points and a small cluster in the center. We run our line detection algorithm and recover the central cluster. The result is consistent with the proposed protocol, although one might claim that a line detection scheme should yield no detections in this case. A stricter definition of the a contrario pruning rule is needed to correct this effect: as before, we need a concentration of points in the orientation orthogonal to the line and *additionally* a dispersion of points along the line.

The values of δ (the distance threshold) and n (number of considered MSSs) are in general very important to obtain good results. Notice that this dependency is not introduced by our bi-clustering formulation, but common to most parametric model estimation approaches. For example, methods based on RANSAC and the randomized Hough transform share this dependency. In the general case, there is no theoretical formulation nor clear practical guideline for properly setting their values. In practice, many cases present a reasonably intuitive range for δ. In our experience, setting the value of n is not critical when time is not a concern, as we can simply select a large value. In time-critical applications, we can learn n from training examples.

Of course, the choices of n and τ_C are related. For a single model, the probability of drawing at least τ_C outlier-free MSSs out of n (and thus guaranteeing its detection) is given by

$$1 - \sum_{k=0}^{\tau_C - 1} \binom{n}{\tau_C} (p_{\text{MSS}})^k (1 - p_{\text{MSS}})^{n-k}, \tag{5.7}$$

where p_{MSS} is the probability of drawing an MSS of cardinality b composed only of inliers (recall that b is the minimum number of elements necessary to uniquely characterize a given parametric model). See [48] for further details. This formulation has two shortcomings. First, p_{MSS} is not a trivial quantity to set and/or estimate without deep knowledge about the structure of the dataset (consider that we are actually trying to discover this structure). Second, the above equation relates n and τ_C for a single model; establishing a sound relation for multiple models requires knowing in advance the number of models. More research is much needed in this theoretically and practically important aspect, as all state-of-the-art approaches would benefit from it. As a side note, this problem is similar in nature to the Chinese restaurant and Indian buffet processes, but with a fixed and unknown number of tables/dishes [2,18].

6. Connection with the computational Gestalt theory. We now provide a brief discussion on the connections between the presented framework and the com-

putational Gestalt theory.

The driving principle in the computational Gestalt theory is the Helmholtz principle [13]. In its simplest form, it states that we do not perceive any structure in a uniform random image [3]. The computational Gestalt theory makes extensive use of a stronger form, namely that whenever some large deviation from randomness occurs, a structure is perceived.

The Helmholtz principle is generically illustrated as follows. Let $\mathcal{X}$ be the set of atomic objects present in an image. Let us assume that there is a subset $\mathcal{X}_{\text{CF}} \subset \mathcal{X}$ whose objects share a common feature, say, color, orientation, position, etc. We then face a decision problem: is this common feature happening by chance or is it significant enough to make $\mathcal{X}_{\text{CF}}$ stand out as a perceptual group? To answer this question, let us make the following mental experiment: assume a priori that the considered feature is randomly and uniformly distributed on all objects of $\mathcal{X}$, i.e., the observed feature is a realization of this uniform process. We finally ask: is this realization probable or not? If not, this proves a contrario that a grouping process (a gestalt) is at play. The Helmholtz principle states roughly that in such mental experiments, the object features are assumed to be uniformly distributed and independent [13].

The theory can be interpreted as a multiple hypothesis testing framework for visual events and works by controlling the Number of False Alarms (NFA), a proxy of the expectation of the number of occurrences of a visual event under the stated a contrario model.

In Section 5 we exploited this detection theory twice: (1) to cleanse the preference matrix, and (2) to adjust τ_R, the minimum number of elements (rows) necessary to accept a bi-cluster. These a contrario tests allow us to assess the improbability of a given event or configuration. However, our framework also computes information regarding the repeatability of this configuration when probing the data. In this sense, we can interpret our framework as repeatedly querying the data at random and looking for configurations that arise over and over. The a contrario tests do not exploit this additional dimension that nonetheless plays a central role when assessing randomness: if a given configuration arises over and over, what are the chances that it is a realization of a random process? This simple discussion brings forward the need to develop statistical tests that also consider the repeatability of a configuration under random sampling, via τ_c in our framework.

As a side note, we would also like to point out the fact that, when $(\forall\theta)\ p(\mu(\theta), \delta) = p_\delta$, as in the case of 2D lines (see Appendix B), $p_\delta \approx \|\mathbf{A}\|_0 / (mn)$, where $\|\bullet\|_0$ stands for the pseudo-norm counting the number of non-zeros. This simple observation might lead to develop simpler, more general, and more extensible geometric probes, by employing a reduced set of assumptions for computing $p(m(\theta), \delta)$ than the ones used in Appendix B, common in all a contrario literature.

The presented framework also has connections with the so-called Prägnanz principle in Gestalt psychology: "...of several geometrically possible organizations that one will actually occur which possesses the best, simplest and most stable shape," quoted in [57] from Koffka's book [26]. Analyzing the set of possible configurations, the proposed framework assigns a formal and mathematical meaning to the terms *best, simplest, and most stable*. Let us analyze these characteristics one by one:

Best. Our resulting configurations (bi-clusters) are obtained as the solution of a non-parametric minimization problem with an intuitive interpretation, see problem 2.4.

Simplest. The rank-one nature of the extracted bi-clusters means that they provide

a minimalistic explanation for the set of configurations. This set is interpreted as a summation of rank-one bi-clusters, plus some noise.

More stable. Configurations that are hard to reproduce exhibit low stability. We already discussed how our framework extracts the configurations that show more repeatability when probing the data.

From this point of view, our framework might provide a mathematically sound way to formulate and implement the Prägnanz principle, while integrating it with the Helmholtz principle, seen as a stopping criterion for the bi-clustering process. Psychophysical experiments are of course needed to fully validate this intuitive conceptual vinculation.

7. Conclusion. In this paper we proposed a framework and a new perspective for reaching consensus in grouping problems. Our general characterization of grouping problems subsumes many different areas, e.g., clustering, community detection in networks, and multiple parametric model estimation. We pose consensus grouping as a bi-clustering problem, obtaining a conceptually simple and descriptively rich modeling. We presented a simple but powerful bi-clustering algorithm, specifically tuned to the nature of the problem we address, though general enough to handle many different instances inscribed within our framework.

In particular, this is the first time that the task of finding/fitting multiple parametric models to a dataset was formally posed as a consensus bi-clustering problem. The equivalence of these tasks is highlighted by the proposed framework, and we devoted special attention to explain the rationale behind this new characterization. As a future line of research, we are currently investigating whether using a hard thresholding scheme is actually necessary. Instead of working with binary data, we could work with a real-valued object-model distance matrix, eliminating a critical parameter that has been haunting the RANSAC framework for years.

We also discussed the connection with the computational Gestalt program [13], that seeks to provide a quantitative psychologically-inspired detection theory for visual events. We provided some cues that show the suitability of our approach as a new research direction in this field. We are working on exploiting these new connections to fully develop a new perspective on this fundamental problem.

We are also exploring how to use a contrario statistical tests in non-parametric scenarios. From this perspective, we point out again that we do not need a very sharp testing mechanism but a coarse procedure to avoid filling the preference matrix with a huge number of poor groups that could clutter the bi-clustering algorithm. This would allow for more freedom in the selection of the pool of input algorithms.

As an alternative, the framework could be extended with individual weights w_{ij} (instead of column-wide weights) in the preference matrix. These weights would thus model a confidence measure of the i-th element belonging to the j-th group. It would be interesting to explore this possibility in depth.

Finally, we would like to stress that the proposed framework is not limited to the presented applications. It is flexible enough to handle any type of grouping problem and we plan to address other applications that can be formulated in this way. Two clear examples of this are image segmentation (seen as an extension of clustering with spatial constraints) and supervised classification, where we aim at fusing the output of different classifiers to obtain robustified results.

Appendix A. We now show how to solve problem (2.4). The problem can be

equivalently re-formulated as

$$\min_{\mathbf{U},\mathbf{V},\mathbf{X},\mathbf{Y},\mathbf{E}} \|\mathbf{E}\|_1 \quad \text{s.t.} \quad \begin{array}{c} \mathbf{E} = \mathbf{A} - \mathbf{X}\mathbf{Y} \\ \mathbf{X} = \mathbf{U}, \mathbf{Y} = \mathbf{V} \\ \mathbf{U}, \mathbf{V} \geq 0, \end{array} \tag{A.1}$$

where $\mathbf{E} \in \mathbb{R}^{m\times n}$, $\mathbf{X}, \mathbf{U} \in \mathbb{R}^{m\times q}$, and $\mathbf{Y}, \mathbf{V} \in \mathbb{R}^{q\times n}$. We consider the augmented Lagrangian of (A.1),

$$\begin{aligned} \mathscr{L}(\mathbf{X}, \mathbf{Y}, \mathbf{U}, \mathbf{V}, \mathbf{E}, \Lambda, \Phi, \Psi) = \|\mathbf{E}\|_1 + \Lambda \bullet (\mathbf{X} - \mathbf{U}) + \Phi \bullet (\mathbf{Y} - \mathbf{V}) + \\ + \Psi \bullet (\mathbf{A} - \mathbf{X}\mathbf{Y} - \mathbf{E}) + \tfrac{\alpha}{2} \|\mathbf{X} - \mathbf{U}\|_F^2 + \\ + \tfrac{\beta}{2} \|\mathbf{Y} - \mathbf{V}\|_F^2 + \tfrac{\gamma}{2} \|\mathbf{A} - \mathbf{X}\mathbf{Y} - \mathbf{E}\|_F^2 , \end{aligned} \tag{A.2}$$

where $\Lambda \in \mathbb{R}^{m\times q}, \Phi \in \mathbb{R}^{q\times n}, \Psi \in \mathbb{R}^{m\times n}$ are Lagrange multipliers, α, β, γ are penalty parameters, and $\mathbf{B} \bullet \mathbf{C} = \sum_{i,j} (\mathbf{B})_{ij} (\mathbf{C})_{ij}$ for matrices $\mathbf{B}, \mathbf{C}$ of the same size.

We use the Alternating Direction Method of Multipliers (ADMM) for solving (A.1). The algorithm works in a coordinate descent fashion, successively minimizing $\mathscr{L}$ with respect to $\mathbf{X}, \mathbf{Y}, \mathbf{U}, \mathbf{V}, \mathbf{E}$, one at a time while fixing the others at their most recent values, i.e.,

$$\mathbf{X}_{k+1} = \operatorname*{argmin}_{\mathbf{X}} \mathscr{L}(\mathbf{X}, \mathbf{Y}_k, \mathbf{U}_k, \mathbf{V}_k, \mathbf{E}_k, \Lambda_k, \Phi_k, \Psi_k), \tag{A.3a}$$

$$\mathbf{Y}_{k+1} = \operatorname*{argmin}_{\mathbf{Y}} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}, \mathbf{U}_k, \mathbf{V}_k, \mathbf{E}_k, \Lambda_k, \Phi_k, \Psi_k), \tag{A.3b}$$

$$\mathbf{U}_{k+1} = \operatorname*{argmin}_{\mathbf{U}\geq 0} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}, \mathbf{U}, \mathbf{V}_k, \mathbf{E}_k, \Lambda_k, \Phi_k, \Psi_k), \tag{A.3c}$$

$$\mathbf{V}_{k+1} = \operatorname*{argmin}_{\mathbf{V}\geq 0} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}, \mathbf{U}_{k+1}, \mathbf{V}, \mathbf{E}_k, \Lambda_k, \Phi_k, \Psi_k), \tag{A.3d}$$

$$\mathbf{E}_{k+1} = \operatorname*{argmin}_{\mathbf{E}} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}, \mathbf{U}_{k+1}, \mathbf{V}_{k+1}, \mathbf{E}, \Lambda_k, \Phi_k, \Psi_k), \tag{A.3e}$$

and then updating the multipliers Λ, Φ, Ψ, i.e.,

$$\Lambda_{k+1} = \operatorname*{argmin}_{\Lambda} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}, \mathbf{U}_{k+1}, \mathbf{V}_{k+1}, \mathbf{E}_{k+1}, \Lambda, \Phi_{k+1}, \Psi_{k+1}), \tag{A.3f}$$

$$\Phi_{k+1} = \operatorname*{argmin}_{\Phi} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}, \mathbf{U}_{k+1}, \mathbf{V}_{k+1}, \mathbf{E}_{k+1}, \Lambda_{k+1}, \Phi, \Psi_{k+1}), \tag{A.3g}$$

$$\Psi_{k+1} = \operatorname*{argmin}_{\Psi} \mathscr{L}(\mathbf{X}_{k+1}, \mathbf{Y}_{k+1}, \mathbf{U}_{k+1}, \mathbf{V}_{k+1}, \mathbf{E}_{k+1}, \Lambda_{k+1}, \Phi_{k+1}, \Psi). \tag{A.3h}$$

Each of these steps can be written in closed form as

$$\mathbf{X}_{k+1} = \left(\gamma (\mathbf{A} - \mathbf{E}_k) \mathbf{Y}_k^{\mathrm{T}} + \alpha \mathbf{U}_k - \Lambda_k + \Psi_k \mathbf{Y}_k^{\mathrm{T}}\right) \left(\mathbf{Y}_k \mathbf{Y}_k^{\mathrm{T}} + \alpha \mathbf{I}\right)^{-1}, \tag{A.4a}$$

$$\mathbf{Y}_{k+1} = \left(\mathbf{X}_{k+1}^{\mathrm{T}} \mathbf{X}_{k+1} + \beta \mathbf{I}\right)^{-1} \left(\gamma \mathbf{X}_{k+1}^{\mathrm{T}} (\mathbf{A} - \mathbf{E}_k) + \beta \mathbf{V}_k - \Phi_k + \mathbf{X}_{k+1}^{\mathrm{T}} \Psi_k\right), \tag{A.4b}$$

$$\mathbf{U}_{k+1} = \mathscr{P}_+ \left(\mathbf{X}_{k+1} + \alpha^{-1} \Lambda_k\right), \tag{A.4c}$$

$$\mathbf{V}_{k+1} = \mathscr{P}_+ \left(\mathbf{Y}_{k+1} + \beta^{-1} \Phi_k\right), \tag{A.4d}$$

$$\mathbf{E}_{k+1} = \operatorname{shrink}\left(\mathbf{A} - \mathbf{X}_{k+1} \mathbf{Y}_{k+1}^{\mathrm{T}} + \gamma^{-1} \Psi_k \ , \ \gamma^{-1}\right), \tag{A.4e}$$

$$\Lambda_{k+1} = \Lambda_k + \xi\alpha \left(\mathbf{X}_{k+1} - \mathbf{U}_{k+1}\right), \tag{A.4f}$$

$$\Phi_{k+1} = \Phi_k + \xi\beta \left(\mathbf{Y}_{k+1} - \mathbf{V}_{k+1}\right), \tag{A.4g}$$

$$\Psi_{k+1} = \Psi_k + \xi\gamma \left(\mathbf{A} - \mathbf{X}_{k+1} \mathbf{Y}_{k+1}^{\mathrm{T}} - \mathbf{E}_{k+1}\right), \tag{A.4h}$$

where $\mathbf{I}$ is the $q \times q$ identity matrix, and

$$(\mathscr{P}_+(\mathbf{B}))_{ij} = \max\{(\mathbf{B})_{ij}, 0\}, \tag{A.5a}$$

$$(\text{shrink}(\mathbf{B}, \lambda))_{ij} = \text{sign}((\mathbf{B})_{ij}) \max\{|(\mathbf{B})_{ij}| - \lambda, 0\}. \tag{A.5b}$$

These iterations define our algorithm, the initialization being done with a rank-q SVD. In practice, we set $\alpha, \beta, \gamma, \xi$ to 1.

Appendix B. We now describe how to compute $p(\mu(\theta), \delta)$ for the examples presented in Section 5. For simplicity, we assume that the objects under the background model are independent and identically distributed, following a uniform law. The presented tests are, in some cases, rather crude. Much tighter bounds can be found by carefully tuning the probabilistic models for each specific application. However, in this simpler forms, they are already useful for demonstrating the capabilities of the proposed framework, and are sufficient to lead to state-of-the-art results.

2D Lines. An object in $\mathcal{X}$ is a 2D point $\mathbf{x} \in \mathbb{R}^2$. We need $b = 2$ points to define a line. The parameter vector $\theta \in \mathbb{R}^3$ of a line passing through two points $\mathbf{x}$, $\mathbf{x}'$, is given by $\theta = \left[\begin{smallmatrix}\mathbf{x}\\1\end{smallmatrix}\right] \times \left[\begin{smallmatrix}\mathbf{x}'\\1\end{smallmatrix}\right]$. Then, a line is the set of points (see Equation (5.1)) such that

$$\mu(\theta) = \{\mathbf{x} \in \mathbb{R}^2,\ \begin{bmatrix}\mathbf{x}^T & 1\end{bmatrix} \theta = 0\}, \tag{B.1}$$

and the distance between a point $\mathbf{x}$ and a line (see Equation (5.2)) with parameter vector $\theta = [A, B, C]^{\mathrm{T}}$ can be written as

$$\mathrm{e}_\mu(\mathbf{x}, \theta) = \left(A^2 + B^2\right)^{-1/2} \begin{bmatrix}\mathbf{x}^T & 1\end{bmatrix} \theta. \tag{B.2}$$

Let a be the area of the bounding box enclosing the points (data). The longest line segment in the bounding box has length D, where D is the diagonal length of the bounding box. For a given line (model) ℓ, we accept points $\mathbf{q}$ such that $\mathrm{e}_\mu(\mathbf{q}, \theta) < \delta$. We can then compute the probability of a random point lying on a band with length D and width 2δ [13]. This is given by $2\delta D/a$, and we set $p(\mu(\theta), \delta)$ to this value for all θ.

2D Circles. An object in $\mathcal{X}$ is a 2D point $\mathbf{p} \in \mathbb{R}^2$. We need $b = 3$ points to define a circle. The parameter vector $\theta = \left[\begin{smallmatrix}\mathbf{c}\\\rho\end{smallmatrix}\right]$ of a circle, with center $\mathbf{c} \in \mathbb{R}^2$ and radius $\rho \in \mathbb{R}^+$, passing through three points $\mathbf{p}$, $\mathbf{p}'$, $\mathbf{p}''$ can be found by solving the system of equations $\|\mathbf{p} - \mathbf{c}\|_2^2 = \|\mathbf{p}' - \mathbf{c}\|_2^2 = \|\mathbf{p}'' - \mathbf{c}\|_2^2 = \rho^2$. A circle is the set of points (see Equation (5.1)) such that

$$\mu(\left[\begin{smallmatrix}\mathbf{c}\\\rho\end{smallmatrix}\right]) = \{\mathbf{x} \in \mathbb{R}^2,\ \|\mathbf{x} - \mathbf{c}\|_2 = \rho\}, \tag{B.3}$$

and the distance between a point $\mathbf{x}$ and a circle (see Equation (5.2)) can be written as

$$\mathrm{e}_\mu(\mathbf{x}, \left[\begin{smallmatrix}\mathbf{c}\\\rho\end{smallmatrix}\right]) = |\ \|\mathbf{x} - \mathbf{c}\|_2 - \rho\ |. \tag{B.4}$$

The probability of a random point lying on a band of width 2δ around a circle with radius ρ is given by $\pi\left[(\rho + \delta)^2 - (\rho - \delta)^2\right]/a$, where a is the area of the bounding box enclosing the data. We set $p(\mu(\theta), \delta)$ to this value.

3D Planes. An object in $\mathcal{X}$ is a 3D point $\mathbf{p} \in \mathbb{R}^3$. We need $b = 3$ points to define a plane. The parameter vector $\theta \in \mathbb{R}^4$ of a plane passing through three points

$\mathbf{p}$, $\mathbf{p}'$, $\mathbf{p}''$, can be found by solving the system of equations $\left[\begin{smallmatrix}\mathbf{p} & \mathbf{p}' & \mathbf{p}'' \\ 1 & 1 & 1\end{smallmatrix}\right]^{\mathrm{T}}\theta = 0$. A plane is the set of points (see Equation (5.1)) such that

$$\mu(\theta) = \{\mathbf{x} \in \mathbb{R}^3,\ \left[\mathbf{x}^T\ 1\right]\theta = 0\}, \tag{B.5}$$

and the distance between a point $\mathbf{x}$ and a plane (see Equation (5.2)) with parameter vector $\theta = [A, B, C, D]^{\mathrm{T}}$ can be written as

$$\mathrm{e}_\mu(\mathbf{x}, \theta) = \left(A^2 + B^2 + C^2\right)^{-1/2}\ \left[\mathbf{x}^T\ 1\right]\theta. \tag{B.6}$$

Let r be half the diagonal length of the 3D bounding box enclosing the points (data). We compute the probability of a random point lying on a band of width 2δ around a plane. This can be approximated by $2\pi r^2\delta/(\frac{4}{3}\pi r^3)$, and we set $p(\mu(\theta), \delta)$ to this value for all θ.

Ellipses in images. An object in the image is a line segment, detected using LSD [19]. We use $b = 3$ segments to define an ellipse. We can define an ellipse, with parameter vector $\theta = [A, B, C, D, E, F]^{\mathrm{T}}$, as the set of points (see Equation (5.1)) such that

$$\mu\left([A, B, C, D, E, F]^{\mathrm{T}}\right) = \{[x, y]^{\mathrm{T}} \in \mathbb{R}^2,\ Ax^2 + Bxy + Cy^2 + Dx + Ey + F = 0\}, \tag{B.7}$$

where $B^2 - 4AC < 0$. An ellipse passing through three line segments can be found by solving the system of equations detailed in [20]. We can compute the distance between a segment, with endpoints $\mathbf{x}, \mathbf{x}'$, and an ellipse (see Equation (5.2)) as

$$\mathrm{e}_\mu\left(\{\mathbf{x}, \mathbf{x}'\}, [A, B, C, D, E, F]^{\mathrm{T}}\right) = \max_{\mathbf{y} \in \mu([A,B,C,D,E,F]^{\mathrm{T}})} \left\{ \begin{array}{l} \|\mathbf{x} - \mathbf{y}\|_2 \\ \|\mathbf{x}' - \mathbf{y}\|_2 \end{array} \right\}. \tag{B.8}$$

Solving this for each endpoint involves finding the roots of a quartic polynomial. The probability of a random point lying on a band of width 2δ around an ellipse with radii ρ, ρ', is given by $\pi\left[(\rho + \delta)(\rho' + \delta) - (\rho - \delta)(\rho' - \delta)\right]/a$, where a is the area of the bounding box enclosing the data. We set $p(\mu(\theta), \delta)$ to this value.

Vanishing points. We refer the reader to [47] for a description of the a contrario test used in this case.

REFERENCES

[1] R. Aldecoa and I. Marín, *Jerarca: Efficient analysis of complex networks using hierarchical clustering*, PLoS ONE, 5 (2010), pp. e11585+.

[2] D. J. Aldous, *Exchangeability and related topics*, in École d'Été de Probabilités de Saint-Flour XIII 1983, vol. 1117 of Lecture Notes in Mathematics, 1985.

[3] F. Attneave, *Some informational aspects of visual perception.*, Psychol. Rev., 61 (1954), pp. 183–193.

[4] K. Bache and M. Lichman, *UCI Machine Learning Repository*, 2013.

[5] S. Ben-David and M. Ackerman, *Measures of clustering quality: A working set of axioms for clustering*, in NIPS, 2008.

[6] V. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, *Fast unfolding of communities in large networks*, J. Stat. Mech., 2008 (2008), pp. P10008+, 0803.0476.

[7] W. Brendel and S. Todorovic, *Segmentation as maximum-weight independent set*, in NIPS, 2010.

[8] N. Burrus, T. M. Bernard, and J. M. Jolion, *Image segmentation by a contrario simulation*, Pattern Recognition, 42 (2009), pp. 1520–1532.

[9] R. Campigotto, J. L. Guillaume, and M. Seifi, *The power of consensus: Random graphs have no communities*, in ASONAM, 2013.

[10] G. Carlsson and F. Mémoli, *Characterization, stability and convergence of hierarchical clustering methods*, J. Mach. Learn. Res., 11 (2010), pp. 1425–1470.
[11] S. Choi, T. Kim, and W. Yu, *Performance evaluation of RANSAC family*, in BMVC, 2009.
[12] P. Denis, J. H. Elder, and F. Estrada, *Efficient edge-based methods for estimating manhattan frames in urban imagery*, in ECCV, 2008.
[13] A. Desolneux, L. Moisan, and J.-M. Morel, *From Gestalt Theory to Image Analysis*, vol. 34, Springer-Verlag, 2008.
[14] E. Elhamifar and R. Vidal, *Sparse subspace clustering: Algorithm, theory, and applications*, IEEE Trans. Pattern Anal. Mach. Intell., 35 (2013), pp. 2765–2781.
[15] T. S. Evans, R. J. Rivers, and C. Knappett, *Interactions in space for archaeological models*, Adv. Complex Syst., 15 (2011), pp. 1150009+.
[16] M. Fischler and R. Bolles, *Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography*, Commun. ACM, 24 (1981), pp. 381–395.
[17] M. Girvan and M. E. J. Newman, *Community structure in social and biological networks*, Proc. Natl. Acad. Sci. U.S.A., 99 (2002), pp. 7821–7826.
[18] T. Griffiths and Z. Ghahramani, *Infinite latent feature models and the indian buffet process*, tech. rep., University College London, 2005.
[19] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall, *LSD: A fast line segment detector with a false detection control*, IEEE Trans. Pattern Anal Mach. Intell., 32 (2010), pp. 722–732.
[20] R. Halir and J. Flusser, *Numerically stable direct least squares fitting of ellipses*, in WSCG, 1998.
[21] A. Ion, J. Carreira, and C. Sminchisescu, *Image segmentation by figure-ground composition into maximal cliques*, in ICCV, 2011.
[22] A. K. Jain, *Data clustering: 50 years beyond K-means*, Pattern Recognit. Lett., 31 (2010), pp. 651–666.
[23] R. Kannan, S. Vempala, and A. Vetta, *On clusterings: Good, bad and spectral*, Journal of the ACM, 51 (2004), pp. 497–515.
[24] J. Kleinberg, *An impossibility theorem for clustering*, in NIPS, 2002.
[25] D. E. Knuth, *The Stanford GraphBase: a platform for combinatorial computing*, ACM, New York, NY, USA, 1993.
[26] K. Koffka, *Principles of Gestalt Psychology*, Lund Humphries, 1935.
[27] A. Lancichinetti and S. Fortunato, *Limits of modularity maximization in community detection*, Phys. Rev. E, 84 (2011), pp. 066122+.
[28] ———, *Consensus clustering in complex networks*, Sci. Rep., 2 (2012).
[29] A. Lancichinetti, S. Fortunato, and F. Radicchi, *Benchmark graphs for testing community detection algorithms*, Phys. Rev. E, 78 (2008).
[30] M. Lee, H. Shen, J. Z. Huang, and J. S. Marron, *Biclustering via sparse singular value decomposition*, Biometrics, 66 (2010), pp. 1087–1095.
[31] J. Lezama, R. Grompone von Gioi, G. Randall, and J.-M. Morel, *Finding vanishing points via point alignments in image primal and dual domains*, in CVPR, 2014.
[32] N. Li and L. J. Latecki, *Clustering aggregation as maximum-weight independent set*, in NIPS, 2012.
[33] T. Li, C. Ding, and M. I. Jordan, *Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization*, in ICDM, 2007.
[34] M. E. J. Newman and M. Girvan, *Finding and evaluating community structure in networks*, Phys. Rev. E, 69 (2004).
[35] A. Ng, M. Jordan, and Y. Weiss, *On spectral clustering: Analysis and an algorithm*, in NIPS, 2001.
[36] P. Paatero and U. Tapper, *Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values*, Environmetrics, 5 (1994), pp. 111–126.
[37] E. E. Papalexakis, N. D. Sidiropoulos, and R. Bro, *From K-means to higher-way co-clustering: Multilinear decomposition with sparse latent factors*, IEEE Trans. Signal Process., 61 (2013), pp. 493–506.
[38] M. Plantié and M. Crampes, *Survey on social community detection*, in Social Media Retrieval, N. Ramzan, R. Zwol, J.-S. Lee, K. Clver, and X.-S. Hua, eds., Computer Communications and Networks, Springer London, 2013, pp. 65–85.
[39] J. Rabin, J. Delon, Y. Gousseau, and L. Moisan, *MAC-RANSAC: A robust algorithm for the recognition of multiple objects*, in 3DPTV, 2010.
[40] I. Ramírez and M. Tepper, *Bi-clustering via MDL-based matrix factorization*, in CIARP, 2013.

[41] B. Ripley, *The second-order analysis of stationary point processes*, Journal of applied probability, (1976), pp. 255–266.
[42] M. Rosvall and C. Bergstrom, *Maps of random walks on complex networks reveal community structure*, Proc. Natl. Acad. Sci. U.S.A., 105 (2008), pp. 1118–1123.
[43] R. Subbarao and P. Meer, *Nonlinear mean shift over Riemannian manifolds*, Int. J. Comput. Vision, 84 (2009), pp. 1–20.
[44] M. Tepper, P. Musé, and A. Almansa, *Meaningful clustered forest: An automatic and robust clustering algorithm*, (2011), arXiv:1104.0651.
[45] M. Tepper and G. Sapiro, *Ants crawling to discover the community structure in networks*, in CIARP, 2013.
[46] ———, *All for one, one for all, consensus community detection in networks*, in ICASSP, 2014.
[47] ———, *Intersecting 2D lines: a simple method for detecting vanishing points*. submitted to ICIP, 2014.
[48] R. Toldo and A. Fusiello, *Robust multiple structures estimation with J-linkage*, in ECCV, 2008.
[49] A. Traud, P. Mucha, and M. Porter, *Social structure of Facebook networks*, Physica A, 391 (2011), pp. 4165–4180.
[50] R. Tron and R. Vidal, *A benchmark for the comparison of 3-d motion segmentation algorithms*, in CVPR, 2007.
[51] S. Vega-Pons and J. Ruiz-Shulcloper, *A survey of clustering ensemble algorithms*, Int. J. of Pattern Recognit. Artif. Intell., 25 (2011), pp. 337–372.
[52] D. M. Witten, R. Tibshirani, and T. Hastie, *A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis*, Biostatistics, 10 (2009), pp. 515–534.
[53] L. Xu, E. Oja, and P. Kultanen, *A new curve detection method: Randomized Hough transform (RHT)*, Pattern Recognit. Lett., 11 (1990), pp. 331–338.
[54] Y. Xu, W. Yin, Z. Wen, and Y. Zhang, *An alternating direction algorithm for matrix completion with nonnegative factors*, Front. Math. China, 7 (2012), pp. 365–384.
[55] L. Zelnik-Manor and P. Perona, *Self-tuning spectral clustering*, in NIPS, 2004.
[56] W. Zhang and J. Kosecka, *Nonparametric estimation of multiple structures with outliers*, in Workshop on Dynamic Vision, ECCV, 2006.
[57] S.-C. Zhu, *Embedding Gestalt laws in markov random fields*, IEEE Trans. Pattern Anal Mach. Intell., 21 (1999), pp. 1170–1187.
[58] M. Zuliani, C. S. Kenney, and B. S. Manjunath, *The multiRANSAC algorithm and its application to detect planar homographies*, in ICIP, 2005.