



Hierarchical Community Detection by Recursive Partitioning

Tianxi Li , Lihua Lei , Sharmodeep Bhattacharyya , Koen Van den Berge ,
Purnamrita Sarkar , Peter J. Bickel & Elizaveta Levina

To cite this article: Tianxi Li , Lihua Lei , Sharmodeep Bhattacharyya , Koen Van den Berge ,
Purnamrita Sarkar , Peter J. Bickel & Elizaveta Levina (2020): Hierarchical Community
Detection by Recursive Partitioning, Journal of the American Statistical Association, DOI:
[10.1080/01621459.2020.1833888](https://doi.org/10.1080/01621459.2020.1833888)

To link to this article: <https://doi.org/10.1080/01621459.2020.1833888>

 View supplementary material 

 Published online: 24 Nov 2020.

 Submit your article to this journal 

 Article views: 529

 View related articles 

 View Crossmark data 

 Citing articles: 1 View citing articles 



Hierarchical Community Detection by Recursive Partitioning

Tianxi Li^a, Lihua Lei^b, Sharmodeep Bhattacharyya^c, Koen Van den Berge^{d,e}, Purnamrita Sarkar^f, Peter J. Bickel^d, and Elizaveta Levina^g

^aDepartment of Statistics, University of Virginia, Charlottesville, VA; ^bDepartment of Statistics, Stanford University, Stanford, CA; ^cDepartment of Statistics, Oregon State University, Corvallis, OR; ^dDepartment of Statistics, University of California, Berkeley, Berkeley, CA; ^eDepartment of Applied Mathematics, Computer Science and Statistics, Ghent University, Gent, Belgium; ^fDepartment of Statistics and Data Sciences, University of Texas at Austin, Austin, TX; ^gDepartment of Statistics, University of Michigan, Ann Arbor, MI

ABSTRACT

The problem of community detection in networks is usually formulated as finding a single partition of the network into some “correct” number of communities. We argue that it is more interpretable and in some regimes more accurate to construct a hierarchical tree of communities instead. This can be done with a simple top-down recursive partitioning algorithm, starting with a single community and separating the nodes into two communities by spectral clustering repeatedly, until a stopping rule suggests there are no further communities. This class of algorithms is model-free, computationally efficient, and requires no tuning other than selecting a stopping rule. We show that there are regimes where this approach outperforms K -way spectral clustering, and propose a natural framework for analyzing the algorithm's theoretical performance, the binary tree stochastic block model. Under this model, we prove that the algorithm correctly recovers the entire community tree under relatively mild assumptions. We apply the algorithm to a gene network based on gene co-occurrence in 1580 research papers on anemia, and identify six clusters of genes in a meaningful hierarchy. We also illustrate the algorithm on a dataset of statistics papers. Supplementary materials for this article are available online.

ARTICLE HISTORY

Received September 2019
Accepted October 2020

KEYWORDS

Community detection;
Hierarchical clustering;
Network; Recursive
partitioning

1. Introduction

Data collected in the form of networks have become increasingly common in many fields, with interesting scientific phenomena discovered through the analysis of biological, social, ecological, and various other networks; see Newman (2010) for a review. Among various network analysis tasks, community detection has been one of the most studied, due to the ubiquity of communities in different types of networks and the appealing mathematical formulations that lend themselves to analysis; see, for example, reviews by Fortunato (2010), Goldenberg et al. (2010), and Abbe (2018). Community detection is the task of clustering network nodes into groups with similar connection patterns, and in many applications, communities provide a useful and parsimonious representation of the network. There are many statistical models for networks with communities, including the stochastic block model (Holland, Laskey, and Leinhardt 1983) and its many variants and extensions, such as, for example, Handcock, Raftery, and Tantrum (2007), Hoff (2008), Airoldi et al. (2008), Karrer and Newman (2011), Xu and Hero (2013), Zhang, Levina, and Zhu (2014), and Matias and Miele (2017). One large class of methods focuses on fitting such models based on their likelihoods or approximations to them (Bickel and Chen 2009; Mariadassou, Robin, and Vacher 2010; Celisse, Daudin, and Pierre 2012; Amini et al. 2013; Bickel et al. 2013); another class of methods takes an algorithmic approach,

designing algorithms, often based on spectral clustering, that can sometimes be proven to work well under specific models (Newman and Girvan 2004; Newman 2006; Bickel, Chen, and Levina 2011; Rohe, Chatterjee, and Yu 2011; Chen, Sanghavi, and Xu 2012; Zhao, Levina, and Zhu 2012; Chen and Xu 2014; Lei and Rinaldo 2014; Cai and Li 2015; Gao et al. 2016, 2017; Joseph and Yu 2016; Le, Levina, and Vershynin 2017; Amini and Levina 2018).

Most work on community detection to date has focused on finding a single K -way partition of the network into K groups, which are sometimes allowed to overlap. This frequently leads to a mathematical structure that allows for sophisticated analysis, but for larger K these partitions tend to be unstable and not easily interpretable. These methods also typically require the “true” number of clusters K as input. Although various methods have been proposed to estimate K (e.g., Chatterjee 2015; Le and Levina 2015; Wang and Bickel 2017; Chen and Lei 2018; Li, Levina, and Zhu 2020), none of them have been especially tested or studied for large K , and in our experience, empirically they perform poorly when K is large. Finally, a single “true” number of communities may not always be scientifically meaningful, since in practice different community structures can often be observed at different scales.

Communities in real networks are often hierarchically structured, and the hierarchy can be scientifically meaningful, for

example, a phylogenetic tree. A hierarchical tree of communities, with larger communities subdivided into smaller ones further down, offers a natural and very interpretable representation of communities. It also simplifies the task of estimating K , since, instead of estimating a large K from the entire network we only need to check whether a particular subnetwork contains more than one community. We can also view a hierarchy as regularizing an otherwise unwieldy model with a large number of communities, which in theory can approximate any exchangeable graph (Olhede and Wolfe 2014), by imposing structural constraints on parameters. We would expect that for large networks with many communities, such regularization can lead to improvements in both computational costs and theoretical guarantees.

Hierarchical community detection methods can be generally divided into three types: estimating the hierarchy directly and all at once, typically with either a Bayesian or an optimization method; agglomerative algorithms that merge nodes or communities recursively in a bottom-up fashion; and partitioning algorithms which split communities recursively in a top-down fashion. The earliest work in the first category we are aware of is Kleinberg (2002), generalized by Clauset, Moore, and Newman (2008) and Peel and Clauset (2015). These models directly incorporate a tree by modeling connection probabilities between pairs of nodes based on their relative distance on the tree. One line of work treats the tree as a parameter and takes a Bayesian approach (e.g., Clauset, Moore, and Newman 2008; Blundell and Teh 2013). Bayesian inference on these models is computationally prohibitive, and thus infeasible for large networks. Even more importantly, treating each node as a leaf involves a large number of parameters, therefore, sacrificing interpretability. Agglomerative clustering algorithms for networks date back to at least Clauset, Newman, and Moore (2004), which combined well-known Ward's hierarchical clustering (Ward 1963) with the Girvan–Newman modularity (Newman and Girvan 2004) as the objective function. The idea of modularity-based clustering was further explored by Pons and Latapy (2005), Reichardt and Bornholdt (2006), Wakita and Tsurumi (2007), Arenas, Fernandez, and Gomez (2008), and Blondel et al. (2008). Divisive algorithms were once very popular in machine learning problems such as graph partitioning and image segmentation (Spielman and Teng 1996; Shi and Malik 2000; Kannan, Vempala, and Vetta 2004). This class of methods appears to have first been applied to networks by Girvan and Newman (2002), who proposed an “elbow-finding” algorithm deleting the edge with highest betweenness centrality recursively, thereby building a top-down hierarchy. This idea was further developed by Wilkinson and Huberman (2002), Holme, Huss, and Jeong (2003), Gleiser and Danon (2003), and Radicchi et al. (2004).

In spite of many practical benefits and applied work on hierarchical community detection, it is hard to come by a rigorous analysis. The first such analysis of a hierarchical algorithm we are aware of was given by Dasgupta et al. (2006), for a recursive bi-partitioning algorithm based on a modified version of spectral clustering. Their analysis allows for sparse networks with average degree growing poly-logarithmically in n , but the procedure involves multiple tuning parameters with no obvious default values. Later on, Balakrishnan et al. (2011) considered

a top-down hierarchical clustering algorithm based on unnormalized graph Laplacian and the model of Clauset, Moore, and Newman (2008), for a pairwise similarity matrix instead of a network. They did not propose a practical stopping rule, but did provide a rigorous frequentist theoretical guarantee for clustering accuracy. However, as we will further discuss in Section 3, their analysis only works for dense networks which are rare in practice. Lyzinski et al. (2017) proposed another hierarchical model based on a mixture of random dot product graph (RDPG) models (Young and Scheinerman 2007). In contrast to Balakrishnan et al. (2011), they used a two-stage procedure which first detects all communities, and then applies agglomerative hierarchical clustering to build the hierarchy from the bottom up. They proved strong consistency of their algorithm, but it hinges on perfect recovery of all communities in the first step, which leads to very strong requirements on network density.

In this article, we consider a framework for hierarchical community detection based on recursive bi-partitioning, an algorithm similar to Balakrishnan et al. (2011). The algorithm needs a partitioning method, which divides any given network into two, and a stopping rule, which decides if a given network has at least two communities; in principle, any partitioning method and any stopping rule can be used. The algorithm starts by splitting the entire network into two and then tests each resulting leaf with the stopping rule, until the stopping rule indicates there is nothing left to split. We prove that the algorithm consistently recovers the entire hierarchy, including all low-level communities, under the binary tree stochastic block model (BTSBM), a hierarchical network model with communities we propose, in the spirit of Clauset, Moore, and Newman (2008). Our analysis applies to networks with average degree as low as $(\log n)^{2+\epsilon}$ for any $\epsilon > 0$, while existing results either require the degree to be polynomial in n , or $\log^a n$ for large a (e.g., $a = 6$ in Dasgupta et al. (2006)) at the cost of numerous tuning parameters. We also allow the number of communities K to grow with n , which is natural for a hierarchy, at a strictly faster rate than previous work, which for the most part treats K as fixed. Even more importantly, when K is too big to recover the entire tree, we can still consistently recover mega-communities at the higher levels of the hierarchy, whereas K -way clustering will fail. Since the stopping rule only needs to decide whether $K > 1$ rather than estimate K , we can use either hypothesis tests (Bickel and Sarkar 2016; Gao and Lafferty 2017; Jin, Ke, and Luo 2019), or various methods for estimating K . Importantly, the main weakness of methods for of K , which is underestimating K when it is large, since empirically they never underestimate it so severely as to conclude $K = 1$. Unlike previous studies of hierarchical community detection, we are able to provide theoretical guarantees for a data-driven stopping rule rather than known K . Finally, our procedure has better computational complexity than K -way partitioning methods.

The rest of the article is organized as follows. In Section 2, we present our general recursive bi-partitioning framework, a specific recursive algorithm, and discuss the interpretation of the resulting hierarchical structure. In Section 3, we introduce a special class of stochastic block models under which a hierarchy of communities can be naturally defined, and provide theoretical guarantees on recovering the hierarchy for that class of models. Section 4 presents extensive simulation studies

demonstrating advantages of recursive bi-partitioning for both community detection and estimating the hierarchy. [Section 5](#) applies the proposed algorithm to a gene co-occurrence network and obtains a readily interpretable hierarchical community structure. [Section 6](#) concludes with discussion. Proofs and an additional data example on a dataset of statistics papers can be found in the Appendix (supplementary materials).

2. Community Detection by Recursive Partitioning

2.1. Setup and Notation

We assume an undirected network on nodes $1, 2, \dots, n$. The corresponding $n \times n$ symmetric adjacency matrix A is defined by $A_{ij} = 1$ if and only if node i and node j are connected, and 0 otherwise. We use $[n]$ to denote the integer set $\{1, 2, \dots, n\}$. We write I_n for the $n \times n$ identity matrix and $\mathbf{1}_n$ for $n \times 1$ column vector of ones, suppressing the dependence on n when the context makes it clear. For any matrix M , we use $\|M\|$ to denote its spectral norm (the largest singular value of M), and $\|M\|_F$ the Frobenius matrix norm. Community detection will output a partition of nodes into K sets, $V_1 \cup V_2 \cup \dots \cup V_K = [n]$ and $V_i \cap V_j = \emptyset$ for any $i \neq j$, with K typically unknown beforehand.

2.2. The Recursive Partitioning Algorithm

In many network problems where a hierarchical relationship between communities is expected, estimating the hierarchy accurately is just as important as finding the final partition of the nodes. A natural framework for producing a hierarchy is recursive partitioning, an old idea in clustering that has not resurfaced much in the current statistical network analysis literature (e.g., Kannan, Vempala, and Vetta 2004; Dasgupta et al. 2006; Balakrishnan et al. 2011). The framework is general and can be used in combination with any community detection algorithm and model selection method; we will give a few options that worked very well in our experiments. In principle, the output can be any tree, but we focus on binary trees, as is commonly done in hierarchical clustering; we will sometimes refer to partitioning into two communities as bi-partitioning.

Recursive bi-partitioning does exactly what its name suggests:

1. Apply a decision / model selection rule to the network to decide if it contains more than one community. If no, stop; if yes, split into two communities.
2. Repeat step 1 for each of the resulting communities, and continue until no further splits are indicated.

This is a top-down clustering procedure which produces a binary tree, but the leaves are small communities, not necessarily single nodes. Intuitively, as one goes down the tree, the communities become closer, so the tree distance between communities reflects their level of connection.

Computationally, while we do have to partition multiple times, each community detection problem we have to solve is only for $K = 2$, which is faster, easier and more stable than for a general K , and the size of networks decreases as we go down the tree and thus it becomes faster. When K is large and connectivity

levels between different communities are heterogeneous, we expect recursive partitioning to outperform K -way clustering, which does best for small K and when everything is balanced.

We call this approach hierarchical community detection (HCD). As input, it takes a network adjacency matrix A ; an algorithm that takes an adjacency matrix A as input and partitions it into two communities, outputting their two induced submatrices, $\mathcal{C}(A) = \{A_1, A_2\}$; and a stopping rule $\mathcal{S} : \mathbb{R}^{n \times n} \rightarrow \{0, 1\}$, where $\mathcal{S}(A) = 1$ indicates there is no evidence A has communities and we should stop, and $\mathcal{S}(A) = 0$ otherwise. Its output $\mathcal{H}_{\mathcal{C}, \mathcal{S}}(A) = (c, T)$ is the community label vector c and the hierarchical tree of communities T . The algorithm clearly depends on the choice of the partitioning algorithm $\mathcal{C}$ and the stopping rule $\mathcal{S}$; we describe a few specific options next.

2.3. The Choice of Partitioning Method and Stopping Rule

Possibly the simplest partitioning algorithm is a simple eigenvector sign check, used in Balakrishnan et al. (2011), Gao et al. (2017), Le, Levina, and Vershynin (2017), and Abbe et al. (2017):

Algorithm 1. Given the adjacency matrix A :

1. Compute the eigenvector $\tilde{u}_2$ corresponding to the second largest eigenvalue in magnitude of A .
2. Let $\hat{c}(i) = 0$ if $\tilde{u}_{2,i} \geq 0$ and $\hat{c}(i) = 1$ otherwise.
3. Return label $\hat{c}$.

A more general and effective partitioning method is regularized spectral clustering (RSC), especially for sparse networks. Several regularized versions are available; in this article, we use the proposal of Amini et al. (2013), shown to improve performance of spectral clustering for sparse networks (Joseph and Yu 2016; Le, Levina, and Vershynin 2017).

Algorithm 2. Given the adjacency matrix A and regularization parameter τ (by default, we use $\tau = 0.1$), do the following:

1. Compute the regularized adjacency matrix as

$$A_\tau = A + \tau \frac{\bar{d}}{n} \mathbf{1}\mathbf{1}^T,$$

where $\bar{d}$ is the average degree of the network.

2. Let $D_\tau = \text{diag}(d_{\tau 1}, d_{\tau 2}, \dots, d_{\tau n})$ where $d_{\tau i} = \sum_j A_{\tau, ij}$ and calculate the regularized Laplacian

$$L_\tau = D_\tau^{-1/2} A_\tau D_\tau^{-1/2}.$$

3. Compute the leading two eigenvectors of L_τ , arrange them in a $n \times 2$ matrix U , and apply K -means algorithm to the rows, with $K = 2$.
4. Return the cluster labels from the K -means result.

The simplest stopping rule is to fix the depth of the tree in advance, though that is not what we will ultimately do. A number of recent papers focused on estimating the number of communities in a network, typically assuming that each community in the network is generated from either the Erdős–Rényi model or the configuration model (Van Der Hofstad 2016). The methods proposed include directly estimating rank by the USVT method of Chatterjee (2015), hypothesis tests of Bickel

and Sarkar (2016), Gao and Lafferty (2017), and Jin, Ke, and Luo (2019), the BIC criteria of Wang and Bickel (2017), the spectral methods of Le and Levina (2015), and cross-validation methods of Chen and Lei (2018) and Li, Levina, and Zhu (2020). The cross-validation method of Li, Levina, and Zhu (2020) works for both unweighted and weighted networks under a low rank assumption, while the others use the block model assumption.

Under block models, empirically we found that the most accurate and computationally feasible stopping criterion is the non-backtracking method of Le and Levina (2015). Let B_{nb} be the non-backtracking matrix, defined by

$$B_{nb} = \begin{pmatrix} 0 & D - I \\ -I & A \end{pmatrix}. \quad (1)$$

Let $\lambda_i, i \in [2n]$ be the real parts of the eigenvalues of B_{nb} (which may be complex). The number of communities is then estimated as the number of eigenvalues that satisfy $|\lambda_i| > \|B_{nb}\|^{1/2}$. This is because if the network is generated from an SBM with K communities, the largest K eigenvectors of B_{nb} will be well separated from the radius $\|B_{nb}\|^{1/2}$ with high probability, at least in sparse networks (Krzakala et al. 2013; Le and Levina 2015). We approximate the norm $\|B_{nb}\|$ by $\frac{\sum_i d_i^2}{\sum_i d_i} - 1$, as suggested by Le and Levina (2015). For our purposes, we only need the real parts of the two leading eigenvalues, not the full spectrum. If we want to avoid the block model assumption, the edge cross-validation (ECV) method of Li, Levina, and Zhu (2020) can be used instead to check whether a rank 1 model is a good approximation to the subnetwork under consideration.

The main benefit of using these estimators as stopping rules (i.e., checking at every step if the estimated K is greater than 1) is that the tree can be of any form; if we fixed K in advance, we would have to choose in what order to do the splits to end up with exactly the chosen K . Moreover, empirically we found the local stopping criterion is more accurate than directly estimating K , especially with larger K . For the rest of the article, we will focus on two versions, ‘‘HCD-sign’’ which uses splitting by eigenvalue sign (Algorithm 1), and ‘‘HCD-spec,’’ which uses regularized spectral clustering (Algorithm 2). Any of the stopping rules discussed above can be used with either method.

2.4. Mega-Communities and a Similarity Measure for Binary Trees

The final communities (leaves of the tree) as well as the intermediate *mega-communities* can be indexed by their position on the tree. Formally, each node or (mega-)community of the binary tree can be represented by a sequence of binary values $x \in \{0, 1\}^{l_x}$, where l_x is the depth of the node (the root node has depth 0). The string x records the path from the root to the node, with $x_q = 1$ if step q of the path is along the right branch of the split and $x_q = 0$ otherwise. We define the x for the root node to be an empty string. Intuitively, the tree induces a similarity measure between communities: two communities that are split further down the tree should be more similar to each other than two communities that are split higher up. The similarity between two mega-communities does not depend on how they are split further down the tree, which is a desirable feature. Note that we do not assume an underlying hierarchical community model; the tree is simply the output of the HCD algorithm.

To quantify this notion of tree similarity, we define a similarity measure between two nodes x, x' on a binary tree by

$$s(x, x') = \min\{q : x_q \neq x'_q\}.$$

For instance, for the binary tree in Figure 1, we have $s(000, 001) = 3$, while $s(000, 11) = s(000, 110) = s(000, 111) = 1$. Note that comparing values of s is only meaningful for comparing pairs with a common tree node. So $s(000, 111) < s(000, 001)$ indicates that community 000 is closer to community 001 than to 111, but the comparison between $s(000, 001)$ and $s(10, 11)$ is not meaningful.

A natural question is whether this tree structure and the associated similarity measure tell us anything about the underlying population model. Suppose that the network is in fact generated from the SBM. The probability matrix $P = \mathbb{E}A$ under the SBM is block-constant, and applying either HCD-sign or HCD-spec to P will recover the correct communities and produce a binary tree. This binary tree may not be unique; for example, for the planted partition model where all communities have equal sizes, all within-block edge probabilities are a and all between-block edge probabilities are b . However, in many situations P does correspond to a unique binary tree (up to a permutation of labels), for example, under the hierarchical model introduced in Section 3. For the moment, assume this is the case. Let c and T be the binary string community labels and the binary tree produced by applying the HCD algorithm to P in exactly the same way we previously applied it to A . Let $\hat{c}$ and $\hat{T}$ be the result of applying HCD to A . The estimated tree $\hat{T}$ depends on the stopping rule and may be very different in size from T ; however, we can always compute the tree-based similarity between nodes based on their labels. Let $S_T = (s_T(c(i), c(j)))$ be the $n \times n$ matrix where s_T is the pairwise similarities induced by T , and $S_{\hat{T}} = (s_{\hat{T}}(\hat{c}(i), \hat{c}(j)))$ the corresponding similarity matrix based on $\hat{T}$. $S_{\hat{T}}$ can be viewed as an estimate of S_T , and we argue that comparing S_T to $S_{\hat{T}}$ may give a more informative measure of performance than just comparing $\hat{c}$ to c . This is because with a large K and weak signals it may be hard or impossible to estimate all communities correctly, but if the tree gets most of the mega-communities right, it is still a useful and largely correct representation of the network.

Finally, we note that an estimate of S_T under the SBM can be obtained for any community detection method: if $\tilde{c}$ are estimated community labels, we can always estimate the corresponding $\tilde{P}$ under the SBM and apply HCD to $\tilde{P}$ to obtain an estimated tree $\tilde{T}$. However, our empirical results in Section 4 show that applying HCD directly to the adjacency matrix A to obtain $S_{\hat{T}}$ gives a better estimate of S_T than the $S_{\tilde{T}}$ constructed from post-processing the estimated probability matrix produced by a K -way partitioning method.

3. Theoretical Properties of the HCD Algorithm

3.1. The Binary Tree Stochastic Block Model

We now proceed to study the properties of HCD on a class of SBMs that naturally admit a binary tree community structure. We call this class the binary tree stochastic block models (BTSBM), formally defined in Definition 1 and illustrated in Figure 1.

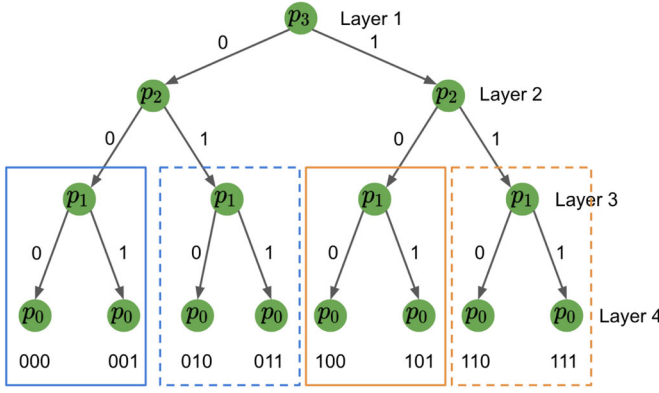


Figure 1. An 8-cluster binary tree SBM. Rectangles correspond to mega-communities.

Definition 1 (BTSBM). Let $S_d := \{0, 1\}^d$ be the set of all length d binary sequences and let $K = |S_d| = 2^d$. Each binary string in S_d encodes a community label and has a 1-1 mapping to an integer in $[K]$ via standard binary representation $\mathcal{I} : S_d \rightarrow [K]$. For node $i \in [n]$, let $c(i) \in S_d$ be its community label, let $C_x = \{i : c(i) = x\}$ be the set of nodes labeled with string $x \in S_d$, and let $n_x = |C_x|$.

1. Let $B \in \mathbb{R}^{K \times K}$ be a matrix of probabilities defined by

$$B_{\mathcal{I}(x), \mathcal{I}(x')} = p_{D(x, x')},$$

where $p_0, p_1, \dots, p_d$ are arbitrary $d+1$ parameters in $[0, 1]$ and

$$D(x, x') = (d + 1 - s(x, x'))I(x \neq x'),$$

for $s(x, x') = \min\{q : x_q \neq x'_q\}$ defined in Section 2.4.

2. Edges between all pairs of distinct nodes i, j are independent Bernoulli, with

$$P(A_{ij} = 1) = B_{\mathcal{I}(c(i)), \mathcal{I}(c(j))}$$

corresponding to the $n \times n$ probability matrix $P = EA$.

For instance, the BTSBM in Figure 1 corresponds to the matrix

$$B = \begin{bmatrix} p_0 & p_1 & p_2 & p_2 & p_3 & p_3 & p_3 & p_3 \\ p_1 & p_0 & p_2 & p_2 & p_3 & p_3 & p_3 & p_3 \\ p_2 & p_2 & p_0 & p_1 & p_3 & p_3 & p_3 & p_3 \\ p_2 & p_2 & p_1 & p_0 & p_3 & p_3 & p_3 & p_3 \\ p_3 & p_3 & p_3 & p_3 & p_0 & p_1 & p_2 & p_2 \\ p_3 & p_3 & p_3 & p_3 & p_1 & p_0 & p_2 & p_2 \\ p_3 & p_3 & p_3 & p_3 & p_2 & p_2 & p_0 & p_1 \\ p_3 & p_3 & p_3 & p_3 & p_2 & p_2 & p_1 & p_0 \end{bmatrix}.$$

A nice consequence of defining community labels through binary strings is that they naturally embed the communities in a binary tree. We can think of each entry of the binary string as representing one level of the tree, with the first digit corresponding to the first split at the top of the tree, and so on. We then define a *mega-community* labeled by a binary string $x \in S_q$ at any level q of the tree as the set $\{i : c(i)_h = x_h, 1 \leq h \leq q\}$, defined on a binary tree T . The mega-communities are unique up to community label permutations, and give a multi-scale view of the community structure; for example, Figure 1 shows

four mega-communities in layer 3 and two mega-communities in layer 2.

The idea of embedding connection probabilities in a tree, to the best of our knowledge, was first introduced as the hierarchical random graph (HRG) by Clauset, Moore, and Newman (2008), and extended by Balakrishnan et al. (2011) to weighted graphs and by Peel and Clauset (2015) to general dendrograms. The BTSBM can be viewed as a hybrid of the original HRG and the SBM, maintaining parsimony by estimating only community-level parameters while imposing a natural and interpretable hierarchical structure. It also provides us with a model that can be used to analyze recursive bi-partitioning on sparse graphs.

3.2. The Eigenstructure of the BTSBM

Let $Z \in \mathbb{R}^{n \times K}$ be the membership matrix with the i th row $Z_i = e_{\mathcal{I}(c(i))}$ containing e_j , the j th canonical basis vector in $\mathbb{R}^K$, where $\mathcal{I}$ is the integer given by the binary representation. Then it is straightforward to show that

$$P = \mathbb{E}A = ZBZ^T - p_0I.$$

The second term comes from the fact that $A_{ii} = 0$. For the rest of the theoretical analysis, we assume equal block sizes, that is,

$$n_1 = n_2 = \dots = n_K = n/K = m. \quad (2)$$

This assumption is stringent but standard in the literature and can be relaxed to a certain extent, as indicated in our Theorem C.3 in Appendix C in the supplementary materials. For the BTSBM, this assumption leads to a particularly simple and elegant eigenstructure for P .

Given a (mega)-community label denoted by a binary string x , we write $x0$ and $x1$ as the binary strings obtained by appending 0 and 1 to x , respectively. We further define $\{x+\}$ to be the set of all binary strings starting with x . The following theorem gives a full characterization of the eigenstructure for the BTSBM.

Theorem 1. Let P be the $n \times n$ community connection probability matrix of the BTSBM with $K = 2^d$ and define $\tilde{P} = P + p_0I = ZBZ^T$. Then the following holds:

1. (Eigenvalues) The distinct nonzero eigenvalues of $\tilde{P}$, denoted by $\lambda_1, \lambda_2, \dots, \lambda_{d+1}$, are given by

$$\lambda_1 = m(p_0 + \sum_{r=1}^d 2^{r-1} p_r),$$

$$\lambda_{q+1} = m \left(p_0 + \sum_{r=1}^{d-q} 2^{r-1} p_r - 2^{d-q} p_{d-q+1} \right), \quad q = 1, \dots, d. \quad (3)$$

2. (Eigenvectors) For any $1 \leq q \leq d$ and each $x \in S_{q-1}$, let v_x^{q+1} be an n -dimensional vector, such that for any $i \in [n]$,

$$v_{x,i}^{q+1} = \begin{cases} 1 & \text{if } c(i) \in S_d \cap \{x0+\}, \\ -1 & \text{if } c(i) \in S_d \cap \{x1+\}, \\ 0 & \text{otherwise.} \end{cases}$$

Then the eigenspace corresponding to eigenvalue λ_{q+1} is spanned by $\{v_x^{q+1} : x \in S_{q-1}\}$. The eigenspace corresponding to λ_1 is spanned by $\mathbf{1}$.

It is easy to see that each v_x^{q+1} corresponds to a split of the two mega-communities in layer q , at an internal tree node x . For instance, consider the colored rectangles in Figure 1, which correspond to $d = 3$ and $q = 2$. The vector v_0^3 has entry 1 for all nodes in the (solid blue) mega-community 00, entry -1 for all nodes in the (dashed blue) mega-community 01 and 0 for all the other nodes, thus separating mega-communities 00 and 01. Similarly, v_1^3 has entry 1 for the nodes in (solid orange) mega-community 10, and entry -1 for nodes in the (dashed orange) mega-community 11. The binary tree structure is thus fully characterized by the signs of eigenvectors' entries. Note that due to multiplicity of eigenvalues, the basis of the eigenspace is not unique. In Appendix A in the supplementary materials, we use another basis which, though less interpretable, is used in the proof of Theorems 2 and 3 to obtain better theoretical guarantees.

While the previous theorem is stated for general configurations of $p_0, \dots, p_d$, the two most natural situations where a hierarchy is meaningful are either *assortative* communities, with

$$p_0 > p_1 > \dots > p_d, \quad (4)$$

or *dis-assortative* communities, with

$$p_0 < p_1 < \dots < p_d. \quad (5)$$

Recall that the HCD-sign algorithm only depends on the eigenvector corresponding to the second largest eigenvalue (in magnitude). Theorem 1 directly implies that under either the assortative or dis-assortative setting, such eigenvalue is unique (has multiplicity 1) with an eigenvector that yields the first split in the tree according to the signs of the corresponding eigenvector entries.

Corollary 1. Let P be the $n \times n$ probability matrix of the BTSBM with $K = 2^d$ and balanced community sizes as in (2). Under either (4) or (5), the second largest eigenvalue (in absolute value) of P for a BTSBM is unique and given by

$$(m-1)p_0 + m \sum_{i=1}^{d-1} 2^{i-1} p_i - m 2^{d-1} p_d$$

and the gap between it and the other eigenvalues is $\Delta_2 = n \min\{p_d, (p_{d-1} - p_d)/2\}$ in the assortative case and $\Delta_2 = n(p_d - p_{d-1})/2$ in the disassortative case. The corresponding (normalized) eigenvector is

$$u_2 = \frac{1}{\sqrt{n}} (\mathbf{1}_{n/2}^T, -\mathbf{1}_{n/2}^T)^T.$$

In a slight abuse of notation, we still denote the k th eigenvalue of P (instead of $\tilde{P}$) by λ_k whenever it is clear from the context.

3.3. Consistency of HCD-Sign Under the BTSBM

The population binary tree T defined in Section 2.4 is unique under the BTSBM, and thus we can evaluate methods under this model on how well they estimate the population tree. Given a community label c and the corresponding balanced binary tree T of depth d , define $\text{mc}(T, c, q) \in [2^q]^n$ to be the community partition of all nodes into the mega-communities at level q

corresponding to c . In particular, at level d , $\text{mc}(T, c, d)$ gives the true community labels c , up to a label permutation. This quantity is well defined only if the binary tree is balanced (i.e., all leaves are at the same depth d), and we will restrict our analysis to the balanced case.

The convention in the literature is to scale all probabilities of connection by a common factor that goes to 0 with n , and have no other dependency on n ; see, for example, the review of Abbe (2018). We similarly reparameterize the BTSBM as

$$(p_0, p_1, \dots, p_d) = \rho_n(1, a_1, \dots, a_d). \quad (6)$$

Let $\tilde{u}_2$ be the eigenvector of the second largest eigenvalue (in magnitude) of A . If

$$\text{sign}(\tilde{u}_{2i}) = \text{sign}(u_{2i}) \text{ for all } i, \quad (7)$$

with high probability, then the first split will achieve exact recovery. A sufficient condition for (7) is concentration of $\tilde{u}_2$ around u_2 in the ℓ_∞ norm. The ℓ_∞ perturbation theory for random matrices is now fairly well studied (e.g., Abbe et al. 2017; Eldridge, Belkin, and Wang 2017). By recursively applying an ℓ_∞ concentration bound, we can guarantee recovery of the entire binary tree with high probability, under regularity conditions.

We start from a condition for the stopping rule. Recall that we defined the stopping rule as a function Ψ such that $\Psi(A) = 1$ indicates the adjacency matrix A contains communities and $\Psi(A) = 0$ indicates there is no evidence of more than one community.

Definition 2. A stopping rule for a network of size n generated from an SBM with K communities is *consistent with rate ϕ* if $\mathbb{P}(\Psi(A) = 1) \geq 1 - n^{-\phi}$ when $K > 1$ and $\mathbb{P}(\Psi(A) = 0) \geq 1 - n^{-\phi}$ when $K = 1$.

With a consistent stopping rule, the strong consistency of binary tree recovery can be guaranteed, as stated in the next two theorems.

Theorem 2 (Consistency of HCD-sign in the assortative setting).

Let $A \in \mathbb{R}^{n \times n}$ be generated from a BTSBM with parameters $(n, \rho_n; a_1, \dots, a_d)$ as defined in (6), with $n = Km = 2^d m$. Let $\hat{c}$ be the community labels and $\hat{T}$ the corresponding binary tree computed with the HCD-sign algorithm with stopping rule Ψ . Suppose the model satisfies the assortative condition (4). Let $a_0 = 1$ and for any $\ell \in [d]$, define

$$\eta_{(\ell)} = \min\{2^{\frac{\ell-r+1}{4}} \eta_{d-r+1} : r \in [\ell]\}, \quad \text{where} \\ \eta_r = \min\{a_r, |a_{r-1} - a_r|/2\}. \quad (8)$$

Fix any $\xi > 1$, $\phi > 1$ and $\phi' > 0$. Then there exists a constant $C(\phi)$, which only depends on ϕ , such that, for any $\ell \in [d]$, if

$$\sqrt{\frac{K^{\ell/d}}{n \rho_n}} \frac{\max\{\log^\xi n, \eta_{(\ell)}^{-1}\}}{\eta_{(\ell)}} < C(\phi), \quad (9)$$

and the stopping rule Ψ is consistent for all the subgraphs corresponding to mega-communities up to the $\ell + 1$ layer with rate ϕ' , then for a sufficiently large n ,

$$\min_{\Pi \in \text{Perm}(q)} \Pi(\text{mc}(\hat{T}, \hat{c}, q)) = \text{mc}(T, c, q), \quad \text{for all } q \leq \ell,$$

with probability at least $1 - 2K^{(\phi+1)\ell/d}n^{-\phi} - K^{(\phi'+1)\ell/d}n^{-\phi'}$. The mega-community partition $\text{mc}(T, c, q)$ is defined at start of Section 3.3 and $\text{Perm}(q)$ is the set of all label permutations on the binary string set S_q . Further, if the conditions hold for $\ell = d$, then with probability at least $1 - 2K^{(\phi+1)}n^{-\phi} - 2K^{(\phi'+1)}n^{-\phi'}$, the algorithm exactly recovers the entire binary tree and stops immediately after it is recovered.

Theorem 2 essentially says that each splitting step of HCD-sign consistently recovers the corresponding mega-community, provided that the condition (9) holds for that layer. Note that, according to (8), in the assortative setting,

$$\begin{aligned} \sqrt{\frac{K^{\ell/d}}{n\rho_n}} \frac{1}{\eta_{(\ell)}^2} &= \sqrt{\frac{2^\ell}{n\rho_n}} \frac{1}{\eta_{(\ell)}^2} \\ &= \frac{1}{\sqrt{n\rho_n} \min\{2^{-\frac{r-1}{2}} \eta_{d-(r-1)}^2, r \in [\ell]\}}. \end{aligned}$$

As ℓ increases, the set over which we minimize grows, while each individual term remains the same. Thus, the whole term increases with ℓ . We also have

$$\sqrt{\frac{K^{\ell/d} \log^\xi n}{n\rho_n}} \frac{1}{\eta_{(\ell)}} = \frac{2^{\ell/4} \log^\xi n}{\sqrt{n\rho_n}} \frac{1}{\min\{2^{-\frac{r-1}{4}} \eta_{d-(r-1)}, r \in [\ell]\}},$$

which also increases in ℓ . Therefore, (9) gets strictly harder to satisfy as ℓ increases, and even if recovering the entire tree is intrinsically hard or simply impossible (condition (9) fails to hold for $\ell = d$), HCD-sign can still consistently recover mega-communities at higher levels of the hierarchy, as long as they satisfy the condition. This is a major practical advantage of recursive partitioning compared to both K -way partitioning and agglomerative hierarchical clustering of Lyzinski et al. (2017). A similar result holds in the dis-assortative setting.

Theorem 3 (Consistency of HCD-sign in the dis-assortative setting). Suppose the model satisfied the dis-assortative condition (5). Under the setting of Theorem 2, the conclusion continues to hold if (9) is replaced by

$$\sqrt{\frac{K^{\ell/d}}{n\rho_n}} \frac{v_{(\ell)} \max\{\log^\xi n, v_{(\ell)} \eta_{(\ell)}^{-1}\}}{\eta_{(\ell)}} < C(\phi), \quad (10)$$

where

$$v_{(\ell)} = \max \left\{ 2^{-\frac{\ell-r+1}{2}} a_{d-r+1} : r \in [\ell] \right\}. \quad (11)$$

It is easy to verify that condition (10) also becomes harder to satisfy for larger ℓ . Therefore in dis-assortative settings, we may also be able to recover mega-communities even if we cannot recover the whole tree.

The theorems apply to any consistent stopping rule satisfying Definition 2. In particular, the non-backtracking matrix method we use in the implementation is a consistent stopping rule based on the recently updated result of Le and Levina (2015), as we show next.

Proposition 1. Define

$$\zeta_{(\ell)} = \min \left\{ \frac{(1 + \sum_{j=1}^{\ell-r} 2^{j-1} a_j - 2^{\ell-r} a_{\ell-r+1})^2}{1 + \sum_{j=1}^{\ell-r+1} 2^{j-1} a_j}, r \in [\ell] \right\}. \quad (12)$$

If

$$\frac{K}{n\rho_n} \log n \leq \min \left\{ \frac{\zeta_{(\ell)}}{25} \log n, 1 + \sum_{j=1}^{d-\ell+1} 2^{j-1} a_j \right\}, \quad (13)$$

and

$$n\rho_n \max\{1, a_d\} \leq n^{2/13}, \quad (14)$$

then for a sufficiently large n , the non-backtracking matrix stopping rule, described in (1), is consistent with rate 1 under BTSBM for all mega-communities up to the layer $\ell + 1$.

Proposition 1 directly implies that if (13), (14), and (9) or (10) hold at the same time, the conclusions of Theorem 2 and Theorem 3 hold when using the non-backtracking matrix as the stopping rule. In the proposition, condition (14) constrains the network from being too dense. We believe this to be an artifact of the proof technique of Le and Levina (2015). Intuitively, if the method works for a sparser network, it should work for a denser one as well, so we expect this condition can be removed, but we do not pursue this direction since the non-backtracking estimator is not the focus of the present article. A similar argument shows that another class of stopping rules based on Bethe–Hessian matrices (Le and Levina 2015) also gives consistent stopping rules which will also give consistent recovery.

Next, we illustrate conditions (13) and (9) in a simplified setting. Consider the assortative setting when the whole tree can be recovered ($\ell = d$).

Example 1. Assume an arithmetic sequence a_r , given by $a_r = 1 - r/(d+1)$. In this case, taking $\xi = 1.01$, it is easy to see that (9) is

$$K(d+1)^2 \log^{2.02} n = O(n\rho_n). \quad (15)$$

Some simple algebra shows that condition (13) simplifies to

$$K \log n = O(n\rho_n) \quad \text{and} \quad Kd^2 = O(n\rho_n).$$

Thus the additional requirement for strong consistency with the non-backtracking matrix stopping rule is redundant and we only need the condition

$$K(d+1)^2 \log^{2.02} n = O(n\rho_n).$$

Example 2. Assume a geometric sequence a_r , given by $a_r = \beta^r$ for a constant $\beta < 1$. Let $\beta = 2^{-\gamma}$ for $\gamma > 0$. The condition (13), after some simplifications, becomes

$$K \log n = O(n\rho_n).$$

On the other hand, for (9), we have

$$\begin{aligned} 2^{r/4} \eta_r &= \min\{2^{r/4} a_r, 2^{r/4} (a_{r-1} - a_r)/2\} \\ &= (2^{1/4-\gamma})^r \min \left\{ 1, \frac{\beta^{-1} - 1}{2} \right\}. \end{aligned}$$

If $\gamma \leq 1/4$, then $\eta_{(d)} \geq \min \left\{ 1, \frac{\beta^{-1}-1}{2} \right\}$, and (9) with $\xi = 1.01$ becomes

$$K \log^{2.02} n = O(n\rho_n).$$

If $\gamma > 1/4$, then $\eta_{(d)} = (2^{1/4-\gamma})^d \min \left\{ 1, \frac{\beta^{-1}-1}{2} \right\} = \Theta(K^{1/4-\gamma})$, and (9) becomes

$$K^{4\gamma} = O(n\rho_n) \quad \text{and} \quad K^{2\gamma+1/2} \log^{2.02} n = O(n\rho_n).$$

In summary, the following conditions are sufficient for exact recovery:

$$K^{4\gamma} = O(n\rho_n) \quad \text{and} \quad K^{\max\{2\gamma+1/2, 1\}} \log^{2.02} n = O(n\rho_n). \quad (16)$$

3.4. Comparison With Existing Theoretical Guarantees

In this section, we compare our result with other strong consistency results for recursive bi-partitioning. We focus on the assortative setting since this most existing results require it (Balakrishnan et al. 2011; Lyzinski et al. 2017). To make this comparison, we have to ignore the stopping rule, since no other method showed consistency for a data-driven stopping rule.

Strong consistency of recursive bi-partitioning was previously discussed by Dasgupta et al. (2006). Their algorithm is far more complicated than ours, with multiple rounds of resampling and pruning, and its computational cost is much higher. For comparison, we can rewrite their assumptions in the BTSBM parameterization. Their Theorem 1 requires $n\rho_n \geq \log^6 n$, and the gap between any two columns of P corresponding to nodes in different communities to satisfy

$$\min_{c(u) \neq c(v)} \|P_{\cdot u} - P_{\cdot v}\|_2^2 = \Omega(K^6 \rho_n \log n). \quad (17)$$

Under the BTSBM, it is straightforward to show that the minimum in (17) is achieved by two communities corresponding to sibling leaves in the last layer and

$$\min_{c(u) \neq c(v)} \|P_{\cdot u} - P_{\cdot v}\|_2^2 = 2 \frac{n}{K} \rho_n^2 (1 - a_1)^2.$$

Assume that $K = O(n^\omega)$ for some $\omega < 1$. Then for sufficiently large $\phi > 0$, $K^{\phi+1} n^{-\phi} = o(1)$. To compare our result (with $\ell = d$) to (17), we consider two cases.

1. Arithmetic sequence: Suppose that a_r is given by $a_r = 1 - r/(d+1)$. Then (17) gives

$$K^7 (d+1)^2 \log n = O(n\rho_n) \iff K^7 \log^2 K \log n = O(n\rho_n),$$

whereas our condition (15) is only

$$K(d+1)^2 \log^{2.02} n = O(n\rho_n) \iff K \log^2 K \log^{2.02} n = O(n\rho_n),$$

which has a much better dependence on K .

2. Geometric sequence: Suppose that a_r is given by $a_r = \beta^r$. Then the condition (17) becomes

$$K^7 \log n = O(n\rho_n).$$

From (16), it is easy to see that HCD-sign has a better rate in K if $\gamma < 7/4$, which is equivalent to $\beta > 2^{-7/4} = 0.2973$.

The algorithm proposed in Balakrishnan et al. (2011) is similar to ours, though their analysis is rather different. Under BTSBM, each entry in the adjacency matrix is sub-Gaussian with parameter 1 (which cannot be improved), that is, $\mathbb{E} \exp\{a(A_{ij} - \mathbb{E}A_{ij})\} \leq \exp\{a^2/2\}$ for any $a \geq 0$. To recover all mega-communities up to layer ℓ (with size at least $n/2^\ell$), it is easy to show that a necessary condition in their analysis (Theorem 1) is

$$\rho_n = \omega(n^{15/16}).$$

Thus, the consistency result in Balakrishnan et al. (2011) only applies to dense graphs.

Lyzinski et al. (2017) developed their hierarchical community detection algorithm under a different model they called the hierarchical stochastic blockmodel (HSBM). The HSBM is defined recursively, as a mixture of lower level models. In particular, when $a_r = \beta^r$ with $\beta < 1/2$, the BTSBM is a special case of the HSBM where each level of the hierarchy has exactly two communities. Lyzinski et al. (2017) showed the exact tree recovery for fixed K and the average expected degree of at least $O(\sqrt{n} \log^2 n)$, implying a very dense network. By contrast, our result allows for a growing K , and if K is fixed, the average degree only needs to grow as fast as $\log^{1+\xi} n$ for an arbitrary $\xi > 1$, which is a much weaker requirement, especially considering that a degree of order $\log n$ is necessary for strong consistency of community labels under a standard SBM with fixed K .

3.5. Computational Complexity

We conclude this section by investigating the computational complexity of HCD, which turns out to be better than that of K -way partitioning, especially for problems with a large number of communities. The intuition behind this somewhat surprising result is that, even though HCD has to perform clustering multiple times, it performs a much simpler task at each step, computing no more than two eigenvectors instead of K , and the number of nodes to cluster decreases after each step.

We start with stating some relevant known facts. Suppose we use Lloyd's algorithm (Lloyd 1982) for K -means and the Lanczos algorithm (Larsen 1998) to compute the spectrum, both with a fixed number of iterations. If the input matrix is $d_1 \times d_2$, the K -means algorithm has complexity of $O(d_1 d_2 K)$. In calculating the leading K eigenvectors, we take advantage of matrix sparsity, resulting in complexity $O(\|A\|_0 K)$ where $\|A\|_0$ is the number of nonzero entries of A . Therefore, for K -way spectral clustering, the computational cost is

$$O(nK^2 + \|A\|_0 K). \quad (18)$$

Turning to HCD, let $A_{j,\ell}$ denote the adjacency matrix of the j th block in ℓ th layer. For comparison purposes, we assume the BTSBM and the conditions for exact recovery, so that we construct the entire tree. Then $A_{j,\ell}$ corresponds to $2^{d-\ell} m$ nodes. Note that for both Algorithms 1 and 2, the complexity is linear in size. As with (18), the splitting step applied to $A_{j,\ell}$ has complexity $O(2^{d-\ell} m + \|A_{j,\ell}\|_0)$ for both HCD-sign and HCD-spec. Adding

the cost over all layers, the total computation cost becomes

$$\begin{aligned} & O\left(\sum_{\ell=1}^d \sum_{j=1}^{2^\ell} (2^{d-\ell} m + \|A_{j,\ell}\|_0)\right) \\ &= O\left(n \log K + \sum_{\ell=1}^d \sum_{j=1}^{2^\ell} \|A_{j,\ell}\|_0\right), \end{aligned} \quad (19)$$

where we use the facts that $K = 2^d$ and $n = mK$. Since the blocks corresponding in ℓ th layer are disjoint,

$$\sum_{j=1}^{2^\ell} \|A_{j,\ell}\|_0 \leq \|A\|_0, \quad (20)$$

and thus (19) is upper bounded by

$$O(n \log K + \|A\|_0 \log K). \quad (21)$$

This is strictly better than the complexity of K -way spectral clustering (18) for large K .

Moreover, the inequality (20) may be overly conservative. Under the BTSBM, the expected number of within-block edges in the ℓ th layer is

$$\begin{aligned} \mathbb{E} \sum_{j=1}^{2^\ell} \|A_{j,\ell}\|_0 &= 2^\ell \left(2^{d-\ell} m\right)^2 \bar{p}_\ell, \quad \text{where} \\ \bar{p}_\ell &= \frac{p_0 + \sum_{i=1}^{d-\ell} 2^{i-1} p_i}{2^{d-\ell}}. \end{aligned}$$

As a result,

$$\begin{aligned} \mathbb{E} \sum_{\ell=1}^d \sum_{j=1}^{2^\ell} \|A_{j,\ell}\|_0 &= \left(\sum_{\ell=1}^d \frac{\bar{p}_\ell}{2^\ell \bar{p}_0}\right) \mathbb{E} \|A\|_0 \\ &= \left(\frac{dp_0 + \sum_{i=1}^d (d-i)2^{i-1} p_i}{p_0 + \sum_{i=1}^d 2^{i-1} p_i}\right) \mathbb{E} \|A\|_0. \end{aligned}$$

The coefficient before $\mathbb{E} \|A\|_0$ is $O(1)$ in many situations, including Examples 1 and 2 discussed at the end of Section 3.3. In these cases, the average complexity of HCD algorithms is only

$$O(n \log K + \mathbb{E} \|A\|_0).$$

Last but not least, the HCD framework, unlike K -way partitioning, can be easily parallelized.

4. Numerical Results on Synthetic Networks

In this section, we investigate empirical performance of HCD on synthetic networks, both on networks with hierarchical structure and those without. We generate networks with hierarchical structure from the proposed BTSBM, and consider BTSBMs with a fully balanced tree and equal block sizes, which are covered by our theory, as well as BTSBMs with unbalanced trees and different block sizes, which are not. The nonhierarchical networks will be generated from the regular SBM, included as a general sanity check.

We compare methods using the following five measures of accuracy.

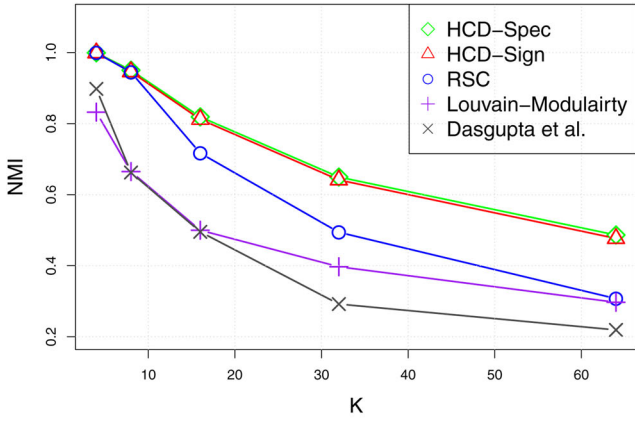
1. Accuracy of community detection, measured by normalized mutual information (NMI) between true and estimated labels (Yao 2003). NMI is commonly used in the network literature (Lancichinetti and Fortunato 2009; Amini et al. 2013) and does not require the two sets of labels to have the same number of clusters, a property we need when comparing trees.
2. Accuracy of recovering the hierarchical structure, measured by the error in the tree distance matrix $h\|S_{\hat{T}} - S_T\|_F^2 / \|S_T\|_F^2$, where S_T is defined in Section 2.4. For baseline methods that do not output a hierarchy, we apply the HCD procedure to their estimate of the probability matrix estimated to construct a tree.
3. Accuracy of mega-community detection at the top two layers, measured by the proportion of correctly clustered nodes. We only compute this for balanced BTSBMs.
4. Accuracy of estimating the probability matrix P , measured by $\|\hat{P} - P\|_F^2 / \|P\|_F^2$.
5. Accuracy of estimating K , measured by comparing the average $\hat{K}$ with K . Spectral clustering is excluded since it requires K as input; instead, we report the error of the non-backtrackign estimator of K , shown to be one of the most accurate options available for the SBM (Le and Levina 2015).

We compare the two versions of HCD (sign-based and spectral clustering) with three other baseline methods: regularized spectral clustering, the Louvain's modularity method (Blondel et al. 2008), which is regarded as a most competitive modularity based community detection algorithms by empirical studies (Yang, Algesheimer, and Tessone 2016), and the recursive partitioning method of Dasgupta et al. (2006). The first two methods are not hierarchical. Spectral clustering requires K as an input, and we use the K estimated by hierarchical spectral clustering (HCD-spec) for a fair comparison. For mega-communities, we also include RSC with the true number of clusters ($K = 2$ for layer 1 and $K = 4$ for layer 2), which can be viewed as an oracle version of RSC. We use the default parameter settings with regularized spectral clustering and Louvain modularity. Unfortunately, the algorithm of Dasgupta et al. (2006) involves several tuning parameters and the article does not provide recommendations on how to tune them. For the purposes of this comparison, we tried a number of settings and used a configuration that seemed the best on average.

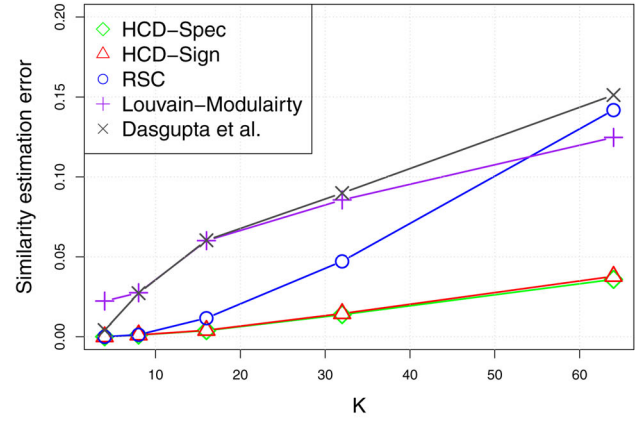
While we include multiple benchmarks, the main focus of our comparisons is on hierarchical versus K -way clustering implemented by the same method (regularized spectral clustering), since it focuses on the central contribution of this work. All simulation results are averaged over 100 independent replications.

4.1. Balanced BTSBM

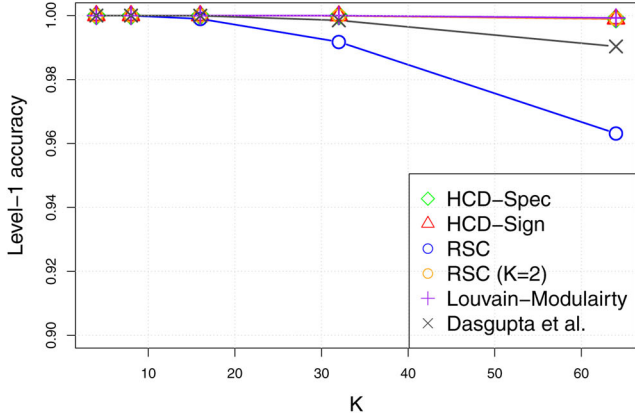
We start with a balanced BTSBM with $n = 3200 = m2^d$, $K = 2^d$, and the values of $d = 2, 3, 4, 5, 6$. The parameter β is set so that the average out-in ratio (between-block edge/within-block edges) for all K is fixed at 0.15 and ρ_n is set so that the average node degree is 50. These values of β and ρ_n are not too challenging, so we can be sure that the main impact on accuracy comes from changing K .



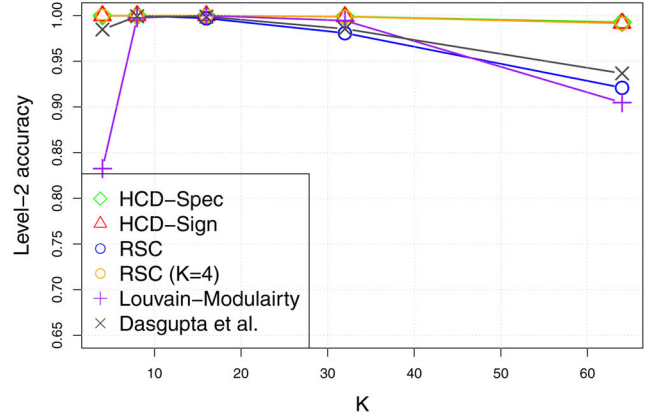
(a) Normalized mutual information



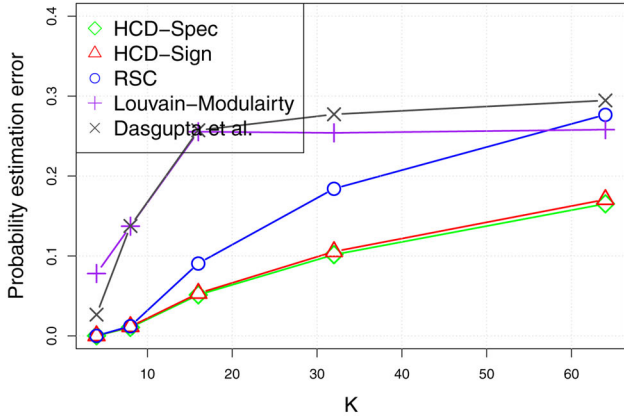
(b) Error in the tree similarity matrix



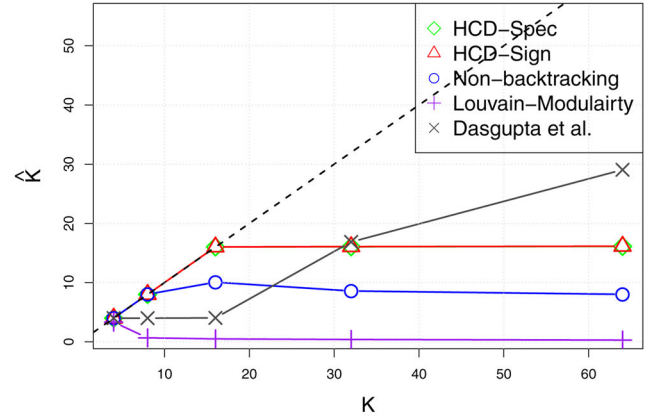
(c) Accuracy for level 1



(d) Accuracy for level 2



(e) Error in the probability matrix

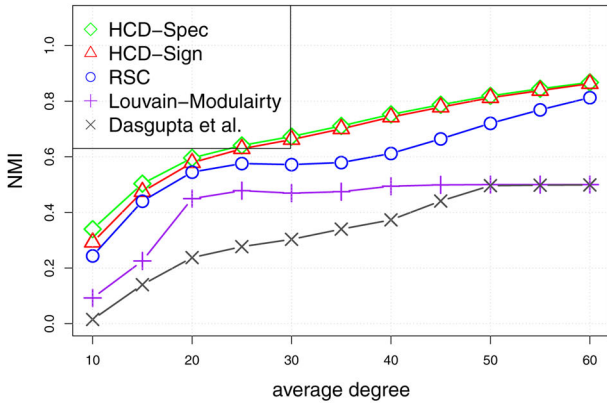


(f) Estimated number of communities

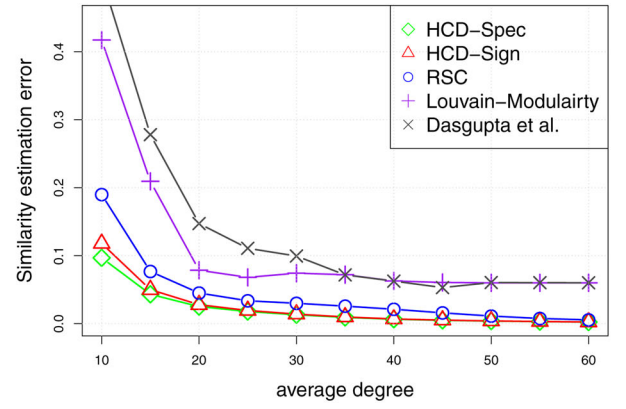
Figure 2. Results for the balanced BTSBM with $n = 3200$ nodes and K varying in $\{4, 8, 16, 32, 64\}$. For (a), (c), and (d) higher values indicate better performance; for (b) and (e), lower values are better. For (f), the truth is shown as the 45° line.

Figure 2 shows the results as a function of K . First, the difference between the two versions of HCD is negligible, with the RSC-based splitting slightly better, so we will simply refer to HCD in comparisons. HCD outperforms all other methods on both finding all the communities and constructing the hierarchy (Figures 2(a) and (b)), and as predicted by theory, the gain increases with K . While all methods' performance on recovering all communities gets worse as K increases, Figures 2(c) and (d)

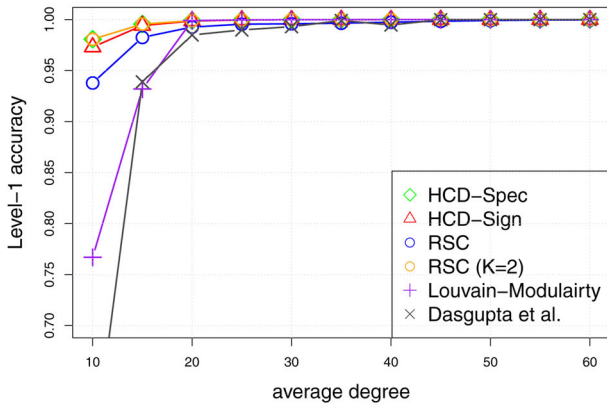
show that HCD recovers the two top layers of the tree essentially perfectly for all values of K , while other methods do not. This is also consistent with theory which predicts that hierarchical clustering can retain full accuracy on the top levels of the tree. HCD methods also consistently outperform other methods on estimating the probability matrix (Figure 2(e)), as one would expect since the probability matrix depends on accurate community estimation. Finally, while all methods underestimate the



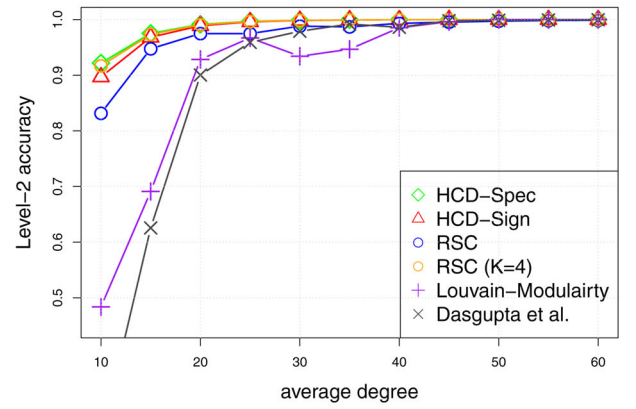
(a) Normalized mutual information



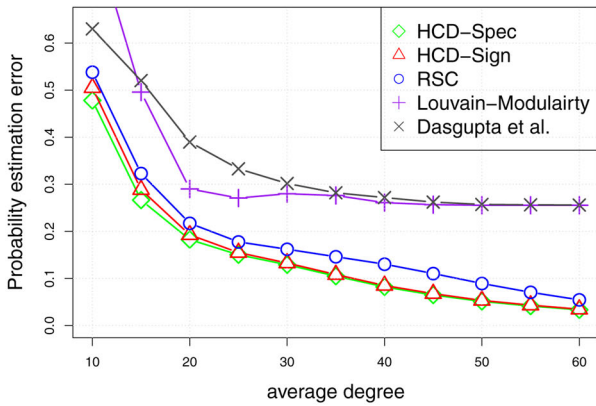
(b) Error in the tree similarity matrix



(c) Accuracy for level 1 mega-communities



(d) Accuracy for level 2 mega-communities



(e) Error in the probability matrix

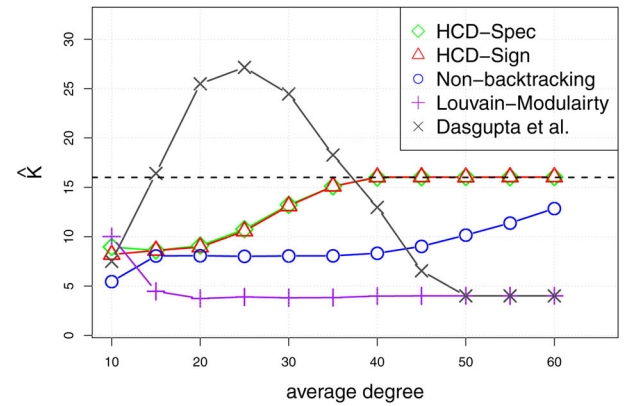
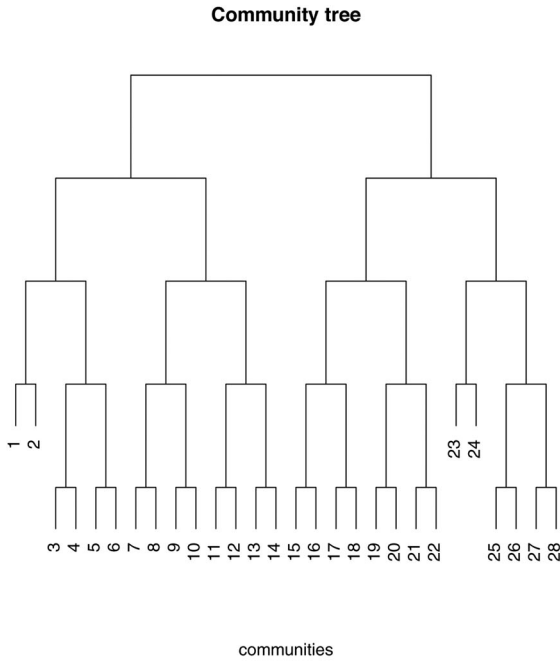
(f) Estimated number of communities $\hat{K}$

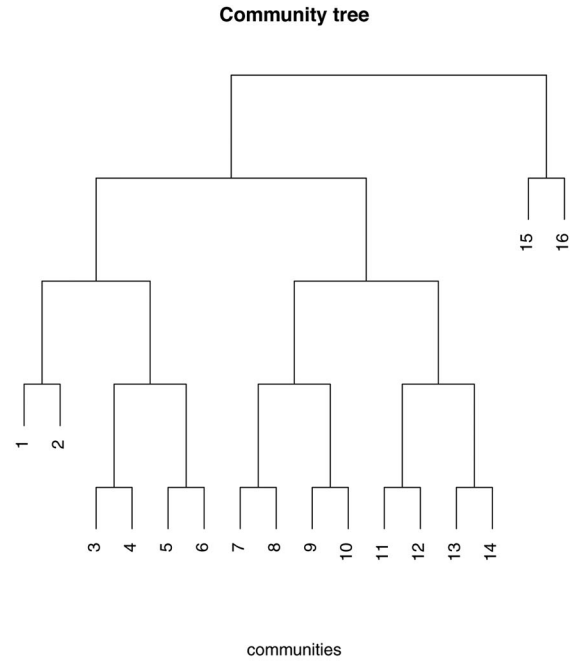
Figure 3. Results for the balanced BTSBM with $n = 3200$ nodes and varying average degree. For (a), (c), and (d) higher values indicate better performance; for (b) and (e), lower values are better. For (f), the truth is shown as the horizontal line.

number of communities when K is large, HCD is more accurate than other methods, except for $K = 64$ when Dasgupta's method is better. In summary, as K grows and the problem becomes more challenging, the advantages of HCD become more and more pronounced.

Next, we use the same configuration, except now we vary the average degree of the network, fixing $K = 16$, and holding the out-in ratio at 0.15. Results are shown in Figure 3. Again, HCD performs better and the accuracy improves with degree as suggested by theory. The errors of the modularity method and



(a) Example 1



(b) Example 2

Figure 4. Examples of unbalanced community trees.

Dasgupta et al. (2006), on the other hand, appear to converge to a constant as the degree increases and do not suggest consistency will hold in this setting. With an exception of one value of K where Dasgupta's method is closer to the true value of K , the HCD algorithm gives the best performance for all tasks.

4.2. Unbalanced BTSBM With a Complex Tree Structure

The BTSBM gives us the flexibility to generate complex tree structures and communities of varying sizes. However, it is difficult to control these features with a single parameter such as K or the average degree, so instead we just include two specific examples as an illustration. The first example corresponds to the hierarchical community structure shown in Figure 4(a). It is generated by merging 4 pairs of the original communities from a balanced BTSBM with 32 communities, resulting in $K = 28$ total, with 4 communities of 200 nodes each and 24 communities of 100 nodes each. This is a challenging community detection problem because of the large K , and the varying community sizes make it harder. The second example is shown in Figure 4(b). It is generated by merging 2 pairs of leaves one level up, and 8 pairs three levels up in 32 balanced communities again, thus making it even more unbalanced. This tree has two communities with 800 nodes, two with 200 each, and the remaining 12 communities have 100 nodes each. In both examples, the average degree is 35.

Table 1 shows performance for these two examples. The HCD methods perform better on all tasks, which matches what we observed in balanced settings. All of the methods give reason clustering for level-1 and level-2 mega-communities but HCD methods are clear advantage as one moves down the tree as indicated by NMI, probability estimation error, and similarity

Table 1. Clustering and model estimation accuracy for Examples 1 and 2.

	Performance metric	HCD-sign	HCD-spec	RSC	Modularity	Dasgupta
Example 1	NMI	0.665	0.677	0.552	0.386	0.282
	Similarity error	0.015	0.014	0.043	0.103	0.107
	$\hat{P}$ error	0.134	0.128	0.214	0.343	0.352
	level-1 accuracy	0.999	0.999	0.992	0.996	0.988
	level-2 accuracy	0.998	0.998	0.73	0.892	0.973
Example 2	NMI	0.753	0.764	0.626	0.533	0.335
	Similarity error	0.025	0.025	0.033	0.085	0.085
	$\hat{P}$ error	0.080	0.075	0.129	0.223	0.236
	level-1 accuracy	0.999	0.999	0.993	0.999	0.997
	level-2 accuracy	0.998	0.999	0.794	0.872	0.982

Table 2. Average $\hat{K}$ in Examples 1 and 2.

	Performance metric	HCD-sign	HCD-spec	NB	Modularity	Dasgupta
Example 1 ($K = 28$)	$\hat{K}$	16.19	16.27	10.45	3.70	24.03
Example 2 ($K = 16$)	$\hat{K}$	10.11	10.11	7.43	3.53	23.75

recovery. Table 2 shows the average selected $\hat{K}$, and the HCD methods are also more accurate in model selection than the others and is inferior to Dasgupta's method for Example 1. However, this only exception may be due to the over-selection of Dasgupta's method we generally observed.

4.3. Networks With No Hierarchical Communities

As a benchmark, we also test HCD methods on networks generated from the SBM without a hierarchical structure. These

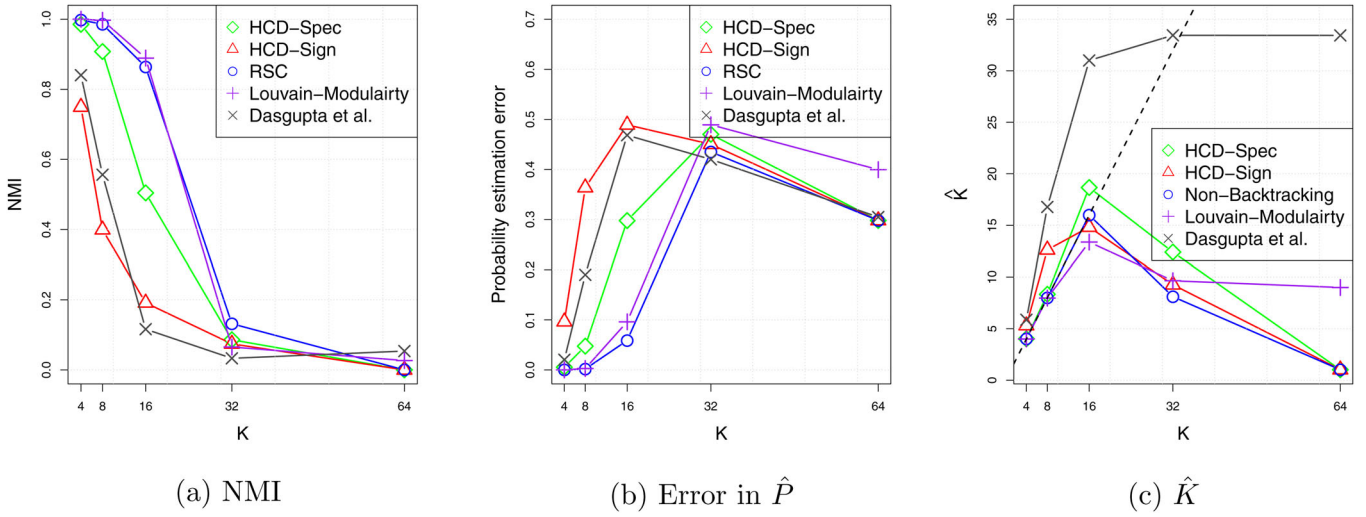


Figure 5. Results for all methods on the SBM with no hierarchy with K varied and the average degree being 50.

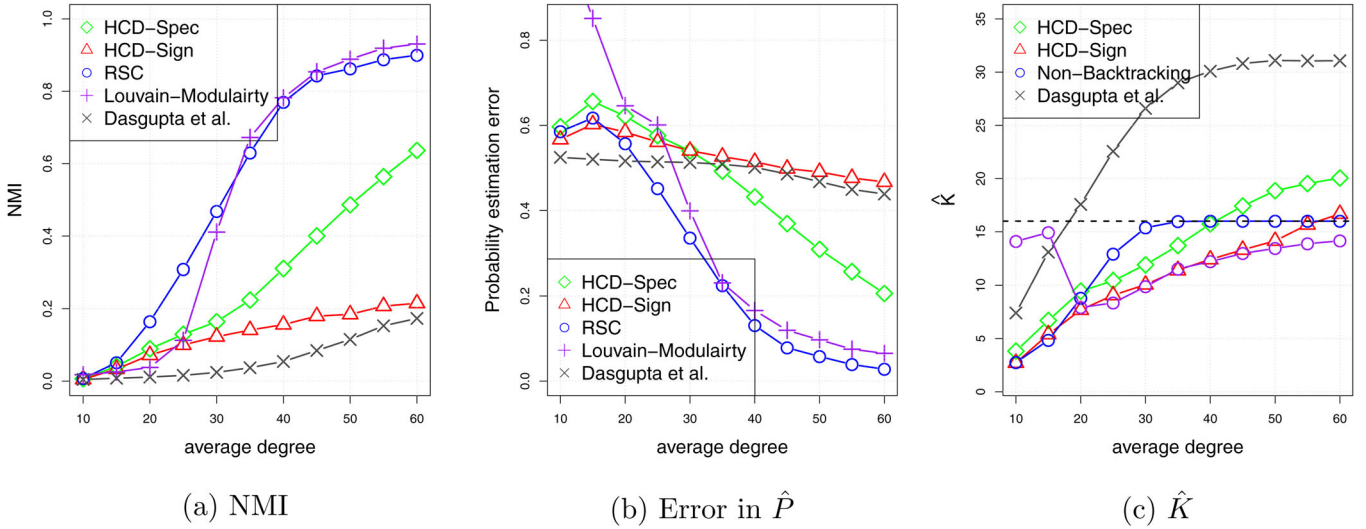


Figure 6. Results for all methods on the SBM with no hierarchy with average degree varied. K is fixed to be 16 in this example.

networks are generated from the planted-partition version of the SBM, with $P_{ij} = p$ if $c(i) = c(j)$ and $P_{ij} = \beta p$ if $c(i) \neq c(j)$. Since all between-community probabilities are the same, no meaningful hierarchy can be defined. To make results fully comparable to the previous settings, we still use $\hat{K}$ estimated by HCD-spec as input to RSC.

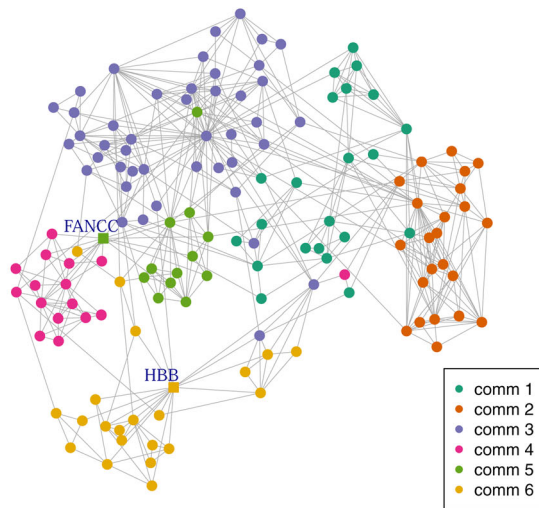
Figures 5 and 6 show performance of all the methods under consideration on clustering accuracy (measured by NMI), estimating the probability matrix, and estimating K , Figure 5 as a function of K (with average degree set to be 50) and Figure 6 as a function of average degree (with $K = 16$). In this nonhierarchical standard SBM setting, the non-backtracking method estimator gives the best estimate of K and regularized spectral clustering and Louvain modularity have the best clustering accuracy, which is not surprising. Unlike in hierarchical settings considered so far, HCD-spec works substantially better than HCD-sign, but is still inferior to RSC. Louvain modularity can be slightly better than RSC in easy settings (small K or high degree), but it degrades quickly as the problem becomes harder. We also observed that the both non-backtracking method and the HCD estimate a small K when the problem becomes really

difficult, which could be seen as indirect evidence that there is just not enough signal to find communities in those settings.

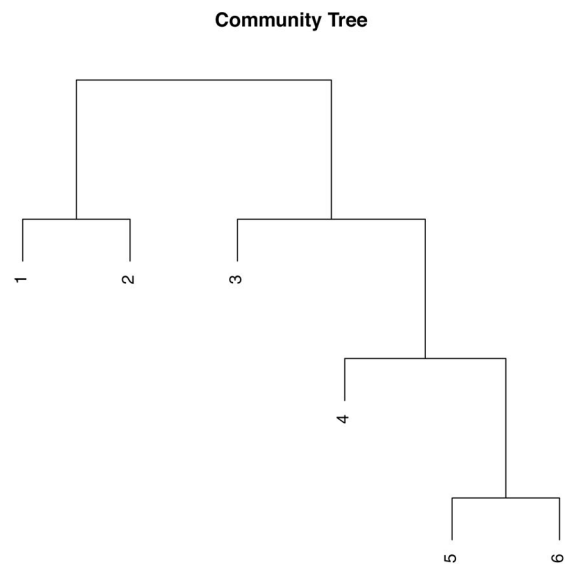
Results in this nonhierarchical setting match empirical observations on clustering in Euclidean space from computer science (Shi and Malik 2000; Kannan, Vempala, and Vetta 2004). In practice, we will not know whether the community structure is hierarchical, although the completely flat structure of the planted partition model is not likely to occur naturally. However, as Figure 5 shows, all methods fail completely with $K > 16$ if there is no hierarchical structure; it might be that for large K community detection is the only possible when there is a hierarchy anyway.

5. Hierarchical Communities in Genes Associated With Anemia

We illustrate the HCD method by fitting hierarchical communities to a network of genes that have been found to associate with anemia. Another detailed example of hierarchical communities of topics in statistical research literature is included in Appendix F in the supplementary materials.



(a) The network and detected communities.



(b) Estimated hierarchy of communities.

Figure 7. Hierarchical communities in a network of 140 genes associated with anemia.

Anemia is a blood disease that is caused by either a decrease in the number of red blood cells, or a general lower capability of the blood for oxygen transport. Arguably the most common blood disorder, it was estimated by the World Health Organization to affect roughly a quarter of humans globally in 2005 (De Benoist et al. 2008). Causes for anemia include acute blood loss, an increased breakdown of red blood cell populations or a decreased production of red blood cells. The causes for an increased breakdown of red blood cells also include genetic conditions, for example, sickle-cell anemia. Here, we investigate associations between genes in the context of anemia using the DigSee framework (Kim, Kim, and Lee 2017), which identifies disease-related genes from Medline abstracts. We look for associations of anemia with biological events related to mutation, gene expression, regulation and transcription. A total of 4449 papers were found that link genes to anemia. We only retain papers that link at least two genes to anemia, resulting in a set of 1580 papers that relate anemia to the biological events defined above. In total, 1657 genes are mentioned across these papers. We represent the gene co-occurrence information by a network in which two genes are connected if they were mentioned together in at least two papers. Following Wang and Rohe (2016), we focus on the 3-core of the largest connected component of the network, ignoring the periphery which typically carries little information about the structure of the core. The resulting network, shown in Figure 7(a), has 140 nodes (genes) and the average node degree is 7.03. Genes are frequently organized into trees, for instance, the gene ontology tree (Ashburner et al. 2000; Mistry and Pavlidis 2008), where functionally similar genes are expected to be close to each other in the tree. While co-occurrence in academic research papers is not the same as functional similarity, it can be intuitively

viewed as another approximation to the gene tree. The hierarchical clustering result on the 140 genes is robust to the core extraction step, since it is similar to the result on the largest connected component of the initial network (see Appendix E in the supplementary materials).

The community labels and hierarchy returned by the HCD-sign algorithm are given in Figures 7(a) and (b), respectively. The gene communities turn out to be associated with different types of anemia or different processes causing the disease. We use the MSigDB database (Liberzon et al. 2011) to perform enrichment analysis and define enriched gene ontology terms by computing overlaps with known functional gene sets using a hypergeometric distribution, as a way to interpret the communities. Table 3 provides a summary of high-level interpretations.

The left branch of the dendrogram consists of communities 1 and 2. The major biological processes related to these communities are both involved in the activation of cells due to exposure to a factor, leading to the activation of immune cells and hence contributing to an immune response (Liberzon et al. 2011). Indeed, many genes in these communities are associated with “cluster of differentiation molecules,” which are genes coding for cell surface markers that can be used by the immune system to activate cells and communicate, amongst others (Zola et al. 2007).

The right branch of the hierarchical splitting tree consists of four communities resulting from three sequential splits. Community 3 contains several genes coding for interleukins, the signaling molecules that are mainly used by the immune system, and other signaling related genes such as MAP kinases. The most significant gene set for this community, response to cytokine, confirms that these genes are related to molecular signaling within the immune system. Community 4 is well sepa-

Table 3. Most significantly enriched gene sets for each community.

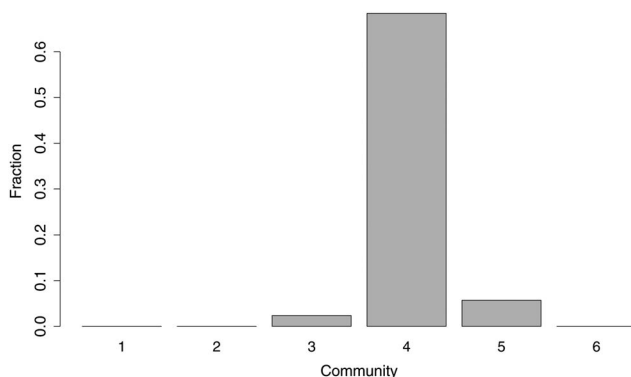
Community	Most significantly enriched gene set
1	Cell activation involved in immune response
2	Cell activation
3	Response to cytokine
4	DNA repair
5	Cytokine mediated signaling pathway
6	Oxygen transport

rated in the network. Nine out of the 16 genes in the community are related to the FANCC protein core complex, a set of proteins that have been linked to fanconi anemia. The fanconi anemia pathway is involved in removing DNA interstrand crosslinks (ICL), a form of DNA repair, during DNA replication and transcription (Ceccaldi, Sarangi, and D'Andrea 2016). Studies have related the mutations in these genes (typically FANCA, FANCC, and FANCG) to bone marrow failure (Schneider et al. 2017). Related, the most significantly enriched gene ontology term of this community is DNA repair. The conjecture that the genes in community 4 are related to fanconi anemia can be verified by checking the keywords in the titles. Specifically, for each community, we define its “signal” papers to be the set of papers that link at least two genes within the same community. Within this set, we found 68% of the signal papers for community 4 have the word “fanconi” in their titles, significantly more than other communities (see Figure 8(a)). The most significantly enriched biological process for genes in community 5 is related to cytokine signaling. It contains one of the core genes of the FANCC protein core complex (the gene “FANCC” in Figure 7(a)), which interestingly connects many FANCC genes with the other genes in the network, therefore, acting as a connection between the fanconi anemia community and the remainder of the network. Finally, the genes in community 6 are related to oxygen transport. Note that impaired oxygen transport can be a result of insufficient healthy red blood cells to transport oxygen around the body. Sick-cell anemia, caused by a mutation in the hemoglobin-beta (HBB) gene, results in red blood cells with a sickle-like morphology that impairs their oxygen transport. We hypothesize that the genes in this community are thus related to sickle-cell anemia. Indeed, the sickle-cell anemia causative gene, HBB, is a hub in community 6 (see

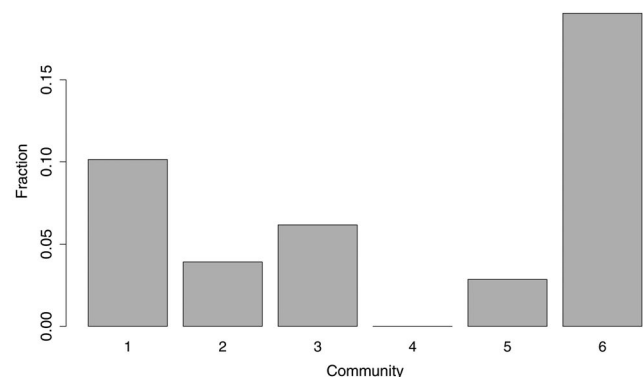
Figure 7(a)). Correspondingly, the fraction of signal papers for community 6 that mention the word “sickle” or “thalassemia” is significantly higher than in other communities (see Figure 8(b)).

For comparison, we also applied regularized spectral clustering to the dataset, with results reported in Appendix E in the supplementary materials. The clustering labels of the RSC match 72% of the HCD labels. However, community sizes from RSC are very unbalanced, and interpretation is more difficult, and of course, there is no hierarchy. A more formal comparison between the BTSBM and the SBM fit to this dataset can be carried out by comparing their ability to predict held-out links. Following the framework of the ECV procedure of Li, Levina, and Zhu (2020), we randomly hold out 10% of the adjacency matrix entries as test data and then fit both models using the remaining 90% as the training data. Then the presence or absence of a link in the held-out entries can be predicted from the fitted probabilities of edges for those entries. A standard way to quantify link prediction performance is the area under the ROC curve (AUC) (Wu, Levina, and Zhu 2018). Over 100 replications, we obtain the average predictive AUC of 0.829 for the BTSBM; the SBM performs noticeably worse, with an average predictive AUC of 0.776. This advantage suggests a hierarchical representation fits this network better, which matches the intuitive analogy about the gene ontology tree. The details are included in Appendix E in the supplementary materials.

To further explore the biological relevance of the hierarchical results, we compared the functional similarity between genes in communities at different levels of the tree. Functional similarity between genes can be measured by their semantic similarity along the gene ontology tree (Ashburner et al. 2000). We used the TopoICSim algorithm (Ehsani and Drablos 2016) to quantify gene-gene similarities between genes based on their “biological process” functional annotation (Ashburner et al. 2000; Barrell et al. 2009). We found that, as expected, communities at the first (top) level of the tree have a lower semantic similarity (and hence more distant gene functions) than communities that split up further down the tree, which provides another indirect piece of evidence of biological relevance of the hierarchy (Appendix Figure E.2(a) in the supplementary materials). One exception is community 6, which is at the lowest level of the tree and has the lowest semantic similarities. However, nodes in community



(a) Papers with “fanconi” in the title.



(b) Papers with “sickle” or “thalassemia” in the title.

Figure 8. The proportion of papers with certain key words in the title, out of the signal papers for each community (papers that mention at least 2 genes within that community).

6 have a lower expected degree than in any other community, with many nodes having the minimum number of 3 edges (Appendix Figure E.2(a) in the supplementary materials), so this community can be expected to have lower average similarities between nodes.

6. Discussion

We studied recursive partitioning as a framework for hierarchical community detection and proposed two specific algorithms for implementing it, using either spectral clustering or sign splitting. This framework requires a stopping rule to decide when to stop splitting communities, but otherwise is tuning-free. We have shown that in certain regimes recursive partitioning has significant advantages in computational efficiency, community detection accuracy, and hierarchical structure recovery, compared with K -way partitioning. An important feature of hierarchical splitting is that it can recover high-level mega-communities correctly even when all K smaller communities cannot be recovered. It also provides a natural interpretable representation of the community structure, and induces a tree-based similarity measure that does not depend on community label permutations and allows us to quantitatively compare entire hierarchies of communities. The algorithm itself is model-free, but we showed it works under a new model we introduced, the binary tree SBM. Under this model, the hierarchical algorithm based on sign splitting is consistent for estimating both individual communities and the entire hierarchy. We conjecture that the advantage of hierarchical clustering carries over to more general models; more work will be needed to establish this formally.

Supplementary Materials

The online supplementary materials contain our proofs for the theoretical results, additional details of the Anemia example, and one additional example of using HCD to analyze a citation network.

Acknowledgments

We thank the associate editor and the referees for their helpful and constructive comments.

Funding

T. Li was supported in part by an NSF grant (DMS-2015298) and the Quantitative Collaborative Award from the College of Arts and Sciences at the University of Virginia. K. Van den Berge is a postdoctoral fellow of the Belgian American Educational Foundation (BAEF) and is supported by the Research Foundation Flanders (FWO), grant 1246220N. P. Sarkar was supported in part by an NSF grant (DMS-1713082). P. Bickel is supported in part by an NSF grant (DMS-1713083). E. Levina is supported in part by NSF grants (DMS-1521551 and DMS-1916222) and an ONR grant (N000141612910).

References

Abbe, E. (2018), "Community Detection and Stochastic Block Models: Recent Developments," *Journal of Machine Learning Research*, 18, 1–86. [1,6]

Abbe, E., Fan, J., Wang, K., and Zhong, Y. (2017), "Entrywise Eigenvector Analysis of Random Matrices With Low Expected Rank," arXiv no. 1709.09565. [3,6]

Airoldi, E. M., Blei, D. M., Fienberg, S. E., and Xing, E. P. (2008), "Mixed Membership Stochastic Blockmodels," *Journal of Machine Learning Research*, 9, 1981–2014. [1]

Amini, A. A., Chen, A., Bickel, P. J., and Levina, E. (2013), "Pseudo-Likelihood Methods for Community Detection in Large Sparse Networks," *The Annals of Statistics*, 41, 2097–2122. [1,3,9]

Amini, A. A., and Levina, E. (2018), "On Semidefinite Relaxations for the Block Model," *The Annals of Statistics*, 46, 149–179. [1]

Arenas, A., Fernandez, A., and Gomez, S. (2008), "Analysis of the Structure of Complex Networks at Different Resolution Levels," *New Journal of Physics*, 10, 053039. [2]

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000), "Gene Ontology: Tool for the Unification of Biology," *Nature Genetics*, 25, 25–29, DOI: 10.1038/75556. [14,15]

Balakrishnan, S., Xu, M., Krishnamurthy, A., and Singh, A. (2011), "Noise Thresholds for Spectral Clustering," in *Advances in Neural Information Processing Systems* (Vol. 24), eds. J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Curran Associates, Inc., pp. 954–962, available at <http://papers.nips.cc/paper/4342-noise-thresholds-for-spectral-clustering.pdf>. [2,3,5,8]

Barrell, D., Dimmer, E., Huntley, R. P., Binns, D., O'Donovan, C., and Apweiler, R. (2009), "The GOA Database in 2009—An Integrated Gene Ontology Annotation Resource," *Nucleic Acids Research*, 37, D396–D403. [15]

Bickel, P. J., and Chen, A. (2009), "A Nonparametric View of Network Models and Newman–Girvan and Other Modularities," *Proceedings of the National Academy of Sciences of the United States of America*, 106, 21068–21073. [1]

Bickel, P. J., Chen, A., and Levina, E. (2011), "The Method of Moments and Degree Distributions for Network Models," *The Annals of Statistics*, 39, 2280–2301. [1]

Bickel, P. J., Choi, D., Chang, X., and Zhang, H. (2013), "Asymptotic Normality of Maximum Likelihood and Its Variational Approximation for Stochastic Blockmodels," *The Annals of Statistics*, 41, 1922–1943. [1]

Bickel, P. J., and Sarkar, P. (2016), "Hypothesis Testing for Automated Community Detection in Networks," *Journal of the Royal Statistical Society, Series B*, 78, 253–273. [2,4]

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., and Lefebvre, E. (2008), "Fast Unfolding of Communities in Large Networks," *Journal of Statistical Mechanics: Theory and Experiment*, 2008, P10008. [2,9]

Blundell, C., and Teh, Y. W. (2013), "Bayesian Hierarchical Community Discovery," in *Advances in Neural Information Processing Systems*, pp. 1601–1609. [2]

Cai, T. T., and Li, X. (2015), "Robust and Computationally Feasible Community Detection in the Presence of Arbitrary Outlier Nodes," *The Annals of Statistics*, 43, 1027–1059. [1]

Ceccaldi, R., Sarangi, P., and D'Andrea, A. D. (2016), "The Fanconi Anaemia Pathway: New Players and New Functions," *Nature Reviews Molecular Cell Biology*, 17, 337. [15]

Celisse, A., Daudin, J.-J., and Pierre, L. (2012), "Consistency of Maximum-Likelihood and Variational Estimators in the Stochastic Block Model," *Electronic Journal of Statistics*, 6, 1847–1899. [1]

Chatterjee, S. (2015), "Matrix Estimation by Universal Singular Value Thresholding," *The Annals of Statistics*, 43, 177–214. [1,3]

Chen, K., and Lei, J. (2018), "Network Cross-Validation for Determining the Number of Communities in Network Data," *Journal of the American Statistical Association*, 113, 241–251. [1,4]

Chen, Y., Sanghavi, S., and Xu, H. (2012), "Clustering Sparse Graphs," in *Advances in Neural Information Processing Systems*, pp. 2204–2212. [1]

Chen, Y., and Xu, J. (2014), "Statistical-Computational Tradeoffs in Planted Problems and Submatrix Localization With a Growing Number of Clusters and Submatrices," arXiv no. 1402.1267. [1]

- Clauset, A., Moore, C., and Newman, M. E. (2008), "Hierarchical Structure and the Prediction of Missing Links in Networks," *Nature*, 453, 98–101. [2,5]
- Clauset, A., Newman, M. E., and Moore, C. (2004), "Finding Community Structure in Very Large Networks," *Physical Review E*, 70, 066111. [2]
- Dasgupta, A., Hopcroft, J., Kannan, R., and Mitra, P. (2006), "Spectral Clustering by Recursive Partitioning," in *European Symposium on Algorithms*, Springer, pp. 256–267. [2,3,8,9,12]
- De Benoist, B., Cogswell, M., Egli, I., and McLean, E. (2008), "Worldwide Prevalence of Anaemia 1993–2005; Who Global Database of Anaemia." [14]
- Ehsani, R., and Drabløs, F. (2016), "Topoisim: A New Semantic Similarity Measure Based on Gene Ontology," *BMC Bioinformatics*, 17, 296. [15]
- Eldridge, J., Belkin, M., and Wang, Y. (2017), "Unperturbed: Spectral Analysis Beyond Davis-Kahan," arXiv no. 1706.06516. [6]
- Fortunato, S. (2010), "Community Detection in Graphs," *Physics Reports*, 486, 75–174. [1]
- Gao, C., and Lafferty, J. (2017), "Testing for Global Network Structure Using Small Subgraph Statistics," arXiv no. 1710.00862. [2,4]
- Gao, C., Lu, Y., Ma, Z., and Zhou, H. H. (2016), "Optimal Estimation and Completion of Matrices With Biclustering Structures," *Journal of Machine Learning Research*, 17, 1–29. [1]
- Gao, C., Ma, Z., Zhang, A. Y., and Zhou, H. H. (2017), "Achieving Optimal Misclassification Proportion in Stochastic Block Models," *The Journal of Machine Learning Research*, 18, 1980–2024. [1,3]
- Girvan, M., and Newman, M. E. (2002), "Community Structure in Social and Biological Networks," *Proceedings of the National Academy of Sciences of the United States of America*, 99, 7821–7826. [2]
- Gleiser, P. M., and Danon, L. (2003), "Community Structure in Jazz," *Advances in Complex Systems*, 6, 565–573. [2]
- Goldenberg, A., Zheng, A. X., Fienberg, S. E., and Airoldi, E. M. (2010), "A Survey of Statistical Network Models," *Foundations and Trends in Machine Learning*, 2, 129–233. [1]
- Handcock, M. S., Raftery, A. E., and Tantrum, J. M. (2007), "Model-Based Clustering for Social Networks," *Journal of the Royal Statistical Society, Series A*, 170, 301–354. [1]
- Hoff, P. (2008), "Modeling Homophily and Stochastic Equivalence in Symmetric Relational Data," in *Advances in Neural Information Processing Systems*, pp. 657–664. [1]
- Holland, P. W., Laskey, K. B., and Leinhardt, S. (1983), "Stochastic Block-models: First Steps," *Social Networks*, 5, 109–137. [1]
- Holme, P., Huss, M., and Jeong, H. (2003), "Subnetwork Hierarchies of Biochemical Pathways," *Bioinformatics*, 19, 532–538. [2]
- Jin, J., Ke, Z. T., and Luo, S. (2019), "Optimal Adaptivity of Signed-Polygon Statistics for Network Testing," arXiv no. 1904.09532. [2,4]
- Joseph, A., and Yu, B. (2016), "Impact of Regularization on Spectral Clustering," *The Annals of Statistics*, 44, 1765–1791. [1,3]
- Kannan, R., Vempala, S., and Vetta, A. (2004), "On Clusterings: Good, Bad and Spectral," *Journal of the ACM*, 51, 497–515. [2,3,13]
- Karrer, B., and Newman, M. E. (2011), "Stochastic Blockmodels and Community Structure in Networks," *Physical Review E*, 83, 016107. [1]
- Kim, J., Kim, J.-J., and Lee, H. (2017), "An Analysis of Disease-Gene Relationship From Medline Abstracts by DigSee," *Scientific Reports*, 7, 1–13. [14]
- Kleinberg, J. M. (2002), "Small-World Phenomena and the Dynamics of Information," in *Advances in Neural Information Processing Systems*, pp. 431–438. [2]
- Krzakala, F., Moore, C., Mossel, E., Neeman, J., Sly, A., Zdeborová, L., and Zhang, P. (2013), "Spectral Redemption in Clustering Sparse Networks," *Proceedings of the National Academy of Sciences of the United States of America*, 110, 20935–20940. [4]
- Lancichinetti, A., and Fortunato, S. (2009), "Community Detection Algorithms: A Comparative Analysis," *Physical Review E*, 80, 056117. [9]
- Larsen, R. M. (1998), "Lanczos Bidiagonalization With Partial Reorthogonalization," DAIMI Report Series 27. [8]
- Le, C. M., and Levina, E. (2015), "Estimating the Number of Communities in Networks by Spectral Methods," arXiv no. 1507.00827. [1,4,7,9]
- Le, C. M., Levina, E., and Vershynin, R. (2017), "Concentration and Regularization of Random Graphs," *Random Structures & Algorithms*, 51, 538–561. [1,3]
- Lei, J., and Rinaldo, A. (2014), "Consistency of Spectral Clustering in Stochastic Block Models," *The Annals of Statistics*, 43, 215–237. [1]
- Li, T., Levina, E., and Zhu, J. (2020), "Network Cross-Validation by Edge Sampling," *Biometrika*, 107, 257–276. [1,4,15]
- Liberzon, A., Subramanian, A., Pinchback, R., Thorvaldsdóttir, H., Tamayo, P., and Mesirov, J. P. (2011), "Molecular Signatures Database (MSigDB) 3.0," *Bioinformatics*, 27, 1739–1740. [14]
- Lloyd, S. (1982), "Least Squares Quantization in PCM," *IEEE Transactions on Information Theory*, 28, 129–137. [8]
- Lyzinski, V., Tang, M., Athreya, A., Park, Y., and Priebe, C. E. (2017), "Community Detection and Classification in Hierarchical Stochastic Blockmodels," *IEEE Transactions on Network Science and Engineering*, 4, 13–26. [2,7,8]
- Mariadassou, M., Robin, S., and Vacher, C. (2010), "Uncovering Latent Structure in Valued Graphs: A Variational Approach," *The Annals of Applied Statistics*, 4, 715–742. [1]
- Matias, C., and Miele, V. (2017), "Statistical Clustering of Temporal Networks Through a Dynamic Stochastic Block Model," *Journal of the Royal Statistical Society, Series B*, 79, 1119–1141. [1]
- Mistry, M., and Pavlidis, P. (2008), "Gene Ontology Term Overlap as a Measure of Gene Functional Similarity," *BMC Bioinformatics*, 9, 327. [14]
- Newman, M. E. (2006), "Modularity and Community Structure in Networks," *Proceedings of the National Academy of Sciences of the United States of America*, 103, 8577–8582. [1]
- (2010), *Networks: An Introduction*, Oxford: Oxford University Press. [1]
- Newman, M. E., and Girvan, M. (2004), "Finding and Evaluating Community Structure in Networks," *Physical Review E*, 69, 026113. [1,2]
- Olhede, S. C., and Wolfe, P. J. (2014), "Network Histograms and Universality of Blockmodel Approximation," *Proceedings of the National Academy of Sciences of the United States of America*, 111, 14722–14727. [2]
- Peel, L., and Clauset, A. (2015), "Detecting Change Points in the Large-Scale Structure of Evolving Networks," in *AAAI*, pp. 2914–2920. [2,5]
- Pons, P., and Latapy, M. (2005), "Computing Communities in Large Networks Using Random Walks," in *International Symposium on Computer and Information Sciences*, Springer, pp. 284–293. [2]
- Radicchi, F., Castellano, C., Cecconi, F., Loreto, V., and Parisi, D. (2004), "Defining and Identifying Communities in Networks," *Proceedings of the National Academy of Sciences of the United States of America*, 101, 2658–2663. [2]
- Reichardt, J., and Bornholdt, S. (2006), "Statistical Mechanics of Community Detection," *Physical Review E*, 74, 016110. [2]
- Rohe, K., Chatterjee, S., and Yu, B. (2011), "Spectral Clustering and the High-Dimensional Stochastic Blockmodel," *The Annals of Statistics*, 39, 1878–1915. [1]
- Schneider, R. K., Mullally, A., Dugourd, A., Peisker, F., Hoogenboezem, R., Van Strien, P. M., Bindels, E. M., Heckl, D., Büsche, G., and Fleck, D. (2017), "Gli1+ Mesenchymal Stromal Cells Are a Key Driver of Bone Marrow Fibrosis and an Important Cellular Therapeutic Target," *Cell Stem Cell*, 20, 785–800. [15]
- Shi, J., and Malik, J. (2000), "Normalized Cuts and Image Segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22, 888–905. [2,13]
- Spielman, D. A., and Teng, S.-H. (1996), "Spectral Partitioning Works: Planar Graphs and Finite Element Meshes," in *37th Annual Symposium on Foundations of Computer Science. Proceedings, IEEE*, pp. 96–105. [2]
- Van Der Hofstad, R. (2016), *Random Graphs and Complex Networks*, New York: Cambridge University Press. [3]
- Wakita, K., and Tsurumi, T. (2007), "Finding Community Structure in Mega-Scale Social Networks," in *Proceedings of the 16th International Conference on World Wide Web*, pp. 1275–1276. [2]
- Wang, S., and Rohe, K. (2016), "Discussion of 'Coauthorship and Citation Networks for Statisticians,'" *The Annals of Applied Statistics*, 10, 1820–1826. [14]
- Wang, Y. R., and Bickel, P. J. (2017), "Likelihood-Based Model Selection for Stochastic Block Models," *The Annals of Statistics*, 45, 500–528. [1,4]
- Ward, J. H., Jr. (1963), "Hierarchical Grouping to Optimize an Objective Function," *Journal of the American Statistical Association*, 58, 236–244. [2]

- Wilkinson, D., and Huberman, B. A. (2002), “Finding Communities of Related Genes,” arXiv no. cond-mat/0210147. [2]
- Wu, Y.-J., Levina, E., and Zhu, J. (2018), “Link Prediction for Egocentrically Sampled Networks,” arXiv no. 1803.04084. [15]
- Xu, K. S., and Hero, A. O. (2013), “Dynamic Stochastic Blockmodels: Statistical Models for Time-Evolving Networks,” in *International Conference on Social Computing, Behavioral-Cultural Modeling, and Prediction*, Springer, pp. 201–210. [1]
- Yang, Z., Algesheimer, R., and Tessone, C. J. (2016), “A Comparative Analysis of Community Detection Algorithms on Artificial Networks,” *Scientific Reports*, 6, 30750. [9]
- Yao, Y. (2003), “Information-Theoretic Measures for Knowledge Discovery and Data Mining,” in *Entropy Measures, Maximum Entropy Principle and Emerging Applications*, Springer, pp. 115–136. [9]
- Young, S. J., and Scheinerman, E. R. (2007), “Random Dot Product Graph Models for Social Networks,” in *International Workshop on Algorithms and Models for the Web-Graph*, Springer, pp. 138–149. [2]
- Zhang, Y., Levina, E., and Zhu, J. (2014), “Detecting Overlapping Communities in Networks Using Spectral Methods,” arXiv no. 1412.3432. [1]
- Zhao, Y., Levina, E., and Zhu, J. (2012), “Consistency of Community Detection in Networks Under Degree-Corrected Stochastic Block Models,” *The Annals of Statistics*, 40, 2266–2292. [1]
- Zola, H., Swart, B., Banham, A., Barry, S., Beare, A., Bensussan, A., Boumsell, L., Buckley, C. D., Bühring, H.-J., Clark, G., and Engel, P. (2007), “CD Molecules 2006—Human Cell Differentiation Molecules,” *Journal of Immunological Methods*, 319, 1–5. [14]