Recovering Locally Distinguishable Communities with Applications to Clustering Training Data

Clarence Worrell[*] Jourdain Lamperski[†]

Department of Industrial Engineering, University of Pittsburgh

January 7, 2024

Abstract

We consider the problem of recovering the ground-truth communities of a vertex-attributed graph that are *locally distinguishable* in the sense that the ground-truth communities exhibit stronger community structure in more local parts of the graph (i.e., in subgraphs induced by vertices whose attributes are closer). As we demonstrate, the problem arises when clustering training data that contains responses generated from different latent sources. More specifically, we show that the ground-truth communities (corresponding to the latent sources) of certain *similarity graphs* for the training data are locally distinguishable under mild conditions. Locally distinguishable communities, however, are not necessarily *globally distinguishable* (i.e. have higher internal than external connection density), posing a challenge for standard community detection methods. Accordingly, we propose a community detection algorithm that first clusters the vertex set into *local* vertex subsets, then applies a spectral community detection algorithm to each graph induced by one of the local vertex subsets to obtain *local* communities, and finally agglomerates the local communities into *global* communities. We establish conditions under which the algorithm *almost exactly* recovers communities when applied to the *local stochastic block model* (LSBM), a generalization of the stochastic block model (SBM) that we propose that incorporates the local distinguishability property. We also evaluate the empirical performance of the method on LSBM instances and its viability for clustering various real training datasets.

1 Introduction

Community detection (or *graph clustering*) involves clustering the vertices of a graph into vertex subsets called *communities*. A community is usually thought of as a subset of vertices that are more densely connected to each other than to vertices that are not in the subset. The amount that the communities are more densely internally than externally connected

[*]Email: clw117@pitt.edu.

[†]Corresponding author. Email: lamperski@pitt.edu.

captures the extent to which the communities are *distinguishable* (or the strength of the community structure). Typically the goal in community detection problems is to either recover ground-truth communities or explore natural groupings of the vertices.

Data clustering problems are often recast as community detection problems through the notion of a *similarity graph*. A similarity graph is a graph whose vertices represent data points and whose edges indicate whether or not data points are "similar." (The edges may also have weights that capture the amount of similarity.) Recasting data clustering problems as community detection problems is particularly effective when it is of interest to find clusters of data points that are spread across space (e.g., when ground-truth communities are known to admit such structure).

In this work we consider a community detection problem in which the goal is to recover ground-truth communities of a vertex-attributed graph that are *locally distinguishable* in the sense that the ground-truth communities exhibit stronger community structure in more local parts of the graph (i.e., in subgraphs induced by vertices whose attributes are closer). A more precise (but still somewhat informal) description of the local distinguishability property that we consider is as follows.

The local distinguishability property. Communities are locally distinguishable if the following two conditions hold:

(i) The amount that the likelihood that there is an edge between two vertices in the same community exceeds the likelihood that there is an edge between two vertices in different communities (with the same attributes as the two vertices in the same community) increases as the distance (under some metric) between the attributes of the vertices decreases.

(ii) The likelihood that there is an edge between two vertices in the same ground-truth community (different ground-truth communities) increases (decreases) as the distance between the attributes of the vertices decreases.

Condition (i) alone captures the fact that locally distinguishable communities exhibit stronger community structure in more local parts of the graph. Condition (ii) is an additional condition that we enforce that is not necessarily implied by condition (i).

As we demonstrate in Subsection 2.1, the recovery problem of interest naturally arises when clustering training data that contains responses generated from different latent sources. More specifically, we show that the ground-truth communities (corresponding to the latent sources) of certain similarity graphs for the training data (whose edges capture whether or not the responses of training data points are similar) satisfy the local distinguishability property under mild conditions.

Locally distinguishable communities, however, may not be *globally distinguishable* (i.e., have higher internal than external connection density). Accordingly, the recovery problem poses a challenge for community detection algorithms that are designed based on standard notions of a community. In this work we design an algorithm specifically for the recovery problem. The algorithm first clusters the vertex set into *local vertex subsets*, then applies a standard (spectral) community detection algorithm to each graph induced by one of the local vertex subsets to obtain *local communities*, and finally agglomerates the local communities into global communities. We present a schematic of the algorithm along with a more detailed discussion about the algorithm in Subsection 2.2.

Our goal in this work is twofold. First, we aim to establish conditions under which our algorithm theoretically recovers when applied to the *local stochastic block model* (LSBM), a variation of the stochastic block model that incorporates the local distinguishability property. We direct the reader to Section 3 for a formal description of the LSBM and Subsection 3.1 for a summary of the main theoretical recovery guarantee that we develop for the algorithm. Second, we aim to evaluate the empirical performance of the method on LSBM instances and its viability for clustering real training data. To this end, we present a computational study in Section 8.

1.1 Organization

First in Section 2 we provide motivation for this work; specifically, we discuss the motivating application of clustering training data, present a schematic of the community detection algorithm that we propose, and discuss related work. Next we present a description of the LSBM in Section 3 along with the main recovery guarantee that we develop for the community detection algorithm. In Section 4 we introduce and study the algorithm that we will use to construct local vertex subsets. Next we present and analyze a spectral community detection algorithm for partitioning the local vertex subsets into local communities in Section 5. Then in Section 6 we introduce and analyze the algorithm that we will use to agglomerate the local communities. In Section 7 we present a full description of the community detection algorithm that we propose along with a formal recovery guarantee. Finally we present a computational study in Section 8, and we conclude with discussion and directions for future research in Section 9.

2 Motivation and related work

In Subsection 2.1 we discuss the motivating application of clustering training data. Next we provide a schematic of the community detection algorithm that we propose in Subsection 2.2. Finally in Subsection 2.3 we discuss related work.

2.1 Clustering training data

Consider training data $D = \{(x_i, y_i)\}_{i\in[n]} \subset \mathbb{R}^m \times \mathbb{R}$, where $[n] := \{1, \dots, n\}$, that contains n feature-response pairs. Let $f_1, f_2 : \mathbb{R}^m \to \mathbb{R}$ be functions that we will refer to as *sources*. Suppose that each response is generated from one of the two sources: for each $i \in [n]$, either $y_i = f_1(x_i) + \epsilon_i$ or $y_i = f_2(x_i) + \epsilon_i$, where ϵ_i is random noise. (Later in this section we will place formal assumptions on the random noise.) Further suppose that the sources are latent, meaning the source of each response is unknown.

Example 2.1. As a running example, let us consider the cars dataset from the CMU Statlib library [1]. (We actually use the slightly revised version of the dataset from the UC Irvine Machine Learning Repository [21].) The dataset contains feature information (e.g., miles per gallon, car weight, and model year) for 406 different cars. Each car has a model year between 1970 and 1982. We restrict our attention to *newer* cars that have a model year in 1979-1982 and *older* cars that have a model year in 1970-1973. This reduces the dataset to $n = 243$ data points. Take x_i to be the weight of the i-th car (i.e., there is $m = 1$ feature), and take y_i to be miles per gallon (mpg) of the i-th car. Also take $f_1(x_i)$ to be the mpg of a newer car of weight x_i, and take $f_2(x_i)$ to be the mpg of an older car of weight x_i. (The sources are known in this case, but we will pretend that they are not.) See the first plot of Figure 1 for a scatter plot of the data. Observe that a newer car that has the same weight as an older car (naturally) tends to have higher mpg. The second plot in Figure 1 depicts overlaid smoothed estimates of the mean mpg (along with 95% confidence intervals) of newer and older cars as a function of their weight. We can think of these estimated functions as the sources f_1 and f_2, respectively. $\square$

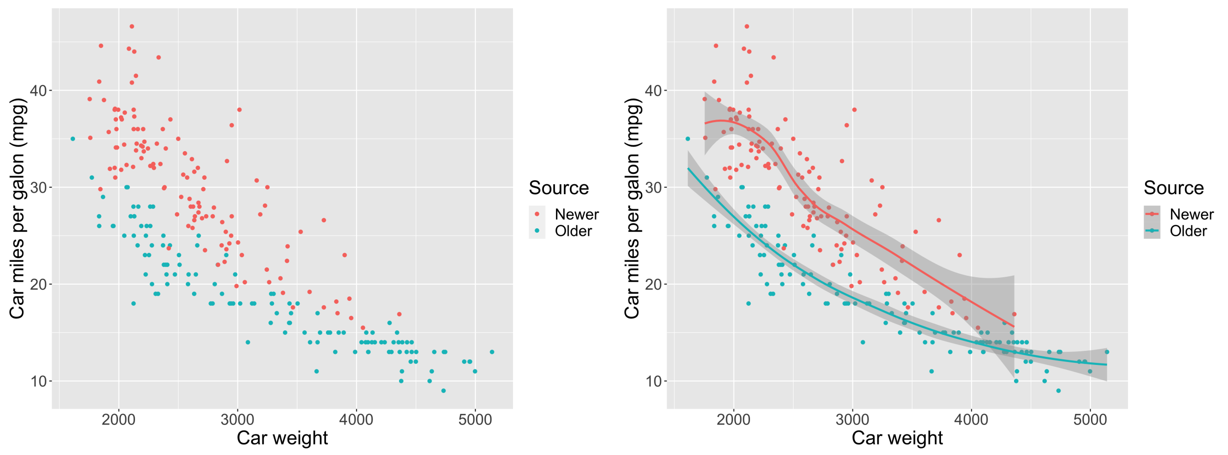

Figure 1: The car training data along with another plot of overlaid smoothed estimates of the mean mpg (along with 95% confidence intervals) of newer and older cars as a function of their weight.

Consider clustering the training data D; there are two key applications of this task. The first is training *clusterwise* models, and the second is labeling unlabeled data for training a classification model. We direct the reader to Subsection 2.3 for a thorough discussion of these applications and related work.

Next we demonstrate that the ground-truth communities (corresponding to the sources) of certain similarity graphs for the training data are locally distinguishable under the following assumptions. The most strict assumption is that the sources are *separable*:

(i) For $\tau > 0$, it holds that $\inf_{x \in \mathbb{R}^n} f_1(x) - f_2(x) \geq \tau$. We say that the sources are τ-*separable*. In words, the first source outputs higher responses than the second source. Note that this condition is satisfied in Example 2.1.

The other assumptions are rather mild in comparison; we assume that the sources are *Lipshitz* and that the random noise terms are independent and identically distributed mean 0 normal random variables:

(ii) For $L > 0$, the sources are L-*Lipshitz*. That is, $|f_1(x_1) - f_1(x_2)| \leq L\|x_1 - x_2\|_2$ and $|f_2(x_1) - f_2(x_2)| \leq L\|x_1 - x_2\|_2$ for all $x_1, x_2 \in \mathbb{R}^n$.

(iii) The noise terms ϵ_i, $i \in [n]$ are independent and identically distributed mean 0 normal random variables.

A similarity graph G for D. We introduce the following vertex-attributed similarity graph G for the training data D. The vertex set of G is $V = [n]$, the attributes of vertex $i \in V$ are x_i (the features for the i-th data point), and there is an edge between vertices $i \neq j \in [n]$ if $|y_i - y_j| \leq \delta$, where δ is a parameter. (We discuss how to choose/tune δ below.) That is, there is an edge between vertices if the responses in the data points that they represent are close. We say that G is the *δ-similarity graph for D*. We also write $G = (V, E, x)$, where E is the edge set of G, and the columns of $x \in \mathbb{R}^{m \times n}$ are the vertex attributes (i.e., the i-th column of x is x_i).

Let $\delta > 0$, $G = (V, E, x)$ be the δ-similarity graph for D, and $i \neq j \in V$. Define the function $\phi_\delta : \mathbb{R} \to [0, 1]$ by $\phi_\delta(\gamma) := \mathbb{P}(|\epsilon_i - \epsilon_j + \gamma| \leq \delta)$ for $\gamma \in \mathbb{R}$. Note that ϕ_δ is decreasing on $\mathbb{R}_{\geq 0}$ because the random variable $\epsilon_i - \epsilon_j$ follows a mean 0 normal distribution. Also let $p_{ij} := \mathbb{P}(|y_i - y_j| \leq \delta)$ denote the probability (with respect to ϵ_i and ϵ_j) that there is an edge between vertices i and j.

Consider Proposition 2.1 below, which lower bounds (upper bounds) p_{ij} when the vertices i and j are in the same (different) ground-truth communities. The proposition readily follows from the three assumptions above.

Proposition 2.1. *Suppose that assumptions (i)-(iii) above hold. Let $\delta > 0$, G be the δ-similarity graph for D, and $i \neq j \in V$. If vertices i and j are in the same ground-truth community,*

$$p_{ij} \geq \phi_\delta(L\|x_i - x_j\|_2). \tag{1}$$

Otherwise, if the vertices are in different ground-truth communities,

$$p_{ij} \leq \phi_\delta(\tau - L\|x_i - x_j\|_2). \tag{2}$$

Proof. Suppose that vertices i, j are in the same ground-truth community. Furthermore, without loss of generality, suppose that $y_i = f_1(x_i) + \epsilon_i$ and $y_j = f_1(x_j) + \epsilon_j$. Then

$$\begin{aligned} p_{ij} &= \mathbb{P}(|y_i - y_j| \leq \delta) \\ &= \mathbb{P}(|\epsilon_i - \epsilon_j + f_1(x_i) - f_1(x_j)| \leq \delta) \\ &= \mathbb{P}(|\epsilon_i - \epsilon_j + |f_1(x_i) - f_1(x_j)|| \leq \delta) \\ &= \phi_\delta(|f_1(x_i) - f_1(x_j)|) \\ &\geq \phi_\delta(L\|x_i - x_j\|_2), \end{aligned}$$

the third equality follows from the fact that $\epsilon_i - \epsilon_j$ follows a mean 0 normal distribution, the third equality from the definition of ϕ_δ, and the inequality from the fact that f_1 is L-Lipschitz and ϕ_δ is decreasing.

Now suppose that vertices i, j are in different ground-truth communities. Furthermore, without loss of generality, suppose that $y_i = f_1(x_i) + \epsilon_i$ and $y_j = f_2(x_j) + \epsilon_j$. From a similar argument as above,

$$p_{ij} = \phi_\delta(|f_1(x_i) - f_2(x_j)|)$$

$$\begin{aligned} &= \phi_\delta(|f_1(x_i) - f_2(x_i) + f_2(x_i) - f_2(x_j)|) \\ &\leq \phi_\delta(|f_1(x_i) - f_2(x_i)| - |f_2(x_i) - f_2(x_j)|) \\ &\leq \phi_\delta(\tau - L\|x_i - x_j\|_2) \end{aligned}$$

where the first inequality follows from the reverse triangle inequality and the fact that ϕ_δ is decreasing, and the second inequality follows from the fact the sources are τ-separable, L-Lipschitz, and ϕ_δ is decreasing. □

We are now setup to verify the two local distinguishability conditions (see Section 1); first we verify the second condition. Recalling that ϕ_δ is decreasing, inequality (1) tells us that the probability that there is an edge between vertices i in j in the same ground-truth community is larger (or at least guaranteed to be) if the distance $\|x_i - x_j\|_2$ between their attributes is smaller. Similarly, assuming that $\|x_i - x_j\|_2 \leq \tau/L$, inequality (2) tells us that the probability that there is an edge between vertices i in j in the different ground-truth communities is smaller (or at least guaranteed to be) if the distance $\|x_i - x_j\|_2$ between the attributes is smaller. Thus the second local distinguishability condition holds (assuming that the attributes are sufficiently close). Now let us verify the first local distinguishability condition. Suppose that vertices i and j are in the same ground-truth community, vertices $u \neq v \in V$ are in different ground-truth communities, and $\|x_i - x_j\|_2 = \|x_u - x_v\|_2$. Further suppose that $\|x_i - x_j\|_2 \leq \tau/(2L)$. From (1) and (2),

$$p_{ij} - p_{uv} \geq \phi_\delta(L\|x_i - x_j\|_2) - \phi_\delta(\tau - L\|x_i - x_j\|_2) \geq \phi(\tau/2) - \phi(\tau/2) = 0.$$

Furthermore, the function $h : [0, \tau/(2L)] \to \mathbb{R}_{\geq 0}$ defined by $h(r) = \phi_\delta(Lr) - \phi_\delta(\tau - Lr)$ is decreasing because its derivative $h'(r) = L(\phi'_\delta(Lr) + \phi'_\delta(\tau - Lr))$ is negative (as ϕ_δ is decreasing). It follows that $p_{ij} - p_{uv}$ is larger (or at least guaranteed to be) if the distance between the attributes $\|x_u - x_v\|_2 = \|x_i - x_j\|_2$ is smaller. Thus, the first local distinguishability property holds (assuming that the attributes are sufficiently close).

In light of this discussion, we revisit Example 2.1:

Example 2.2. Let us consider the 2-similarity graph for the training car data. We report the proportion of edges between vertices in the same ground-truth community and different ground truth communities at different distances between the attributes (car weight in this case) in the first plot of Figure 2. Observe that both local distinguishability conditions hold as long as the distance between attributes is sufficiently small (specifically, the difference between car weights is less than 500), resonating with the above argument.

Now consider applying the standard spectral community detection method (that uses the signs of the entries of eigenvector for the second largest eigenvalue of the adjacency matrix) to the 2-similarity graph. We report the communities that the method outputs in the second plot of Figure 2. Notice that the communities do not agree much with the ground-truth communities; this is not surprising in light of the fact that locally distinguishable ground-truth communities are not necessarily globally distinguishable.

Tuning the parameter δ in a semi-supervised setup. While our argument for local distinguishability does not rely on a particular specification of the parameter δ, the specification could impact the performance of a community detection algorithm applied to the δ-similarity

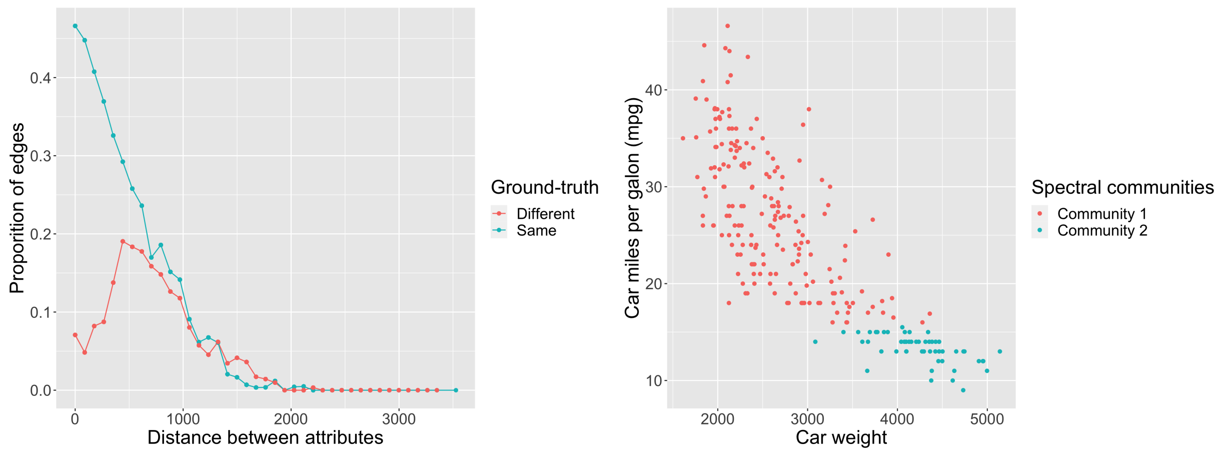

Figure 2: The proportion of edges between vertices in the same ground-truth community and different ground truth communities of the 2-similarity graph at different distances between attributes (car weight in this case), and the communities output by a standard spectral method applied to the adjacency matrix of the 2-similarity graph.

graph. One option is to choose δ to produce the most distinguishable *local* communities under some pre-defined measure (e.g., the second largest eigenvalue). Another option is to use labels $\{z_i\}_{i \in I} \subseteq \{-1, +1\}$ for a subset of the data points (index by $I \subseteq [n]$) if such labels are available. We explore this direction in our computational experiments in Section 8, and we discuss connections with the semi-supervised learning literature in Subsection 2.3.

We conclude by noting that we will primarily focus on the community detection problem of interest for the remainder of this work, for the most part forgetting the underlying motivating application of clustering training data. An exception, however, is the computational study in Section 8, in which we use the community detection algorithm that we develop to cluster similarity graphs for training data that we introduce here.

2.2 Schematic of community detection algorithm

We present a schematic of our community detection algorithm in Algorithm 1. Note that the algorithm takes a metric $d : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}_{\geq 0}$ as input. As we discuss in more detail below, the metric will be used to cluster the attributes to construct local vertex subsets. Also note that the algorithm takes parameters (k, r) as input. We discuss these parameters in more detail below as well, along with Steps 1-3 of the algorithm. We present a more formal description of the steps as separate algorithms in Section 4, 5, and 6, respectively, and we present a more formal description of the community detection algorithm as a whole in Section 7.

Step 1: Local vertex subset construction. A *local vertex subset with respect to x* is a subset $V' \subseteq V$ of the form $V' = \{i \in V : d(x_i, c) \leq r\}$ for some $c \in X$ and $r > 0$. We refer to c and r as the *center* and *local radius* of V', respectively.

Step 1 of Algorithm 1 constructs k local vertex subsets $V^{(i)} \subseteq V$, $i \in [k]$ with respect to the input vertex attributes x. The number of local vertex subsets k is a parameter specified

Algorithm 1 Schematic of community detection algorithm.

```
Input: Vertex-attributed graph $G = (V, E, x)$, metric $d : \mathbb{R}^m \times \mathbb{R}^m \to \mathbb{R}_{\geq 0}$, and parameters $(k, r)$
1: Construct $k$ local vertex subsets $V^{(i)} \subseteq V$, $i \in [k]$
2: Partition each local vertex subset $V^{(i)}$ into local communities $C_1^{(i)}$ and $C_2^{(i)} = V^{(i)} \setminus C_1^{(i)}$
3: Agglomerate the local communities into global communities $C_1$ and $C_2 = V \setminus C_1$
4: Return the global communities $C_1$ and $C_2$
```

by the user. We will establish values of k that ensure theoretical recovery performance and discuss how to tune k in practice. To construct the local vertex subsets, we apply a k-center clustering algorithm to the vertex attributes; see Algorithm 2 in Section 4. The algorithm computes k centers c_i, $i \in [k]$ and then uses these centers to construct the local vertex subsets

$$V^{(i)} := \{j \in V : d(x_j, c_i) \leq r\}, \quad i \in [k],$$

where the local radius r is another parameter specified by the user. Like k, we will establish values of r that ensure theoretical recovery performance and discuss how to tune r in practice. We note that r must be chosen large enough that each vertex is in at least one of the local vertex subsets. Also, some vertices might belong to multiple local vertex subsets (so $V^{(i)}, i \in [k]$ will not necessarily partition V).

Step 2: Local community detection. Step 2 of Algorithm 1 partitions each local vertex subset $V^{(i)}$ into *local communities* $C_1^{(i)}$ and $C_2^{(i)} = V^{(i)} \setminus C_1^{(i)}$, which should be thought of as estimates of the *local ground-truth communities* (i.e., the intersection of the ground-truth communities and $V^{(i)}$). To obtain $C_1^{(i)}$ and $C_2^{(i)}$, we apply a spectral algorithm to (a recentered version of) the adjacency matrix $A^{(i)}$ of the subgraph $G[V^{(i)}]$ in G induced by $V^{(i)}$; see Algorithm 3 in Section 5. That is, $A^{(i)}$ is the principal submatrix of the adjacency matrix A of G that contains the columns and rows of A indexed by $V^{(i)}$. We will refer to $A^{(i)}$ as the *local adjacency matrix* of $V^{(i)}$.

We remark that existing analysis for spectral methods applied to the SBM does not readily lend itself to the setting of interest. Existing analysis relies on closed-form formula for the eigenvectors and eigenvalues of expected value of the adjacency matrix of the SBM that do not hold in the setting of interest. To circumvent this, we will establish spectral properties of the expected value of local adjacency matrices.

Step 3: Agglomeration of local communities. Step 3 of Algorithm 1 agglomerates the local communities into global communities C_1 and C_2. For each $i \in [k]$, define the local community pair $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$. In each iteration of our agglomeration method (Algorithm 4 in Section 6), we agglomerate two community pairs together using *vertex agglomeration.* The first iteration of the vertex agglomeration method proceeds as follows (the remaining iterations proceed similarly). We select the indices $i \neq j \in [k]$ that maximize $|V^{(i)} \cap V^{(j)}|$, and then we agglomerate $C^{(i)}$ and $C^{(j)}$ based on how many vertices are shared between these local community pairs.

2.3 Related work

Below we discuss three related bodies of research.

The SBM and community detection. Naturally it is of interest to understand conditions under which community detection algorithms can recover ground-truth communities. Towards this end, a number of works study the theoretical performance of community detection algorithms applied to random graphs models that have built-in community structure. Most of these works consider the SBM (or some variation of it). In the SBM, n vertices are first randomly partitioned into ground-truth communities, and then edges are placed between pairs of vertices in the same community with probability p (the *within probability*) and between pairs of vertices in different communities with probability q (the *between probability*). Ground-truth communities are more distinguishable (or there is stronger community structure) if the difference $p - q$ is larger.

Classic work on the topic establishes lower bounds on the difference $p - q$ that guarantee that an algorithm of interest (numerous algorithms have been considered) *recovers* the ground-truth communities with high probability (i.e., with probability tending to 1 as $n \to \infty$) [5, 6, 8, 9, 10, 13, 22, 18, 25]; we discuss different notions of recovery in Section 3. While classic work lower bounds $p - q$, more recent work considers the case in which p and q are small as well, i.e. when the graph is sparse. On one hand, this is practical consideration, as graphs that arise in practice are often sparse. On the other hand, it is a theoretical consideration, as there is an information-theoretic phase transition in the sparse regime [3]. It is now known that semidefinite programming methods and the standard spectral method *exactly* recover (see Section 3) up to the information-theoretic threshold in the sparse regime [3, 4].

Our work is inherently different in two ways from these developments. First, in the LBSM (see Section 3), the within and between probability are not homogeneous (they are functions of the distance between vertex attributes), so the model is more challenging to study. Specifically, existing work on the SBM relies on analytic formulae that do not extend to our setting. Second, it is not clear that sparse graphs arise in the underlying application of labeling training data. In fact, if the points are densely distributed enough, then one could argue that the graphs are not sparse if the similarity parameter δ (see Subsection 2.1) is of the appropriate order. Based on these considerations, we establish a lower bound on a quantity that is analogous to the difference $p - q$ (see Section 3), rather than additionally considering a sparse regime of some sort.

We note that there are numerous studies (like ours) that consider vertex-attributed variations of the SBM [12, 15, 20, 28, 27, 26, 30]. To the best of our knowledge, the most closely related variation is the geometric block model [15]. In this model, there is an edge between two vertices if the their random attributes are close enough. The setup differs from ours in two ways. First, even if two vertices have the same attributes, it is not guaranteed that there will be an edge between them in our model. Second, we allow our algorithm to use the attributes, specifically a metric that captures the distance between them.

We conclude by directing the reader to the survey [2] for a comprehensive overview of existing work on the SBM.

Clustering training data. There are two key applications of clustering training data. The first application is training *clusterwise* models. When responses are generated from different sources, it might not be possible to adequately fit a model (especially an interpretable model, such as a linear regression model) to the data. To circumvent this, one could first cluster the training data and then fit a model to each cluster of data. The models could then be studied (in which case interpretability is important) or even aggregated for prediction purposes. Most of the work in this area has focused on *clusterwise linear regression*, in which the goal is to construct linear models for each cluster [11]. Our work is different in that it does not make any parametric modeling assumptions. A consequence of this, however, is that our methods are potentially only practical in applications that have low-dimensional feature spaces.

The second application is labeling unlabeled data for training a classification model. The goal in this application is to label the data points according to their source (i.e., cluster the training data) in order to construct a classification model that predicts the source of a new data point. The problem arises, for example, when constructing classification models for predicting whether or not a patient is taking their medication because labeled data (that captures whether or not a patient is taking their medication) is often not readily available or even obtainable [14, 17]. When partially labeled data is available, the problem is a semi-supervised learning problem; below we discuss work in this direction.

Transductive semi-supervised learning. The semi-supervised learning literature primarily considers semi-supervised binary classification problems; see the surveys [7, 24]. The data in these problems is comprised of features $\{x_i\}_{i\in[n]} \subset \mathbb{R}^m$ and labels $\{y_i\}_{i\in I} \subseteq \{-1, +1\}$ for a subset of the features (indexed by $I \subset [n]$). Recall from Subsection 2.1 that the semi-supervised version of clustering training data is a slightly different; specifically, we are given labels $\{z_i\}_{i\in I}$ for a subset of the feature-response data $\{(x_i, y_i)\}_{i\in[n]}$.

Methods for semi-supervised binary classification fall into one of two classes: *inductive* and *transductive*. Inductive methods use the data to directly construct a classification model. Transductive methods first use the data to label the unlabeled features and then use the resulting fully labeled data to construct a classification model. Transductive methods typically involve the following three steps. First they construct a similarity graph for the feature data. A common approach is to add an edge between two vertices if the corresponding features are close enough. After the graph is constructed, they create weights for the edges, using the labels for the subset of the points. Finally they apply a graph clustering method to the weighted graph to obtain a partition and hence labels.

In our approach, we place an edge between vertices if the corresponding responses are close enough. Also, we do not explicitly use the labeled data points (e.g., to construct edge weights), but we use them to tune the similarity parameter δ and the parameters (k, r) of Algorithm 1.

3 The local stochastic block model

Define $V := [n]$, where n is a positive integer. Let $\mathcal{U}$ denote the uniform distribution on $[0, 1]^m$. Also let d_2 denote the Euclidean metric on $\mathbb{R}^m$ (i.e., $d_2(x_1, x_2) = \|x_1 - x_2\|_2$ for $x_1, x_2 \in \mathbb{R}^m$). Note that $\max_{x_1,x_2\in X} d_2(x_1, x_2) \le \sqrt{m}$. Finally let $p, q, \alpha, \beta \in [0, 1]$.

The local stochastic block model (LSBM). A vertex-attributed graph $G = (V, E, x)$ together with ground-truth communities $C_1^* \subseteq V$, $C_2^* = V \backslash C_1^*$ *follow the LSBM under param-*

eters $(n, m, p, q, \alpha, \beta)$ (or we write $(G, C^*) \sim \mathrm{LSBM}(n, m, p, q, \alpha, \beta)$, where $C^* = (C_1^*, C_2^*)$ is the pair of ground-truth communities) if the graph and communities are generated as follows.

(i) $C_1^* = \{i \in V : Z_i = 1\}$, where Z_i, $i \in V$ are independently drawn from the Bernoulli distribution with probability parameter $1/2$. That is, in SBM parlance, we consider a *symmetric* model.

(ii) The vertex attributes x_i, $i \in V$ are independently drawn from $\mathcal{U}$. The attributes x and ground-truth community assignments Z_i, $i \in V$ are independent.

(iii) For each $i \neq j \in V$, an edge is placed between vertices i and j (i.e., $\{i, j\} \in E$) with probability

$$p_{ij} = \begin{cases} p + \alpha\,(1 - d_2(x_i, x_j)/\sqrt{m}) & \text{if } i, j \in C_1^* \text{ or } i, j \in C_2^* \\ q - \beta\,(1 - d_2(x_i, x_j)/\sqrt{m}) & \text{otherwise,} \end{cases} \tag{3}$$

independently of the other distinct vertex pairs.

To ensure that (3) is well-defined, we assume that $p + \alpha \leq 1$ and $0 \leq q - \beta \leq 1$. We also assume that $p \geq q$, which guarantees that *within probability* $p + \alpha(1 - d_2(x_i, x_j)/\sqrt{m})$ is at least as large as the *between probability* $q - \beta(1 - d_2(x_i, x_j)/\sqrt{m})$:

$$p + \alpha(1 - d_2(x_i, x_j)/\sqrt{m}) \geq q - \beta(1 - d_2(x_i, x_j)/\sqrt{m}).$$

Note that if $\alpha = \beta = 0$, then the LSBM is exactly the (symmetric) SBM (with parameters p and q).

The local distinguishability property. Consider Figure 3, which presents plots of the probability p_{ij} as a function of scaled distance $d_2(x_i, x_j)/\sqrt{m}$ for different values of α and β. Assuming $\alpha > 0$, the within probability $p + \alpha(1 - d_2(x_i, x_j)/\sqrt{m})$ decreases as a function of the distance $d_2(x_i, x_j)$. That is, the probability that we add an edge between two vertices in the same ground-truth community decreases as the distance between their attributes grows. Now consider the between probability $q - \beta(1 - d_2(x_i, x_j)/\sqrt{m})$. If $\beta < 0$, then the between probability increases as a function of distance. That is, the probability that we incorrectly add an edge between two vertices in different ground-truth communities increases as distances increases. Lastly, we note that if $\alpha > -\beta$, then the difference of the within probabilities and between probabilities, namely $p - q + (\alpha + \beta)(1 - d_2(x_i, x_j)/\sqrt{m})$, decreases as a function of the distance $d_2(x_i, x_j)$.

Local and global difference. Define the *local difference*

$$D_\ell := p + \alpha - (q - \beta)$$

and the *global difference*

$$D_g := p - q.$$

In the three panels of Figure 3, we have that $D_\ell = 0.2, 0.4, 0.6$ and $D_g = 0.1, 0.1, 0.1$, respectively. Note that $D_\ell \geq D_g$. The local and global difference capture how distinguishable ground-truth communities are at a local and global level. Analogous to results in the SBM

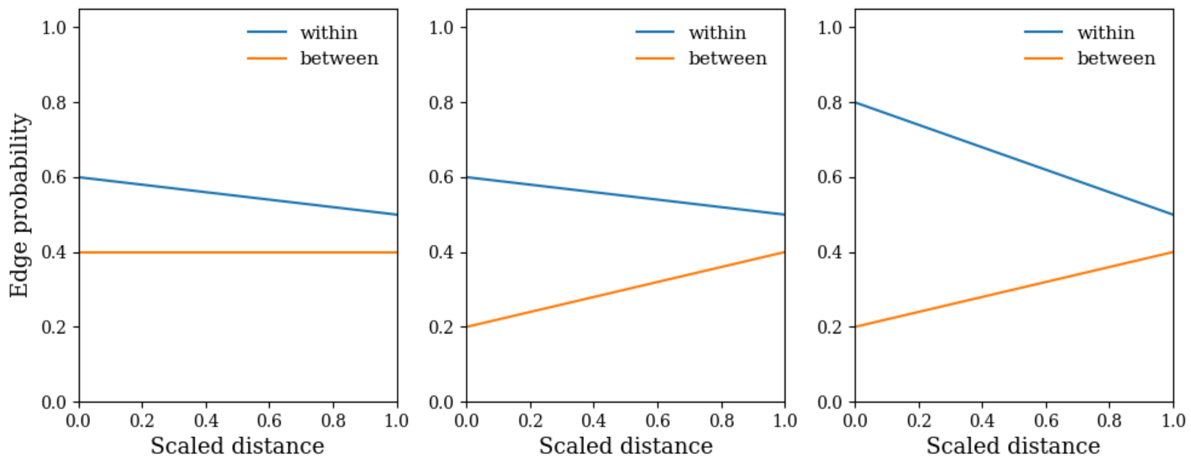

Figure 3: Edge probability p_{ij} as a function of scaled distance $d_2(x_i, x_j)/\sqrt{m}$ for different values of α and β. More specifically, $p = 0.5$ and $q = 0.4$ in all three plots; $(\alpha, \beta) = (0.1, 0), (0.1, 0.2), (0.3, 0.2)$ in the left, center, and right plots, respectively.

literature, we will establish conditions on D_ℓ and D_g under which our algorithm recovers with high probability, bringing us to our next point.

Recovery in the LSBM. Let $\hat{C}_1, \hat{C}_2 \subseteq V$ such that $\hat{C}_1 \cap \hat{C}_2 = \emptyset$ and $\hat{C}_1 \cup \hat{C}_2 \neq \emptyset$. Similarly let $\tilde{C}_1, \tilde{C}_2 \subseteq V$ such that $\tilde{C}_1 \cap \tilde{C}_2 = \emptyset$ and $\tilde{C}_1 \cup \tilde{C}_2 \neq \emptyset$. Define the *agreement* between the pairs $\hat{C} = (\hat{C}_1, \hat{C}_2)$ and $\tilde{C} = (\tilde{C}_1, \tilde{C}_2)$ to be the proportion of the vertices in $\hat{C}_1 \cup \hat{C}_2 \cup \tilde{C}_1 \cup \tilde{C}_2$ that the pairs agree upon:

$$A(C, C^*) := \frac{\max\{|\hat{C}_1 \cap \tilde{C}_1| + |\hat{C}_2 \cap \tilde{C}_2|, |\hat{C}_1 \cap \tilde{C}_2| + |\hat{C}_2 \cap \tilde{C}_1|\}}{|\hat{C}_1 \cup \hat{C}_2 \cup \tilde{C}_1 \cup \tilde{C}_2|}.$$

(We define agreement in a slightly more general way than usual because we will examine the agreement of *local communities* in addition to *global communities*.)

Let $(G, C^*) \sim \text{LSBM}(n, m, p, q, \alpha, \beta)$. A community detection algorithm that takes input $(G, d, p, q, \alpha, \beta)$ and outputs a global community pair $C = (C_1, C_2)$ (i.e., $C_1 \cup C_2 = V$) *exactly recovers* if

$$\mathbb{P}(A(C, C^*) = 1) = 1 - o(1)$$

and *almost exactly recovers* if

$$\mathbb{P}(A(C, C^*) = 1 - o(1)) = 1 - o(1).$$

Throughout our little-o notation is with respect to the number of vertices n. We treat m as a fixed constant, given the underlying motivation of low-dimensional feature/attribute spaces. In this work we will establish conditions on D_ℓ and D_g under which our algorithm almost exactly recovers.

We allow the algorithm to use these parameters (p, q, α, β) to obtain a theoretical guarantee; specifically, the algorithm uses the parameters in the local spectral community detection algorithm to recenter local adjacency matrices (see Section 5). In the computational experiments of (8), we will use an alternative spectral community detection method that does not require the parameters as input (but does not yield a theoretical guarantee, at least as far as we could show). Similarly, the algorithms that we develop run in the general setup of Subsection 2.2, in which the attributes are not necessarily in $[0,1]^m$ and d is any metric, but we only establish theoretical guarantees for them when the attributes belong to $[0,1]^m$ and $d = d_2$.

3.1 Almost exact recovery result

First we set the stage for our results. The standard spectral method (that computes the top eigenvector of a centered version of the adjacency matrix) applied to the SBM almost exactly recovers if $p - q = \Omega(1/\sqrt{n})$, i.e. $(p-q)\sqrt{n} \to \infty$ as $n \to \infty$ [2]. The result follows from the Davis-Kahan theorem and a concentration bound for the operator norm; we will use similar techniques in our analysis of the local spectral community detection method in Section 5. While there are stronger guarantees for the standard spectral method (see the review of related work in Subsection 2.3), they require more involved arguments that we were unable to generalize to the setting of interest.

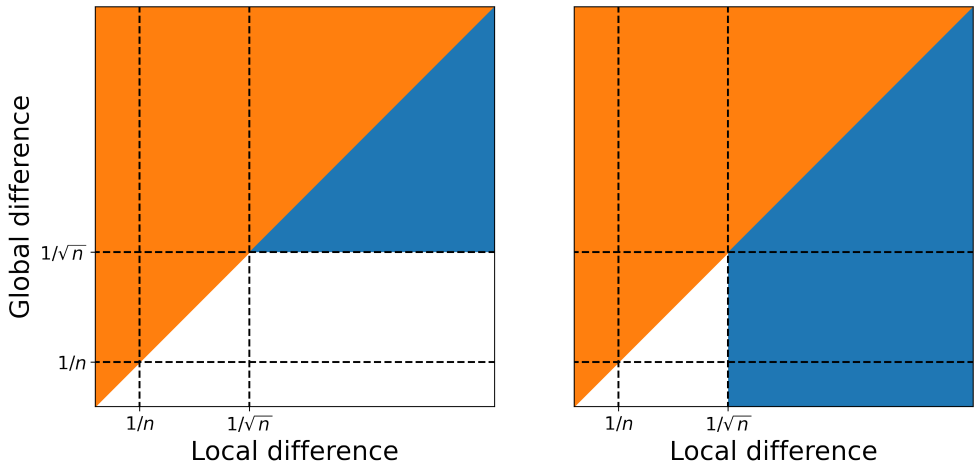

Figure 4: Comparison of almost exact recovery regions (depicted in blue) in terms of the local difference D_ℓ and global difference D_g that we would expect for a standard spectral method and that we establish for our community detection method. The orange area captures the region in which the global difference is larger than the local difference (a regime that we do not consider).

One could guess that applying the standard spectral method to the LSBM would almost exactly recover under the same condition that $D_g = p - q = \Omega(1/\sqrt{n})$ because $p - q$ lower bounds the difference of the within and between probability in the sense that $p - q \leq p - q + (\alpha + \beta)(1 - d_2(x_i, x_j)/\sqrt{m})$ for all $i \neq j \in V$. More strongly, it actually holds that $p - q \leq p_{ij} - p_{uv}$ for $i \neq j$ in the same ground-truth community and $u \neq v$ in different ground-truth communities. We show that we can remove the condition that $D_g = \Omega(1/\sqrt{n})$ by assuming that the local difference D_ℓ is sufficiently large, namely $D_\ell = \Omega(1/\sqrt{n})$:

Theorem 3.1. *If $D_\ell = \Omega(1/\sqrt{n})$, then Algorithm 5 (run with appropriate input) almost exactly recovers.*

Proof. See Section 7. □

In Figure 4 we present an illustrative comparison of the almost exact recovery regions (in terms of the local difference D_ℓ and global difference D_g) that we would expect for a standard spectral method and that we establish for our community detection method.

4 Constructing local vertex subsets

We present the algorithm that we will use to construct local vertex subsets in Algorithm 2. Note that the algorithm takes vertex attributes x as input, as opposed to a vertex-attributed graph $G = (V, E, x)$; the algorithm will not use the edges of the graph to construct the local vertex subsets. Also note that the algorithm takes two input parameters: the number of local vertex subsets k and the local radius r for the output local vertex subsets.

Algorithm 2 Local vertex subset construction algorithm.

```
Input: Vertex attributes $x$, metric $d$, number of local vertex subsets $k$, and local radius $r$
 1: Select any index $j \in [n]$
 2: $U \leftarrow \{j\}$
 3: while $|U| < k$ do
 4:     $j \leftarrow \operatorname{argmax}_{k \in [n] \setminus U} \min_{i \in U} d(x_i, x_k)$
 5:     $U \leftarrow U \cup \{j\}$
 6: end while
 7: for $i \in [k]$ do
 8:     Select any index $\ell \in U$
 9:     $V^{(i)} \leftarrow \{j \in [n] : d(x_j, x_\ell) \le r\}$
10:     $U \leftarrow U \setminus \{\ell\}$
11: end for
12: Return $V^{(i)}$, $i \in [k]$
```

Greedy k-center clustering. The first part of Algorithm 2 (specifically Steps 1-6) runs part of the greedy algorithm of [16] designed for the *k-center problem.* Given points (attributes) $x_1, \dots, x_n \in X$ and a metric d on X, the goal in the k-center problem is to choose k of the n points to be centers to minimize the maximum distance (under d) from a point to its nearest center. In Step 1 the algorithm chooses any index $j \in [n]$; the index should be thought of as the index of the point x_j that will serve as the first "center." Next the algorithm initializes the set U of *center indices* in Step 2. Then in Step 4 the algorithm chooses the index of the next center to correspond to the point that is farthest from the centers whose indices have been chosen thus far. The algorithm adds the index to U in Step 5. The algorithm stops adding indices to U (i.e., the while loop terminates) once k indices have been chosen.

In Steps 7-11 the algorithm uses the center indices U to construct local vertex subsets $V^{(1)}, \dots, V^{(k)}$. Specifically, for each $\ell \in U$, the algorithm constructs a local vertex subset that contains the indices of all points that are within distance r to x_ℓ. In other words, the algorithm assigns each point to each center that is within distance r. Accordingly, a vertex could be either be in one local vertex subset, multiple local vertex subsets, or no vertex subsets. In contrast, the greedy algorithm of [16] assigns each point to a center that it is closest to, ensuring that each point is assigned to exactly one cluster.

Coverage, containment, and intersection properties. Our aim in running Algorithm 2 is to obtain local vertex subsets that (i) *cover* all of the vertices and (ii) individually *contain* enough vertices to recover local communities (later with a local community detection method), and (ii) *intersect* enough to eventually agglomerate via vertex agglomeration. Here we establish conditions under which these desired outcomes hold.

First we present a lower bound on k (in terms of r) that ensures Algorithm 2 returns local vertex subsets that cover all of the vertices. Our proof of Theorem 4.1 in Appendix A first constructs a $(r/2)$-net for $[0,1]^m$ (see Definition A.1 in Appendix A) of cardinality at most $(4\sqrt{m}/r)^m$ and then utilizes the fact that the greedy algorithm of [16] is a 2-approximation algorithm for the k-center problem.

Theorem 4.1. *Let $x \in [0,1]^m$. Further suppose that $0 < r < 1$ and $n \geq k \geq (4\sqrt{m}/r)^m$. Let $V^{(i)}$, $i \in [k]$ denote the local vertex subsets that Algorithm 2 outputs when run with input (x, d_2, k, r). Then the local vertex subsets $V^{(i)}$, $i \in [k]$ cover V; that is, $\cup_{i\in[k]} V^{(i)} = V$.*

Proof. See Appendix A. □

Note that the lower bound on k (and hence n) in Theorem 4.1 grows exponentially with the dimension m of X. From a theoretical standpoint, this will not be an issue because we are investigating recovery performance as $n \to \infty$ and m is a fixed constant. From a practical standpoint, we are interested in the case in which the attribute space is low-dimensional.

The next result that we present (see Theorem 4.2 below) assumes that the attributes are independently drawn from the uniform distribution $\mathcal{U}$. The result depends on the quantity $\gamma(r) := \min_{x\in[0,1]^m} \mathbb{P}_{y\sim\mathcal{U}}(y \in B(x,r))$, where $r > 0$. The quantity $\gamma(r)$ lower bounds the probability that the attributes of a vertex in the LSBM are in a given ball of radius r centered at a point in $[0,1]^m$. Also, $\gamma(r) > 0$. It is possibly to lower bound by an exponentially small quantity in m, but to keep the presentation clean, we forgo doing this. The result also depends on the notion of an *intersection graph.* For $\eta > 0$, define the *η-intersection graph* of the local vertex subsets $V^{(i)}$, $i \in [k]$ to have vertex set $[k]$ and an edge between vertices $i, j \in [k]$ if $|V_i \cap V_j| \geq \eta$.

Theorem 4.2. *Suppose that $x_i \sim \mathcal{U}$, $i \in [n]$ are independent. Further suppose that $0 < r < 1$ and $n \geq k \geq (16\sqrt{m}/r)^m$. Let $V^{(i)}$, $i \in [k]$ denote the local vertex subsets that Algorithm 2 outputs when run with input (x, d_2, k, r). Then the $\frac{\gamma_{\mathcal{D}}(r/8)}{2}n$-intersection graph of the local vertex subsets is connected with probability at least*

$$1 - \left(16\sqrt{m}/r\right)^m \exp\left(-\gamma\left(r/8\right)n/8\right).$$

Proof. See Appendix A. □

Theorem 4.2 shows that the $\frac{\gamma(r/8)}{2}n$-intersection graph of the local vertex subsets that Algorithm 2 outputs is connected with high probability. This implies that each local vertex subset contains a constant (in n) fraction of the vertices, namely $\frac{\gamma(r/8)}{2}$. Also we will see in

Subsection 6 that we can successfully agglomerate local communities via vertex agglomeration if the local vertex subsets intersect on enough vertices precisely in the sense that the intersection graph is connected. The proof of Theorem 4.2 follows from a similar argument to the one that we use in the proof of Theorem 4.1 together with a Chernoff bound and the union bound.

5 Local community detection

In this section we propose and study the algorithm that we will use for local community detection. More specifically, let $G = (V, E, x)$ be a vertex-attributed graph that has ground truth-community pair $C^* = (C_1^*, C_2^*)$. Also let $V' \subseteq V$ be a local vertex subset with respect to x. We consider applying a spectral community detection algorithm to the local adjacency matrix A' of V' (i.e., the adjacency matrix of the subgraph $G[V']$ in G induced by V') to recover local ground-truth communities $C_1^* \cap V'$ and $C_2^* \cap V'$. We remark that the algorithm is specifically designed for the case when (G, C^*) is generated according to the LSBM. At the end of this section we describe an alternative local community detection method that we use in the computational study of Section 8.

Define $n' := |V'|$ be the number of vertices in V'. For the sake of exposition, we will assume throughout this section that $V' = [n']$. Suppose that r is the local radius of V'. Also for the sake of exposition, we define the *local multiplier* of V' as $\lambda := \frac{r}{\sqrt{m}}$ (i.e., the radius, rescaled), and we say V' is a λ-local vertex subset.

Algorithm 3 Local spectral community detection algorithm.

```
Input: Local adjacency matrix $A'$ of $V' = [n']$, local multiplier $\lambda$ of $V'$, and model parameters $(p, q, \alpha, \beta)$
1: Compute a top eigenvector $u_1$ of
       $A' + (p + \alpha(1-\lambda))I - \frac{p + q + (\alpha - \beta)(1-\lambda)}{2} ee^\top$
2: Return the local communities $C = (C_1, C_2)$ defined by
       $C_1 = \{i \in [n'] : (u_1)_i < 0\}$   and   $C_2 = \{i \in [n'] : (u_1)_i \geq 0\}$
```

We present a description of the local spectral community detection algorithm in Algorithm 3. In Step 1 the algorithm computes a top eigenvector of the *centered* local adjacency matrix

$$\tilde{A} := A' + (p + \alpha(1-\lambda))I - \frac{p + q + (\alpha - \beta)(1-\lambda)}{2} ee^\top, \tag{4}$$

where I is the $n' \times n'$ identity matrix, and e is the n'-dimensional vector of all ones. In Step 2 the algorithm uses the sign of the entries in u_1 to construct local communities C_1 and C_2. Some motivation for the algorithm is as follows.

Motivation. Let us continue to treat x and C^* as fixed, but suppose that A' is generated according to (3) under x and C^*, i.e.,

$$\mathbb{P}(A'_{ij} = 1) = \begin{cases} p_{ij}(x, C^*) & i \neq j \\ 0 & i = j \end{cases} \tag{5}$$

for $i \leq j \in [n']$, and $A_{ij} = A_{ji}$ for $i > j$. Define the $n' \times n'$ matrix $P' := \mathbb{E}[A']$, where the expectation is with respect to (5) (i.e., not additionally with respect to x or C^*, as we have assumed that they are fixed). Note that $P'_{ij} = p_{ij}(x, C^*)$ if $i \neq j$, and $P'_{ij} = 0$ if $i = j$. Also define $u \in \{-1, +1\}^{n'}$ by

$$u_i := \begin{cases} +1 & i \in C_1^* \\ -1 & i \in C_2^*, \end{cases},$$

and define the $n' \times n'$ matrix D by

$$D_{ij} := \begin{cases} \alpha \left(\lambda - \frac{d_2(x_i, x_j)}{\sqrt{m}} \right) & i \neq j \text{ and either } i, j \in C_1^* \text{ or } i, j \in C_2^* \\ -\beta \left(\lambda - \frac{d_2(x_i, x_j)}{\sqrt{m}} \right) & i \neq j \text{ and either } i \in C_1^*,\ j \in C_2^* \text{ or } i \in C_2^*,\ j \in C_1^* \\ 0 & i = j. \end{cases}$$

Note that, since V' is a λ-local vertex subset with respect to x, we have $D_{ij} \geq 0$ if vertices i, j are in the same ground-truth community, and $D_{ij} \leq 0$ if vertices i, j are in different ground-truth communities.

From (3), for $i \neq j$, we can write

$$p_{ij}(x, C^*) = \begin{cases} p + \alpha(1 - \lambda) + \alpha \left(\lambda - \frac{d_2(x_i, x_j)}{\sqrt{m}} \right) & i, j \in C_1^* \text{ or } i, j \in C_2^* \\ q - \beta(1 - \lambda) - \beta \left(\lambda - \frac{d_2(x_i, x_j)}{\sqrt{m}} \right) & \text{otherwise}, \end{cases}$$

and hence

$$P' + (p + \alpha(1 - \lambda))I = \frac{p + q + (\alpha - \beta)(1 - \lambda)}{2} ee^\top + \frac{p - q + (\alpha + \beta)(1 - \lambda)}{2} uu^\top + D.$$

It follows from (4) and $P' = \mathbb{E}[A']$ that

$$\mathbb{E}[\tilde{A}] = \left(\frac{p - q + (\alpha + \beta)(1 - \lambda)}{2} \right) uu^\top + D. \tag{6}$$

As long as either $p - q > 0$ or $\alpha + \beta > 0$ and $\lambda < 1$ (ensuring that no entry of $\mathbb{E}[\tilde{A}]$ equals 0), we see from (6) that $\mathbb{E}[\tilde{A}]_{ij}$ is positive (negative) if the vertices i, j belong to the same (different) communities. It follows from the Perron-Frobenius theorem (applied to the positive matrix $\operatorname{diag}(u)\mathbb{E}[\tilde{A}]\operatorname{diag}(u)$) that the matrix $\mathbb{E}[\tilde{A}]$ has a unique top eigenvector, the signs of whose entries tell us the local ground-truth community assignments. Accordingly, by computing the top eigenvector of $\tilde{A}$ in Step 1 of Algorithm 3, we hope to recover local ground-truth community assignments.

Almost exact local recovery. Theorem 5.1 shows that Algorithm 3 almost exactly recovers local ground-truth communities if $D_\ell = \Omega\left(\frac{1}{\sqrt{n'}}\right)$ and $\lambda \leq \frac{1}{4}$ (i.e., if the radius of the local vertex subset is not too large). As discussed above, the condition that $p - q > 0$ or $\alpha + \beta > 0$ (and $\lambda < 1$, which is implied by $\lambda \leq \frac{1}{4}$) ensures that $\mathbb{E}[\tilde{A}]$ has all non-zero entries.

Theorem 5.1. *Suppose that $p - q > 0$ or $\alpha + \beta > 0$. Let $x \in \mathbb{R}^{n \times m}$, and let C^* be a global community pair. Suppose that $V' \subseteq V$ is a λ-local vertex subset with respect to x of cardinality $n' = |V'|$ such that $\lambda \leq \frac{1}{4}$. Further suppose that $A' \in \mathbb{R}^{n' \times n'}$ is generated according to* (5) *under x and C^*. Let C denote the output of Algorithm 3 when run with input $(A', \lambda, p, q, \alpha, \beta)$. Then there exist absolute constants $c_1, c_2 > 0$ such that*

$$\mathbb{P}\left(A(C, C^*) \geq 1 - \frac{c_1}{D_\ell^2 n'}\right) \geq 1 - \exp(-c_2 n').$$

Proof. See Appendix B. □

Practical implementation. Centering the local adjacency matrix in Algorithm 3 requires the model parameters (p, q, α, β). The standard spectral method that computes the second eigenvector of A' (instead of the top eigenvector of $\tilde{A}$) bypasses this centering step and does not require any model parameters. We were not able to establish a guarantee for the standard spectral method, but we use it anyways in our computational study in Section 8 instead of Algorithm 3. We leave it as an open question to establish a theoretical guarantee for the standard spectral method applied locally (and globally); see Section 9.

6 Local community agglomeration

Let $G = (V, E, x)$ be a vertex-attributed graph that has ground-truth community pair $C^* = (C_1^*, C_2^*)$. Also let $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$, $i \in [k]$ be local community pairs, i.e. $C_1^{(i)}, C_2^{(i)} \subseteq V$ such that $C_1^{(i)} \cap C_2^{(i)} = \emptyset$ for each $i \in [k]$. Assume that each vertex in V belongs to at least one of the local communities. In this section we propose an algorithm for agglomerating the local community pairs into a global community pair $C = (C_1, C_2)$ to recover C^*, presuming that the local community pairs have high enough agreement with their corresponding local ground-truth community pairs.

First we establish preliminary definitions. For $i \in [k]$, define $V^{(i)} = C_1^{(i)} \cup C_2^{(i)}$ to be the corresponding local vertex subset for $C^{(i)}$. Also, for $i \neq j \in [k]$, define the local communities $C^{(i \setminus j)} = (C_1^{(i \setminus j)}, C_2^{(i \setminus j)})$ by $C_\ell^{(i \setminus j)} := C_\ell^{(i)} \setminus V^{(j)}$ for $\ell \in [2]$.

We present a description of the agglomeration method in Algorithm 4. At each iteration the algorithm agglomerates two local community pairs together with *vertex agglomeration*. More specifically, in Step 3 the algorithm chooses the indices i and j of the two community pairs that intersect on the most vertices. Then in Steps 4-6 the algorithm agglomerates the communities within those pairs that share the most vertices. Exactly $k - 1$ vertex agglomerations are needed to obtain global communities; indeed, the while loop in the algorithm terminates after $k - 1$ iterations (as it holds that $|U| = 1$ by this point). The algorithm returns a community pair C that is a global community pair, as long each vertex in V belongs to at least one of the input local communities (as we assumed).

From local to global agreement. One would hope that if the agreement of each of the local community pairs is sufficiently high, then the agreement of the global community pair that Algorithm 4 outputs will be high. Here we present a result in this direction; see Theorem 6.1. The result additionally assumes that the η-intersection graph (for some $\eta > 0$)

Algorithm 4 Vertex agglomeration algorithm.

```
Input: Vertex attributed graph $G$, local community pairs $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$, $i \in [k]$
1: $U \leftarrow [k]$
2: while $|U| > 1$ do
3:     $(i, j) \leftarrow \text{argmax}_{i \neq j \in U} |V^{(i')} \cap V^{(j')}|$
4:     $(a, b) \leftarrow \text{argmax}_{(a',b') \in \{(1,2),(2,1)\}} |C_1^{(i)} \cap C_{a'}^{(j)}| + |C_2^{(i)} \cap C_{b'}^{(j)}|$
5:     $C_1^{(i)} \leftarrow C_1^{(i)} \cup C_a^{(j \setminus i)}$
6:     $C_2^{(i)} \leftarrow C_2^{(i)} \cup C_b^{(j \setminus i)}$
7:     $U \leftarrow U \setminus \{j\}$
8: end while
9: $C \leftarrow C^{(i)}$, where $i \in U$ is the only index left in $U$
10: Return community pair $C$
```

of the corresponding local vertex subsets is connected (which recall from Section 4 we can guarantee holds with high probability for certain values of η).

Theorem 6.1. *Let $G = (V, E, x)$ be a vertex-attributed graph that has ground-truth community pair $C^* = (C_1^*, C_2^*)$. Also let $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$, $i \in [k]$ be local community pairs. Suppose that each vertex in V belongs to at least one of the local communities. Also suppose that the η-intersection graph of $V^{(i)}$, $i \in [k]$ is connected. Finally suppose that*

$$\min_{i \in [k]} A(C^{(i)}, C^*) \geq 1 - \frac{\zeta \eta}{2^{k+1} n},$$

where $0 \leq \zeta \leq 1$. Then Algorithm 4 applied with input $(G, (C^{(i)})_{i \in [k]})$ outputs a global community pair $C = (C_1, C_2)$ that satisfies

$$A(C, C^*) \geq 1 - \frac{\zeta \eta}{4n}.$$

Proof. See Appendix C. □

Note that Theorem 6.1 requires that the disagreement of each of the local community pairs is exponentially small in k. It might be possible to improve this dependence on k in the analysis, but such a result is not required to establish almost exact recovery results with respect to n. We leave this as a direction for future research. More generally, it is of interest to consider other agglomeration methods as well (e.g., optimization-based agglomeration methods); see Section 9 for further discussion.

7 Community detection algorithm

We present a formal description of the community detection algorithm in Algorithm 5. Step 1 of the algorithm runs Algorithm 2 to obtain local vertex subsets $V^{(i)}$, $i \in [k]$. Next Steps 2-4 apply Algorithm 3 to each of the local vertex subsets to obtain local community pairs

$C^{(i)}$, $i \in [k]$. Finally the algorithm agglomerates the local community pairs into a global community pair in Step 5 using Algorithm 4.

Algorithm 5 Community detection algorithm.

```
Input: Vertex-attributed graph $G = (V, E, x)$, metric $d$, algorithm parameters $(k, r)$, and
       model parameters $(p, q, \alpha, \beta)$
1: Compute local vertex subsets $V^{(i)}$, $i \in [k]$ using Algorithm 2 run with input $(x, d, k, r)$
2: for $i = 1, \ldots, k$ do
3:     Run Algorithm 3 with input $(A^{(i)}, \frac{r}{\sqrt{m}}, p, q, \alpha, \beta)$ to obtain local community pair
       $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$, where $A^{(i)}$ is the local adjacency matrix $A^{(i)}$ of $V^{(i)}$
4: end for
5: Run Algorithm 4 with input $(G, (C^{(i)})_{i \in [k]})$ to obtain $C = (C_1, C_2)$
6: Return $C = (C_1, C_2)$
```

Almost exact recovery. Theorem 7.1 below presents conditions that ensure Algorithm 5 (under specific input specifications) almost exactly recovers LSBM communities. Our proof of the theorem follows from Theorem 4.1, 4.2, 5.1, and 6.1. Note that Theorem 3.1 follows from Theorem 7.1.

Theorem 7.1. *Suppose that $n \geq (64\sqrt{m})^m$. Let $(G, C^*) \sim \mathrm{LSBM}(n, m, p, q, \alpha, \beta)$. Also let C denote the global community that Algorithm 5 outputs when run with input G, d_2, $(k, r) = \left(\left\lceil (64\sqrt{m})^m \right\rceil, \frac{1}{4}\right)$, and (p, q, α, β). Then there are constants $c_i(m) > 0$, $i \in [4]$ (that depend on m) such that if*

$$D_\ell \geq \frac{c_1(m)}{\sqrt{n}}, \tag{7}$$

then it holds that

$$\mathbb{P}\left(A(C, C^*) \geq 1 - \frac{c_2(m)}{D_\ell^2 n}\right) \geq 1 - c_3(m)\exp(-c_4(m)n).$$

Proof. See Appendix D. □

The supposition that $n \geq (64\sqrt{m})^m$ in Theorem 7.1 is included to ensure that $k = \left\lceil (64\sqrt{m})^m \right\rceil \leq n$. Condition (7) is required to guarantee that local community agreement is high enough to achieve "successful" agglomeration.

Going into more detail, the proof of Theorem 7.1 reveals that $c_1(m)$ is doubly exponentially large in m. (This in turn through (7) implies an even larger lower bound on n than $(64\sqrt{m})^m$.) The intuition is as follows. Recall that Theorem 6.1 requires local disagreement to be exponentially small in k. Accordingly, because Theorem 7.1 requires k to exponentially large in m, we must set $c_1(m)$ to be doubly exponentially large in m.

8 Computational experiments

Our first goal in this computational study is to evaluate the empirical recovery performance of Algorithm 5 applied to LSBM instances. Our second goal is to investigate its viability for clustering training data.

We consider the following modified implementation of Algorithm 5 that does require model parameters (p, q, α, β) as input. Instead of applying Algorithm 3 in Steps 2-4 to construct local communities, we apply the standard spectral method to the local adjacency matrices. In our experiments with clustering data, we tune the input parameters (k, r) to a subset of the data that is assumed to be labeled. Specifically, we consider $k \in \{2, \dots, 10\}$ and $r \in \{0.1, 0.2, \dots, 0.5\}$ such that $2rk \geq 1$. The condition $2rk \geq 1$ ensures that it is possible to cover $[0, 1]$ with k Euclidean balls of radius r (intervals of length $2r$). (If Algorithm 4 does not output local vertex subsets that cover V, we arbitrarily assign uncovered vertices to one community after Algorithm 5 terminates.) We also tune the similarity parameter δ. As a benchmark, we compare the performance of Algorithm 5 with the standard spectral method; we also tune the similarity parameter for the benchmark.

We consider recovering LSBM communities in Subsection 8.1, clustering synthetic training data from Lipschitz sources in Subsection 8.2, and clustering the car dataset from Subsection 2.1 in Subsection 8.3.

8.1 Recovering LSBM communities

We generate $(G, C^*) \sim \text{LSBM}(n, 1, p, q, \alpha, \beta)$ with the following specification of parameters: $n = 200$, $p, q \in \{0.4, 0.45, \dots, 0.7\}$, and $\alpha = \beta = .3$. Under each specification of the parameters p and q, it holds that $p+\alpha > q-\beta$ (i.e., the ground-truth communities are locally distinguishable). Under some specifications, $p > q$, $p = q$, or $p < q$ (i.e., the parameters are not globally distinguishable).

We generate 10 LSBM instances under each specification of parameters and apply Algorithm 5 along with the standard spectral method to each of the instances. When applying Algorithm 5 to each instance, we give it the benefit of the doubt and choose the parameter specification that yields the highest accuracy. (In subsequent sections we actually tune the parameters.) We report average accuracy of both algorithms across all of the instances in the heat plots of Figure 5.

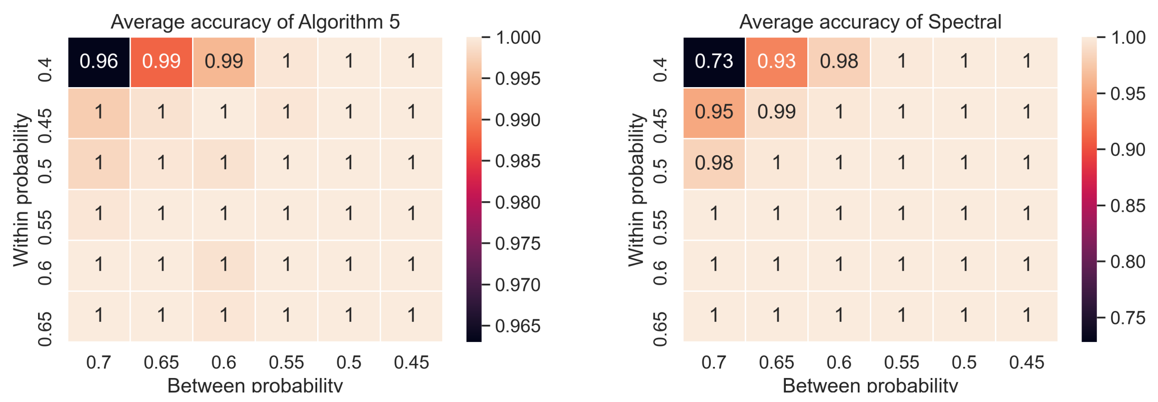

Figure 5: Average accuracy of Algorithm 5 and the standard spectral method against the within probability p and between probability q.

We observe that when $p \geq q$, the algorithms perform similarly, even when $p - q = 0$. This is interesting in light of the discussion in Subsection 3.1, i.e., perhaps the standard spectral algorithm almost exactly recovers when $D_\ell = \Omega(1/\sqrt{n})$ and $p - q \geq 0$. However, when $p \ll q$, we see that Algorithm 5 outperforms the standard spectral method. This is not surprising in light of the fact that the ground-truth communities are locally but not globally distinguishable in this regime.

8.2 Clustering training data from separable, Lipschitz sources

We revisit the setup from Subsection 2.1. Our goal is to understand how the Lipschitz constant L of the sources and the percentage of labeled data points (i.e, $|I|/n$) impacts the accuracy of Algorithm 5. (The percentage of labeled data points will impact our ability to tune the parameters (k, r) and δ.) We consider one-dimensional linear sources (i.e., $m = 1$) defined by $f_1(x) = 1/2 + Lx$ and $f_2(x) = -1/2 + Lx$ for $x \in \mathbb{R}$, where $L > 0$. The sources are separable and L-Lipschitz. We independently generate the noise terms ϵ_i, $i \in [n]$ from the normal distribution that has a mean of 0 and a standard deviation of $1/10$.

To generate an instance, we first independently sample $n = 200$ features x_i, $i \in [200]$ from the uniform distribution on $[0, 1]$. Then we uniformly at random partition the features into two equally sized subsets (i.e., each subset contains 100 features). Next we a generate a response $y_i = f_j(x_i) + \epsilon_i$ for each feature x_i, where $j \in \{1, 2\}$ captures which subset of the partition that the feature belongs to. We use the responses to generate the δ-similarity graph for the training data $D = \{(x_i, y_i)\}_{i \in [200]}$. Finally, we uniformly at random label $|I|$ of the data points.

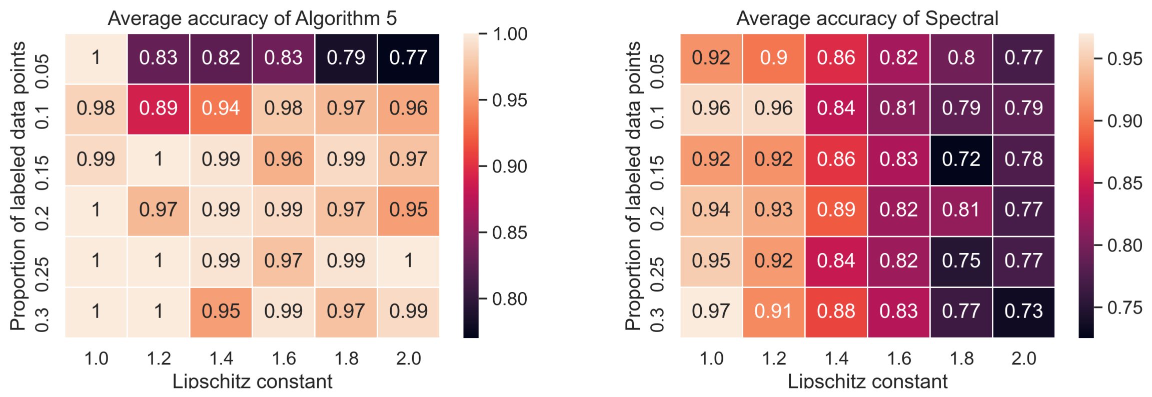

Figure 6: Average accuracy of Algorithm 5 and the standard spectral method against the Lipschitz constant L and the percentage $|I|/n$ of labeled data.

For each $L \in \{1, 1.2, \ldots, 2\}$ and $|I|/n \in \{.05, .1, \ldots, .3\}$, we generate 10 instances. We apply Algorithm 5 and the standard community detection method to each of these instances. More specifically, for each value of (k, r) (specified earlier at the beginning of Section 8) and $\delta \in \{0.2, 0.4, \ldots, 1\}$, we apply Algorithm 5 and choose the output communities that give the highest accuracy on the subset of labeled data. Similarly, for each $\delta \in \{0.2, 0.4, \ldots, 1\}$, we apply the standard spectral method and choose the output communities that give the highest accuracy on the subset of labeled data. We report the average accuracy of both methods in the heat plots of Figure 6.

We observe in general that Algorithm 5 outperforms the standard spectral method, especially on instances that have a larger Lipschitz constant, as the performance of the standard spectral method naturally deteriorates as the the Lipschitz constant gets larger. While the performance of the spectral method does not change much with the amount of labeled data, the performance of Algorithm 5 significantly improves when at least 10% (versus 5%) of the data is labeled.

8.3 Clustering the car dataset

Recall the car dataset from Subsection 2.1. In this subsection we consider using Algorithm 5 to cluster the dataset under different amounts $|I|$ of labeled data.

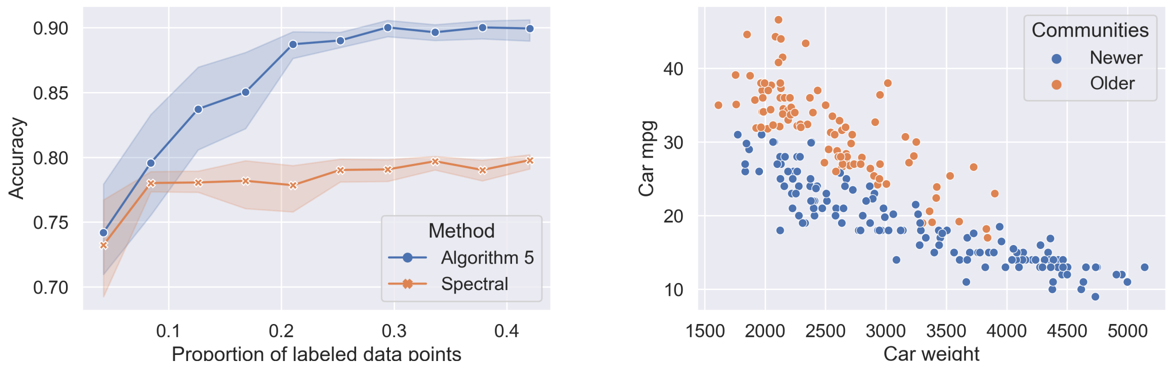

Figure 7: Average accuracy (with 95% confidence intervals) of Algorithm 5 and the standard spectral method against the percentage $|I|/n$ of labeled data, along with predicted communities output by Algorithm 5.

First we apply the transform $(x_i - \min_i x_i)/(\max_i x_i - \min_i x_i)$ to the features (i.e. the car weights) so that they take on values in $[0, 1]$. (This allows us to "successfully" tune the parameters (k, r) of Algorithm 5 over the values specified at the beginning of Section 8). Similar to Subsection 5, we uniformly at random label $|I|$ of the data points. For each value $|I| \in \{10, 20, \ldots, 100\}$, we generate 10 instances of labeled data in this way. We apply Algorithm 5 and the standard community detection method to each of the instances. We tune the parameters of the methods in the same way as Subsection 8.2, but we tune over $\delta \in \{1, \ldots, 10\}$ instead. We report average accuracy across the different fractions of available labeled data in Figure 7. We also present an illustration of communities output by Algorithm 5 that have an accuracy of approximately 0.91.

Both methods on average find communities that have higher agreement with the ground-truth communities when more data is available. Algorithm 5, however, tends to find communities with higher agreement, especially when more data is available. We observe in the second plot of Figure 7 Algorithm 5 fails to identify correct community structure amongst cars with very low or very high weight, as these cars typically only come from one source, either newer or older cars, respectively.

9 Discussion

We considered the problem of recovering the ground-truth communities of a vertex-attributed graph that are *locally distinguishable* in the sense that the ground-truth communities exhibit stronger community structure in more local parts of the graph (i.e., in subgraphs induced by vertices whose attributes are closer). As we demonstrated in Subsection 2.1, the problem arises when clustering training data that contains responses generated from different latent sources. More specifically, we showed that the ground-truth communities (corresponding to the latent sources) of certain *similarity graphs* for the training data are locally distinguishable under mild conditions. Locally distinguishable communities, however, are not necessarily *globally distinguishable* (i.e. have higher internal than external connection density), posing a challenge for standard community detection methods. Accordingly, in Sections 2.2 and 4-7, we proposed and studied a community detection algorithm that first clusters the vertex set into *local* vertex subsets, then applies a spectral community detection algorithm to each graph induced by one of the local vertex subsets to obtain *local* communities, and finally agglomerates the local communities into *global* communities. In Theorem 3.1 we established conditions under which the algorithm *almost exactly* recovers communities when applied to the *local stochastic block model* (LSBM), a generalization of the stochastic block model (SBM) that we propose in Section 3 that incorporates the local distinguishability property. In Section 8 we evaluated the empirical performance of the method on LSBM instances and its viability for clustering training data. We observed that our method outperforms a spectral method benchmark when ground-truth communities are locally but not globally distinguishable, as we would expect. Furthermore, the method's performance can significantly improve when a small subset of labeled data is available.

There are a number of interesting directions for future work. First, naturally it is of interest to develop exact (instead of almost exact) recovery guarantees. Second, in the context of clustering training data, it might make more sense to consider an alternative to the local spectral community detection method (i.e., Algorithm 3), particularly a method designed for low-dimensional clustering, given that clustering points with similar features more or less boils down to clustering their one-dimensional responses. Third, and we think most importantly, there is a need for a better agglomeration method; one direction is to develop an optimization-based agglomeration method. Furthermore, if partially labeled data is available, naturally it is of interest to exploit this when agglomerating. Finally, it is of interest to explore using our method for community detection problems that arise in applications other than clustering training data.

References

[1] `http://lib.stat.cmu.edu/datasets/`. Accessed: 2023-09-21.

[2] E. Abbe, *Community detection and stochastic block models: recent developments*, The Journal of Machine Learning Research, 18 (2017), pp. 6446–6531.

[3] E. Abbe, A. S. Bandeira, and G. Hall, *Exact recovery in the stochastic block model*, IEEE Transactions on information theory, 62 (2015), pp. 471–487.

[4] E. Abbe, J. Fan, K. Wang, and Y. Zhong, *Entrywise eigenvector analysis of random matrices with low expected rank*, Annals of statistics, 48 (2020), p. 1452.

[5] P. J. Bickel and A. Chen, *A nonparametric view of network models and newman–girvan and other modularities*, Proceedings of the National Academy of Sciences, 106 (2009), pp. 21068–21073.

[6] R. B. Boppana, *Eigenvalues and graph bisection: An average-case analysis*, in 28th Annual Symposium on Foundations of Computer Science (sfcs 1987), IEEE, 1987, pp. 280–285.

[7] O. Chapelle, B. Schölkopf, and A. Zien, *Semi-supervised learning*, (2006).

[8] Y. Chen and J. Xu, *Statistical-computational tradeoffs in planted problems and submatrix localization with a growing number of clusters and submatrices*, The Journal of Machine Learning Research, 17 (2016), pp. 882–938.

[9] D. S. Choi, P. J. Wolfe, and E. M. Airoldi, *Stochastic blockmodels with a growing number of classes*, Biometrika, 99 (2012), pp. 273–284.

[10] A. Condon and R. M. Karp, *Algorithms for graph partitioning on the planted partition model*, Random Structures & Algorithms, 18 (2001), pp. 116–140.

[11] W. S. DeSarbo and W. L. Cron, *A maximum likelihood methodology for clusterwise linear regression*, Journal of classification, 5 (1988), pp. 249–282.

[12] Y. Deshpande, S. Sen, A. Montanari, and E. Mossel, *Contextual stochastic block models*, Advances in Neural Information Processing Systems, 31 (2018).

[13] M. E. Dyer and A. M. Frieze, *The solution of some random np-hard problems in polynomial expected time*, Journal of Algorithms, 10 (1989), pp. 451–489.

[14] J. M. Franklin, C. Gopalakrishnan, A. A. Krumme, K. Singh, J. R. Rogers, J. Kimura, C. McKay, N. E. McElwee, and N. K. Choudhry, *The relative benefits of claims and electronic health record data for predicting medication adherence trajectory*, American heart journal, 197 (2018), pp. 153–162.

[15] S. Galhotra, A. Mazumdar, S. Pal, and B. Saha, *The geometric block model*, 32 (2018).

[16] T. F. Gonzalez, *Clustering to minimize the maximum intercluster distance*, Theoretical Computer Science, 38 (1985), pp. 293–306.

[17] G. S. Karanasiou, E. E. Tripoliti, T. G. Papadopoulos, F. G. Kalatzis, Y. Goletsis, K. K. Naka, A. Bechlioulis, A. Errachid, and D. I. Fotiadis, *Predicting adherence of patients with hf through machine learning techniques*, Healthcare technology letters, 3 (2016), pp. 165–170.

[18] F. McSherry, *Spectral partitioning of random graphs*, in Proceedings 42nd IEEE Symposium on Foundations of Computer Science, IEEE, 2001, pp. 529–537.

[19] H. Minc, *On the maximal eigenvector of a positive matrix*, SIAM Journal on Numerical Analysis, 7 (1970), pp. 424–427.

[20] M. E. Newman and A. Clauset, *Structure and inference in annotated networks*, Nature communications, 7 (2016), pp. 1–11.

[21] R. Quinlan, *Auto MPG.* UCI Machine Learning Repository, 1993. DOI: https://doi.org/10.24432/C5859H.

[22] T. A. Snijders and K. Nowicki, *Estimation and prediction for stochastic blockmodels for graphs with latent block structure*, Journal of classification, 14 (1997), pp. 75–100.

[23] T. Tao, *Topics in random matrix theory*, vol. 132, American Mathematical Society, 2023.

[24] J. E. Van Engelen and H. H. Hoos, *A survey on semi-supervised learning*, Machine learning, 109 (2020), pp. 373–440.

[25] V. Vu, *A simple svd algorithm for finding hidden partitions*, Combinatorics, Probability and Computing, 27 (2018), pp. 124–140.

[26] Z. Xu, Y. Ke, Y. Wang, H. Cheng, and J. Cheng, *A model-based approach to attributed graph clustering*, in Proceedings of the 2012 ACM SIGMOD international conference on management of data, 2012, pp. 505–516.

[27] J. Yang, J. McAuley, and J. Leskovec, *Community detection in networks with node attributes*, in 2013 IEEE 13th international conference on data mining, IEEE, 2013, pp. 1151–1156.

[28] T. Yang, R. Jin, Y. Chi, and S. Zhu, *Combining link and content for community detection: a discriminative approach*, in Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, 2009, pp. 927–936.

[29] Y. Yu, T. Wang, and R. J. Samworth, *A useful variant of the davis–kahan theorem for statisticians*, Biometrika, 102 (2015), pp. 315–323.

[30] H. Zanghi, S. Volant, and C. Ambroise, *Clustering based on random graph model embedding vertex features*, Pattern Recognition Letters, 31 (2010), pp. 830–836.

A Analysis of Algorithm 2

Here we set out to prove Theorem 4.1 and 4.2. First we introduce the notion of an ϵ-net (with respect to the Euclidean metric d_2) for a set $X \subseteq \mathbb{R}^m$.

Definition A.1. For $\epsilon > 0$ and $X \subseteq \mathbb{R}^m$, we say that a set $\mathcal{N} \subseteq X$ is an *ϵ-net for X* (with respect to the Euclidean metric d_2) if for each $y \in X$ there exists $z \in \mathcal{N}$ such that $d_2(y, z) \leq \epsilon$.

If $X \subseteq [0,1]^m$, then there is an ϵ-net for X of cardinality $|\mathcal{N}| \leq \left(\frac{2\sqrt{m}}{\epsilon}\right)^m$:

Lemma A.1. *Let $X \subseteq [0,1]^m$. For $0 < \epsilon < 1$, there is an ϵ-net $\mathcal{N}$ for X of cardinality $|\mathcal{N}| \leq \left(\frac{2\sqrt{m}}{\epsilon}\right)^m$.*

Proof. For $\zeta \in \mathbb{Z}_{>0}$, it is straightforward to see that there is a subset $\mathcal{N}' \subset [0,1]^m$ of cardinality $|\mathcal{N}'| = \zeta^m$ such that the collection $\left\{B_\infty\left(y, \frac{1}{2\zeta}\right)\right\}_{y \in \mathcal{N}'}$ partitions $[0,1]^m$. Because $B_\infty\left(y, \frac{1}{2\zeta}\right) \subseteq B_2\left(y, \frac{\sqrt{m}}{2\zeta}\right)$ for each $y \in \mathcal{N}'$, it follows that the collection $\left\{B_2\left(y, \frac{\sqrt{m}}{2\zeta}\right)\right\}_{y \in \mathcal{N}'}$ covers $[0,1]^m$. So, taking $\zeta = \left\lceil \frac{\sqrt{m}}{\epsilon} \right\rceil$, we have that there is a subset $\mathcal{N}_1 \subset [0,1]^m$ of cardinality $|\mathcal{N}_1| = \left\lceil \frac{\sqrt{m}}{\epsilon} \right\rceil^m \leq \left(\frac{2\sqrt{m}}{\epsilon}\right)^m$ such that the collection $\left\{B_2\left(y, \frac{\sqrt{m}}{2\zeta}\right)\right\}_{y \in \mathcal{N}_1}$ covers $[0,1]^m$. Because $B_2\left(y, \frac{\sqrt{m}}{2\zeta}\right) \subseteq B_2\left(y, \frac{\epsilon}{2}\right)$ for each $y \in \mathcal{N}_1$, the collection $\left\{B_2\left(y, \frac{\epsilon}{2}\right)\right\}_{y \in \mathcal{N}_1}$ also covers $[0,1]^m$. Let $\mathcal{N}_2 = \{y \in \mathcal{N}_1 : B_2\left(y, \frac{\epsilon}{2}\right) \cap X \neq \emptyset\}$. For each $y \in \mathcal{N}_2$, take $z_y \in B_2\left(y, \frac{\epsilon}{2}\right) \cap X$. Then the set $\mathcal{N} = \{z_y\}_{y \in \mathcal{N}_2}$ is an ϵ-net for X. □

With Lemma A.1 in hand, we are now prepared to prove Theorem 4.1:

Proof of Theorem 4.1. From Proposition A.1 there is a $\frac{r}{2}$-net $\mathcal{N}$ for $X = \{x_i\}_{i \in [n]}$ of cardinality $|\mathcal{N}| \leq \left(\frac{4\sqrt{m}}{r}\right)^m \leq k$. It follows that

$$\min_{y_1, \ldots, y_k \in X} \max_{i \in [n]} \min_{j \in [k]} d_2(x_i, y_j) \leq \frac{r}{2}. \tag{8}$$

In other words, the optimal value of the k-center problem is less than or equal to $\frac{r}{2}$. Because the greedy algorithm of [16] is a 2-approximation for the k-center problem, the set U of center indices that Algorithm 2 constructs satisfies

$$\max_{i \in [n]} \min_{j \in U} d_2(x_i, x_j) \leq 2 \min_{y_1, \ldots, y_k \in X} \max_{i \in [n]} \min_{j \in [k]} d_2(x_i, y_j) \leq r,$$

where the second inequality follows from (8). Thus, the local vertex subsets cover V. □

Before proving Theorem 4.2, consider Lemma A.2. The lemma states that every ball in a collection of balls centered at points in an ϵ-net for $[0,1]^m$ contains at least $\frac{\gamma(\epsilon)}{2}n$ attributes with high probability. The proposition follows from a Chernoff bound and the union bound.

Lemma A.2. *Suppose that $x_i \sim \mathcal{U}$, $i \in [n]$ are independent. Let $0 < \epsilon < 1$ and $\mathcal{N}$ be any ϵ-net for $[0,1]^m$. Then*

$$\mathbb{P}\left(|B(y,\epsilon) \cap \{x_i\}_{i\in[n]}| \geq \frac{\gamma(\epsilon)}{2}n,\ \forall y \in \mathcal{N}\right) \geq 1 - |\mathcal{N}| \exp\left(-\frac{\gamma_{\mathcal{D}}(\epsilon)\, n}{8}\right).$$

Proof. Let $y \in \mathcal{N}$. The random variable $N_y := |B(y,\epsilon) \cap \{x_i\}_{i\in[n]}|$ follows a binomial distribution with parameter n and success probability parameter $p_y = \mathbb{P}(x_i \in B(y,\epsilon)) \geq \gamma(\epsilon)$, where the inequality follows from the definition of $\gamma(\epsilon)$. Hence

$$\mathbb{P}\left(N_y \leq \frac{\gamma(\epsilon)}{2}n\right) \leq \mathbb{P}\left(N_y \leq \frac{p_y}{2}n\right) \leq \exp\left(-\frac{p_y n}{8}\right) \leq \exp\left(-\frac{\gamma(\epsilon) n}{8}\right), \tag{9}$$

where the second inequality follows from a standard Chernoff bound for the binomial distribution. Now observe that

$$\begin{aligned}\mathbb{P}\left(N_y \geq \frac{\gamma(\epsilon)}{2}n,\ \forall y \in \mathcal{N}\right) &= 1 - \mathbb{P}\left(\exists y \in \mathcal{N} : N_y \leq \frac{\gamma(\epsilon)}{2}n\right)\\ &\geq 1 - \sum_{y\in\mathcal{N}} \mathbb{P}\left(N_y \leq \frac{\gamma(\epsilon)}{2}n\right)\\ &\geq 1 - |\mathcal{N}| \exp\left(-\frac{\gamma(\epsilon)\, n}{8}\right),\end{aligned}$$

where the first inequality follows from the union bound, and the second inequality follows from (9). □

Now we prove Theorem 4.2:

Proof of Theorem 4.2. By Lemma A.1, there is a $\frac{r}{8}$-net $\mathcal{N}$ of cardinality $|\mathcal{N}| \leq \left(\frac{16\sqrt{m}}{r}\right)^m$ for $[0,1]^m$. Suppose that $|B(y,\frac{r}{8}) \cap \{x_i\}_{i\in[n]}| \geq \frac{1}{2}\gamma\left(\frac{\lambda}{8}\right) n$ for all $y \in \mathcal{N}$, which happens with probability at least $1 - \left(\frac{16\sqrt{m}}{r}\right)^m \exp\left(-\frac{\gamma(r/8)n}{8}\right)$ by Proposition A.2 together with $|\mathcal{N}| \leq \left(\frac{16\sqrt{m}}{r}\right)^m$. It is sufficient to show that the $\frac{\gamma(r/8)}{2}$-intersection graph of the local vertex subsets $V_1,\ldots,V_k$ is connected.

Let U denote the set of center indices that Algorithm 2 constructs. From Theorem 4.1, it follows that the collection $\{B\left(x_i,\frac{r}{4}\right)\}_{i\in U}$ covers $\{x_i\}_{i\in[n]}$. (While we run Algorithm 2 with input r, not $r/4$, the algorithm only uses r after the set U is constructed.) So, because $B(y,\frac{r}{8}) \cap \{x_i\}_{i\in[n]} \neq \emptyset$ for all $y \in \mathcal{N}$, the collection $\{B\left(x_i,\frac{r}{2}\right)\}_{i\in U}$ covers $\cup_{y\in\mathcal{N}} B_2\left(y,\frac{r}{8}\right) \supseteq [0,1]^m$. Consider the graph H that has vertex set U and an edge between vertices $i,j \in U$ if $B\left(x_i,\frac{r}{2}\right) \cap B\left(x_j,\frac{r}{2}\right) \neq \emptyset$. We see that H is connected because the collection $\{B\left(x_i,\frac{r}{2}\right)\}_{i\in U}$ covers $[0,1]^m$ and $\{x_i\}_{i\in U} \subseteq [0,1]^m$. Let $i,j \in U$ such that there is an edge between i and j in H. To show that the $\frac{1}{2}\gamma\left(\frac{r}{8}\right)$-intersection graph of the local vertex subsets $V_1,\ldots,V_k$ is connected, it is sufficient to show that $B(x_i,r) \cap B(x_j,r)$ contains at least $\frac{1}{2}\gamma\left(\frac{r}{8}\right)$ of the attributes. By definition of H, there exists $y \in B\left(x_i,\frac{r}{2}\right) \cap B\left(x_j,\frac{r}{2}\right)$. Take $z \in \mathcal{N}$ such that $z \in B\left(y,\frac{r}{8}\right)$. Then

$$B\left(y,\frac{r}{8}\right) \subset B\left(z,\frac{r}{2}\right) \subseteq B(x_i,r) \cap B(x_j,r),$$

and hence $B(x_i,r) \cap B(x_j,r)$ contains at least $\frac{1}{2}\gamma\left(\frac{r}{8}\right)$ attributes as $B\left(y,\frac{r}{8}\right)$ contains at least this amount of attributes. □

B Analysis of Algorithm 3

Our main goal in this section is to establish Theorem 5.1. Let us recall the setup of Section 5. Let $x \in \mathbb{R}^{n\times m}$ and $C^* = (C_1^*, C_2^*)$ be global ground-truth community pair. Suppose that $V' \subseteq V$ is a $\frac{1}{4}$-local vertex subset with respect to x of cardinality $n' = |V'|$. Without loss of generality, we assume that $V' = [n']$. Further suppose that $A' \in \mathbb{R}^{n'\times n'}$ is generated according to (5) under x and C^*. Let $\tilde{P} := \mathbb{E}[\tilde{A}]$, where $\tilde{A}$ is defined as in (4). In the proofs that we present in this section, we will use the definitions of P', u, and D from Section 5 along with the expression for $\mathbb{E}[\tilde{A}] = \tilde{P}$ in (6).

Our analysis follows in the footsteps of existing analysis for related spectral methods applied to the SBM. The main challenge is that this existing analysis heavily relies on analytic formulas for the eigenvectors and eigenvalues of $\tilde{P}$ that do not generalize to the context of the LSBM. Accordingly, we first establish properties of the eigenvalues and eigenvectors of $\tilde{P}$ in Lemma B.1 and B.2 below, respectively.

We introduce the following notation. Let $\lambda_1(M) \geq \lambda_2(M) \geq \cdots \geq \lambda_{n'}(M)$ denote the eigenvalues of a symmetric matrix $M \in \mathbb{R}^{n'\times n'}$. For each $i \in [n']$, define $u_i(M)$ to be an eigenvector of M with eigenvalue $\lambda_i(M)$. Finally let $\|M\|_F$ and $\|M\|_2$ denote the Frobenius and operator norm of M, respectively.

Lemma B.1 presents a lower bound on $\lambda_1(\tilde{P})$ and an upper bound on $\lambda_2(\tilde{P})$; these bounds will ultimately enable us to lower bound the difference $\lambda_1(\tilde{P}) - \lambda_2(\tilde{P})$.

Lemma B.1. *It holds that*

1. $\lambda_1(\tilde{P}) \geq \left(\frac{p-q+(\alpha+\beta)(1-\lambda)}{2}\right) n'$ *and*

2. $\lambda_2(\tilde{P}) \leq (\alpha+\beta)\lambda n'$.

Proof. First we define the sets

$$\mathcal{S}_1 := \{(i,j) \in V' \times V' : i \neq j \text{ and either } i,j \in C_1^* \text{ or } i,j \in C_2^*\},$$
$$\mathcal{S}_2 := \{(i,j) \in V' \times V' : i \neq j \text{ and either } i \in C_1^*,\ j \in C_2^* \text{ or } i \in C_2^*,\ j \in C_1^*\}.$$

Note that $\mathcal{S}_1$, $\mathcal{S}_2$, and $\{(i,i)\}_{i\in V'}$ partition $V' \times V' = [n']^2$.

Let us establish the first statement of the lemma. Observe that

$$\begin{aligned}
&\lambda_1(\tilde{P}) \\
&= \max_{\|x\|_2=1} x^\top \tilde{P} x \\
&\geq \left(\frac{1}{\sqrt{n'}}u\right)^\top \tilde{P} \left(\frac{1}{\sqrt{n'}}u\right) \\
&= \left(\frac{p-q+(\alpha+\beta)(1-\lambda)}{2}\right) n' + \frac{1}{n} u^\top D u \\
&= \left(\frac{p-q+(\alpha+\beta)(1-\lambda)}{2}\right) n' + \frac{1}{n'} \sum_{i,j\in[n']} D_{ij} u_i u_j
\end{aligned}$$

$$= \left(\frac{p-q+(\alpha+\beta)(1-\lambda)}{2}\right)n' + \frac{\alpha}{n'}\sum_{(i,j)\in\mathcal{S}_1}\left(\lambda - \frac{d_2(x_i,x_j)}{\sqrt{m}}\right) + \frac{\beta}{n'}\sum_{(i,j)\in\mathcal{S}_2}\left(\lambda - \frac{d_2(x_i,x_j)}{\sqrt{m}}\right)$$
$$\geq \left(\frac{p-q+(\alpha+\beta)(1-\lambda)}{2}\right)n',$$

where the first inequality follows from $\|u\|_2 = \sqrt{n}$, the second equality follows from (6), the fourth equality follows from the definitions of u and D, and the last inequality follows from the fact that V' is a λ-local vertex subset.

Now let us show the second statement of the lemma. Because $|D_{ij}| \leq \lambda\alpha$ for $(i,j)\in\mathcal{S}_1$, and $|D_{ij}| \leq \lambda\beta$ for $(i,j)\in\mathcal{S}_2$, we have that

$$\|D\|_F \leq \lambda\left(\sum_{(i,j)\in\mathcal{S}_1}\alpha^2 + \sum_{(i,j)\in\mathcal{S}_1}\beta^2\right)^{1/2} \leq \lambda n'\sqrt{\alpha^2+\beta^2} \leq (\alpha+\beta)\lambda n'. \tag{10}$$

From (6) and Weyl's inequality,

$$\lambda_2(\tilde{P}) \leq \left(\frac{p-q+(\alpha+\beta)(1-\lambda)}{2}\right)\lambda_2\left(uu^\top\right) + \lambda_1(D) \leq \|D\|_F \leq (\alpha-\beta)\lambda n',$$

where the second inequality follows from $\lambda_2(uu^\top) = 0$ and $\lambda_1(D) \leq \|D\|_F$, and the last inequality from (10). □

Lemma B.2 shows that the signs of the entries of $u_1(\tilde{P})$ capture the ground-truth communities. The lemma also establishes a lower bound on the magnitude of the entries.

Lemma B.2. *Suppose that $p-q>0$ or $\alpha>\beta$ and $\lambda<1$. Then the following statements hold.*

1. *Each entry of $u_1(\tilde{P})$ is nonzero.*
2. *For $i,j\in[n']$, the sign of $(u_1(\tilde{P}))_i$ is different from the sign of $(u_1(\tilde{P}))_j$ if $i\in C_1^*$, $j\in C_2^*$ or $i\in C_2^*$, $j\in C_1^*$.*
3. $\min_{i\in[n']}|(u_1(\tilde{P}))_i| \geq \frac{1-\lambda}{2\sqrt{n'}}$.

Proof. Let $\tilde{M} = \mathrm{diag}(u)\tilde{P}\,\mathrm{diag}(u)$. The matrix $\tilde{M}$ has the same eigenvalues as $\tilde{P}$, and $u_1(\tilde{M}) = \mathrm{diag}(u)u_1(\tilde{P})$ is an eigenvector of $\tilde{M}$ with eigenvalue $\lambda_1(\tilde{P})$ (i.e., the largest eigenvalue of $\tilde{M}$). As discussed in Section 5, all entries of $\tilde{M}$ are positive under the assumption that $p-q>0$ or $\alpha>\beta$ and $\lambda<1$. Consequently, from the Perron-Frobenius theorem, $\mathrm{diag}(u)u_1(\tilde{P})$ has either all positive or all negative entries. It follows that the first and second statement of the lemma hold.

To establish the third statement, it is sufficient to show that $\min_{i\in[n']}(u_1(\tilde{M}))_i \geq \frac{1-\lambda}{2\sqrt{n'}}$. The main result of [19] implies that

$$\min_{i\in[n']}(u_1(\tilde{M}))_i \geq \left(\frac{\min_{i,j\in[n']}\tilde{M}_{ij}}{\max_{i,j\in[n']}\tilde{M}_{ij}}\right)\max_{i\in[n']}(u_1(\tilde{M}))_i. \tag{11}$$

We bound the right hand side of (11) from below. Because $\|u_1(\tilde{M})\|_2 = 1$,

$$\max_{i\in[n']}(u_1(\tilde{M}))_i \geq \frac{1}{\sqrt{n'}}. \tag{12}$$

From (6) and the definition of $\tilde{M}$,

$$\begin{aligned}\min_{i,j\in[n']} \tilde{M}_{ij} &= \frac{p-q+(\alpha+\beta)(1-\lambda)}{2} + \min_{i,j\in[n']}|D_{ij}| \\ &\geq \frac{p-q+(\alpha+\beta)(1-\lambda)}{2} \\ &\geq \frac{(1-\lambda)(p-q+\alpha+\beta)}{2}. \end{aligned} \tag{13}$$

Similarly, we have that

$$\begin{aligned}\max_{i,j\in V'} \tilde{M}_{ij} &= \frac{p-q+(\alpha+\beta)(1-\lambda)}{2} + \max_{i,j\in[n']}|D_{ij}| \\ &\leq \frac{p-q+(\alpha+\beta)(1-\lambda)}{2} + \max\{\alpha,\beta\}\lambda \\ &= \frac{p-q+\alpha+\beta}{2} + \frac{(2\max\{\alpha,\beta\}-(\alpha+\beta))\lambda}{2} \\ &\leq \frac{p-q+\alpha+\beta}{2} + \frac{(\alpha+\beta)\lambda}{2} \\ &\leq p-q+\alpha+\beta. \end{aligned} \tag{14}$$

The second statement follows from (11) together with (12), (13), and (14). □

We are now prepared to prove Theorem 5.1. The proof utilizes the version of the Davis-Kahan theorem from [29] (specifically, Corollary 3). In our setting, the theorem tell us that surely

$$\min_{s\in\{-1,1\}}\|u_1(\tilde{A}) - su_1(\tilde{P})\|_2 \leq \frac{2^{3/2}\|\tilde{A}-\tilde{P}\|_2}{\lambda_1(\tilde{P}) - \lambda_2(\tilde{P})}. \tag{15}$$

We also use the following concentration inequality for the operator norm of $A' - P'$:

$$\mathbb{P}(\|A' - P'\|_2 \leq \hat{c}_1\sqrt{n'}) \geq 1 - \exp(-\hat{c}_2 n'), \tag{16}$$

where $\hat{c}_1, \hat{c}_2 > 0$ are absolute constants. The inequality holds because the entries of A' are (up to symmetry) independent Bernoulli random variables and $P' = \mathbb{E}[A']$; see, for example, Section 2.3 of [23].

Proof of Theorem 5.1. From Lemma B.2 and $\lambda \leq 1/4$, we have that $\min_{i\in[n']}|(u_1(\tilde{P}))_i| \geq \frac{3}{8\sqrt{n'}}$. Hence, surely

$$1 - A(C, C^*) \leq \frac{1}{n}\min_{s\in\{-1,1\}}\left\|\frac{8}{3}\sqrt{n}(u_1(\tilde{A}) - su_1(\tilde{P}))\right\|_2^2$$

$$
\begin{aligned}
&= \left(\frac{8}{3}\min_{s\in\{-1,1\}}\|u_1(\tilde{A}) - su_1(\tilde{P})\|_2\right)^2 \\
&\le \left(\frac{8}{3}\cdot\frac{2^{3/2}\|\tilde{A}-\tilde{P}\|_2}{\lambda_1(\tilde{P})-\lambda_2(\tilde{P})}\right)^2 \\
&= \left(\frac{8}{3}\cdot\frac{2^{3/2}\|A'-P'\|_2}{\lambda_1(\tilde{P})-\lambda_2(\tilde{P})}\right)^2
\end{aligned}
$$

where the second inequality follows from (15). It follows from (16) that there are absolute constants $\tilde{c}_1, c_2 > 0$ such that

$$
\begin{aligned}
&1-\exp(-c_2 n') \\
&\le \mathbb{P}\left(1-A(C,C^*) \le \left(\frac{\tilde{c}_1\sqrt{n'}}{\lambda_1(\tilde{P})-\lambda_2(\tilde{P})}\right)^2\right) \\
&\le \mathbb{P}\left(1-A(C,C^*) \le \left(\frac{\tilde{c}_1\sqrt{n'}}{((p-q)/2+(\alpha+\beta)3/8-(\alpha+\beta)/4)n'}\right)^2\right) \\
&= \mathbb{P}\left(1-A(C,C^*) \le \left(\frac{\tilde{c}_1}{((p-q)/2+(\alpha+\beta)/8)\sqrt{n'}}\right)^2\right) \\
&\le \mathbb{P}\left(1-A(C,C^*) \le \left(\frac{8\tilde{c}_1}{(p-q+\alpha+\beta)\sqrt{n'}}\right)^2\right) \\
&= \mathbb{P}\left(1-A(C,C^*) \le \frac{c_1}{D_\ell^2 n'}\right),
\end{aligned}
$$

where the second inequality follows from Lemma B.1 and $\lambda \le 1/4$, and the last equality follows from the definition of D_ℓ and taking $c_1 = 64\tilde{c}_1^2$. □

C Analysis of Algorithm 4

Here we prove Theorem 6.1. Let us recall the setup of Section 6. Let $G = (V, E, x)$ be a vertex-attributed graph that has ground-truth community pair $C^* = (C_1^*, C_2^*)$. Also let $C^{(i)} = (C_1^{(i)}, C_2^{(i)})$, $i \in [k]$ be local community pairs. Assume that each vertex in V belongs to at least one of the local communities. Recall that $V^{(i)} = C_1^{(i)} \cup C_2^{(i)}$ for $i \in [k]$, and define $n_i := |V^{(i)}|$. Also recall the definition of $C^{(i\setminus j)}$ for $i \neq j \in [k]$.

Consider any two local community pairs, say $C^{(1)}$ and $C^{(2)}$. Lemma C.1 shows that if we "appropriately" agglomerate $C^{(1)}$ and $C^{(2)}$, then we will not lose too much agreement. That is, the second condition of the lemma posits (without loss of generality) that for each $\ell \in [2]$, it holds that $C_1^{(\ell)}$ and $C_2^{(\ell)}$ share more vertices with the ground-truth communities C_1^* and C_2^*, respectively, than with C_2^* and C_1^*, respectively. So it is appropriate to agglomerate $C_1^{(1)}$ with $C_1^{(2)}$ and agglomerate $C_2^{(1)}$ with $C_2^{(2)}$, i.e. to construct new agglomerated communities $C_1^{(3)} = C_1^{(1)} \cup C_1^{(2\setminus 1)}$ and $C_2^{(3)} = C_2^{(1)} \cup C_2^{(2\setminus 1)}$.

Lemma C.1. *Let $0 \leq \delta \leq \frac{1}{2}$. For each $\ell \in [2]$, suppose that*

1. *$A(C^{(\ell)}, C^*) \geq 1 - \delta$ and*

2. *$A(C^{(\ell)}, C^*) = \frac{1}{n_\ell}(|C_1^{(\ell)} \cap C_1^*| + |C_2^{(\ell)} \cap C_2^*|)$.*

Let $C_1^{(3)} = C_1^{(1)} \cup C_1^{(2\setminus 1)}$ and $C_2^{(3)} = C_2^{(1)} \cup C_2^{(2\setminus 1)}$. Then for $C^{(3)} = (C_1^{(3)}, C_2^{(3)})$, we have

$$A(C^{(3)}, C^*) \geq 1 - 2\delta.$$

Proof. Let $n_{1\wedge 2} = |V^{(1)} \cap V^{(2)}|$. Also let $V^{(3)} = V^{(1)} \cup V^{(2)}$ and $n_3 = |V^{(3)}|$. Note that $n_3 = n_1 + n_2 - n_{1\wedge 2}$.

From the definition of $A(C^{(3)}, C^*)$,

$$\begin{aligned}
A(C^{(3)}, C^*) &\geq \frac{1}{n_3}(|C_1^{(3)} \cap C_1^*| + |C_2^{(3)} \cap C_2^*|) \\
&= \frac{1}{n_3}(|(C_1^{(1)} \cup C_1^{(2\setminus 1)}) \cap C_1^*| + |(C_2^{(1)} \cup C_2^{(2\setminus 1)}) \cap C_2^*|) \\
&= \frac{1}{n_3}(|C_1^{(1)} \cap C_1^*| + |C_1^{(2\setminus 1)} \cap C_1^*| + |C_2^{(1)} \cap C_2^*| + |C_2^{(2\setminus 1)} \cap C_2^*|) \\
&= \frac{1}{n_3}(|C_1^{(1)} \cap C_1^*| + |C_1^{(2)} \cap C_1^*| - |C_1^{(2)} \cap V^{(1)} \cap C_1^*|) \\
&+ \frac{1}{n_3}(|C_2^{(1)} \cap C_2^*| + |C_2^{(2)} \cap C_2^*| - |C_2^{(2)} \cap V^{(1)} \cap C_2^*|) \\
&= \frac{1}{n_3}(n_1 A(C^{(1)}, C) + n_2 A(C^{(2)}, C) - |C_1^{(2)} \cap V^{(1)} \cap C_1^*| - |C_2^{(2)} \cap V^{(1)} \cap C_2^*|) \\
&\geq \frac{1}{n_3}((1-\delta)(n_1 + n_2) - |C_1^{(2)} \cap V^{(1)}| - |C_2^{(2)} \cap V^{(1)}|) \\
&= \frac{1}{n_3}((1-\delta)(n_3 + n_{1\wedge 2}) - |V^{(1)} \cap V^{(2)}|) \\
&= \frac{1}{n_3}((1-\delta)n_3 - \delta n_{1\wedge 2}) \\
&\geq 1 - 2\delta
\end{aligned}$$

where the fourth equality follows from the second condition of the lemma, the second inequality follows from the first condition of the lemma, and the last inequality follows from $n_{1\wedge 2} \leq n_3$. □

Next we show in Lemma C.2 that if the agreement of local community pairs $C^{(1)}$ and $C^{(2)}$ is sufficiently high (relative to the number of vertices that they intersect on), then vertex agglomeration appropriately agglomerates them. Condition (17) ensures that Algorithm 4 selects the appropriate local communities to agglomerate together in Step 4.

Lemma C.2. *Suppose that the following conditions hold.*

1. *$|V^{(1)} \cap V^{(2)}| \geq \eta$.*

2. *For $\ell \in [2]$, it holds that $A(C^{(\ell)}, C^*) \geq 1 - \frac{\eta}{5n}$ and*

3. $A(C^{(\ell)}, C^*) = \frac{1}{n_\ell}(|C_1^{(\ell)} \cap C_1^*| + |C_2^{(\ell)} \cap C_2^*|)$.

Then

$$|C_1^{(1)} \cap C_1^{(2)}| + |C_2^{(1)} \cap C_2^{(2)}| > |C_1^{(1)} \cap C_2^{(2)}| + |C_2^{(1)} \cap C_1^{(2)}|. \tag{17}$$

Proof. For the sake of contradiction, suppose that

$$|C_1^{(1)} \cap C_1^{(2)}| + |C_2^{(1)} \cap C_2^{(2)}| \leq |C_1^{(1)} \cap C_2^{(2)}| + |C_2^{(1)} \cap C_1^{(2)}|. \tag{18}$$

Now observe that

$$\begin{aligned}
\eta &\leq |(C_1^{(1)} \cup C_2^{(1)}) \cap (C_1^{(2)} \cup C_2^{(2)})| \\
&= |C_1^{(1)} \cap C_1^{(2)}| + |C_1^{(1)} \cap C_2^{(2)}| + |C_2^{(1)} \cap C_1^{(2)}| + |C_2^{(1)} \cap C_2^{(2)}| \\
&\leq 2(|C_1^{(1)} \cap C_2^{(2)}| + |C_2^{(1)} \cap C_1^{(2)}|) \\
&= 2(|C_1^{(1)} \cap C_2^{(2)} \cap C_1^*| + |C_1^{(1)} \cap C_2^{(2)} \cap C_2^*| + |C_2^{(1)} \cap C_1^{(2)} \cap C_1^*| + |C_2^{(1)} \cap C_1^{(2)} \cap C_2^*|) \\
&\leq 2(|C_2^{(2)} \cap C_1^*| + |C_1^{(1)} \cap C_2^*| + |C_2^{(1)} \cap C_1^*| + |C_1^{(2)} \cap C_2^*|) \\
&= 2(n_1(1 - A(C^{(1)}, C^*)) + n_2(1 - A(C^{(2)}, C^*))) \\
&\leq 2(\frac{n_1\eta}{5n} + \frac{n_2\eta}{5n}) \\
&\leq \frac{4\eta}{5}
\end{aligned}$$

where the second inequality follows from (18), the fourth inequality from the second condition of the lemma, and the last inequality from $n_1, n_2 \leq n$. We have obtained a contradiction. □

We are now prepared to prove Theorem 6.1:

Proof of Theorem 6.1. It is sufficient to show that after the ℓ-th agglomeration for $0 \leq \ell \leq k-1$, it holds that

$$\min_{i \in U} A(C^{(i)}, C^*) \geq 1 - \frac{\zeta\eta}{2^{k+1-\ell}n}. \tag{19}$$

We proceed by induction. Clearly (19) holds for $\ell = 0$, so suppose it holds for $0 \leq \ell < k-1$. Then at the beginning of the iteration $\ell + 1$ of Algorithm 4, we have that

$$\min_{i \in U} A(C^{(i)}, C^*) \geq 1 - \frac{\zeta\eta}{2^{k+1-\ell}n} \geq 1 - \frac{\zeta\eta}{8n} \geq 1 - \frac{\zeta\eta}{5n},$$

where the second inequality follows from $\ell < k-1$. Lemma C.2 implies that the appropriate communities are agglomerated together in iteration $\ell + 1$. It follows from Lemma C.1 that after the $(\ell + 1)$-th agglomeration,

$$\min_{i \in U} A(C^{(i)}, C^*) \geq 1 - \frac{2\zeta\eta}{2^{k+1-\ell}n} = 1 - \frac{\zeta\eta}{2^{k+1-(\ell+1)}n}.$$

Thus, by induction, the desired result holds. □

D Analysis of Algorithm 5

Proof of Theorem 7.1. We break the proof into four steps. The first three correspond to the three main steps of Algorithm 5, and the last combines the results from the first three together.

Step 1: Local vertex subset construction. Let $\eta = \frac{\gamma(1/32)n}{2}$. We introduce the event

$$E_1 = \left\{\cup_{i\in[k]} V^{(i)} = V \text{ and } \eta\text{-intersection graph of } V^{(i)}, i \in [k] \text{ is connected}\right\}.$$

Because $k = \lceil (64\sqrt{m})^m \rceil \geq (16\sqrt{m})^m = \left(\frac{4\sqrt{m}}{r}\right)^m$ (as $r = \frac{1}{4}$), we have from Theorem 4.1 that $\cup_{i\in[k]} V^{(i)} = V$ surely. Accordingly,

$$\begin{aligned}
\mathbb{P}(E_1) &= \mathbb{P}(\eta\text{-intersection graph of } V^{(i)}, i \in [k] \text{ is connected}) \\
&\geq 1 - \left(64\sqrt{m}\right)^m \exp\left(-\frac{\gamma(1/32)n}{8}\right), \\
&\geq 1 - \hat{c}_1(m)\exp(-\hat{c}_2(m)n), \qquad (20)
\end{aligned}$$

where the first inequality follows from Theorem 4.2 with $r = \frac{1}{4}$, and the second equality follows from taking $\hat{c}_1(m) = (64\sqrt{m})^m$ and $\hat{c}_2(m) \leq \frac{\gamma(1/32)}{8}$. (Such a constant $\hat{c}_2(m)$ indeed exists; see Section 4.)

Step 2: Local community detection. For each $i \in [k]$, consider the event

$$E_{2i} = \left\{A(C^{(i)}, C^*) \geq 1 - \frac{\hat{c}_3}{D_\ell^2 \eta}\right\},$$

where $\hat{c}_3$ equals the absolute constant c_1 in Theorem 5.1. Because $V^{(i)}$ is a $\frac{1}{4\sqrt{m}}$-local vertex subset and $E_1 \subseteq \{|V^{(i)}| \geq \eta\}$, we can apply Theorem 5.1 to obtain that

$$\mathbb{P}(E_{2i} \mid E_1) \geq 1 - \exp(-\hat{c}_4 \eta), \qquad (21)$$

where $\hat{c}_4$ equals the absolute constant c_2 in the theorem. For

$$E_2 = \left\{\min_{i\in[k]} A(C^{(i)}, C^*) \geq 1 - \frac{\hat{c}_3}{D_\ell^2 \eta}\right\},$$

we can apply (21) and the union bound to obtain that

$$\begin{aligned}
\mathbb{P}(E_2 \mid E_1) &\geq 1 - k\exp(-\hat{c}_4\eta) \\
&\geq 1 - (65\sqrt{m})^m \exp\left(-\frac{\hat{c}_4\gamma(1/32)n}{2}\right) \\
&= 1 - \hat{c}_5(m)\exp(-\hat{c}_6(m)n), \qquad (22)
\end{aligned}$$

where the second inequality follows from $k \leq (65\sqrt{m})^m$ and the definition of η, and the equality from taking $\hat{c}_5(m) = (65\sqrt{m})^m$ and $\hat{c}_6(m) \leq \frac{\hat{c}_4\gamma(1/32)}{2}$.

Step 3: Agglomeration. Let us take

$$c_1(m) = \frac{\hat{c}_3^{1/2} 2^{(\lceil (64\sqrt{m})^m \rceil + 3)/2}}{\gamma(1/32)} = \frac{\hat{c}_3^{1/2} 2^{(k+3)/2}}{\gamma(1/32)},$$

and define

$$\zeta = \frac{c_1(m)^2}{D_\ell^2 n}.$$

First note that from (7) that $\zeta \leq 1$. Second note that from the definitions of ζ and η,

$$E_2 = \left\{ \min_{i \in [k]} A(C^{(i)}, C^*) \geq 1 - \frac{\zeta \eta}{2^{k+1} n} \right\}.$$

Accordingly, for $E_3 = \left\{ A(C, C^*) \geq 1 - \frac{\zeta\eta}{4n} \right\}$, we can apply Theorem 6.1 to obtain that

$$\mathbb{P}(E_3 \mid E_1 \cap E_2) = 1. \tag{23}$$

Step 4: Putting it all together. Finally, take

$$c_2(m) = \frac{\hat{c}_3 2^{(64\sqrt{m})^m}}{\gamma(1/32)} = \frac{\hat{c}_3 2^k}{\gamma(1/32)}.$$

Then we have that $E_3 = \left\{ A(C, C^*) \geq 1 - \frac{c_2(m)}{D_\ell^2 n} \right\}$. Observe that

$$\begin{aligned}
\mathbb{P}(E_3) &\geq \mathbb{P}(E_1 \cap E_2 \cap E_3) \\
&= \mathbb{P}(E_3 \mid E_1 \cap E_2) \mathbb{P}(E_2 \mid E_1) \mathbb{P}(E_1) \\
&\geq 1 - \hat{c}_1(m) \exp(-\hat{c}_2(m) n) - \hat{c}_5(m) \exp(-\hat{c}_6(m) n),
\end{aligned}$$

where the second inequality follows from (20), (22), and (23). Thus the desired result follows. □