Locally Differentially Private Graph Clustering via the Power Iteration Method

Abstract

We propose a locally differentially private graph clustering algorithm. Previous works have explored this problem, including approaches that apply spectral clustering to graphs generated via the randomized response algorithm. However, these methods only achieve accurate results when the privacy budget is in $\Omega(\log n)$, which is unsuitable for many practical applications. In response, we present an interactive algorithm based on the power iteration method. Given that the noise introduced by the largest eigenvector constant can be significant, we incorporate a technique to eliminate this constant. As a result, our algorithm attains local differential privacy with a constant privacy budget when the graph is well-clustered and has a minimum degree of $\tilde{\Omega}(\sqrt{n})$. In contrast, while randomized response has been shown to produce accurate results under the same minimum degree condition, it is limited to graphs generated from the stochastic block model. We perform experiments to demonstrate that our method outperforms spectral clustering applied to randomized response results.

1. Introduction

As the adoption of artificial intelligence expands, ensuring the protection of user privacy has become a critical priority. Various techniques have been proposed to tackle privacy concerns, with differential privacy emerging as a leading approach. Differential privacy, introduced in (Dwork, 2008), quantifies the privacy leakage of a system using a parameter known as the privacy budget. The core idea involves introducing noise to users' data to obscure individual information while still enabling meaningful statistical analysis. The challenge of designing algorithms that can draw accurate insights from this noisy data has garnered significant attention from researchers (Zhu et al., 2017), as it is essential to balance privacy protection with the utility of the resulting analysis.

In this work, we focus on a specific variant of differential privacy known as local differential privacy (LDP) (Kasiviswanathan et al., 2011). Unlike traditional differential privacy, which allows data collection before noise is added, LDP requires users to anonymize their data directly on their local devices before transmitting it to a central server. This approach ensures that sensitive information remains protected during transmission, as the data is already corrupted at the source. LDP has been adopted by several major companies (Erlingsson et al., 2014; Apple's Differential Privacy Team, 2017) in their services to safeguard user privacy while still enabling data analysis at scale.

We focus on developing LDP algorithms for social networks, where users are represented as nodes and their relationships as edges. Since these connections are considered sensitive, they are protected using privacy notions such as edge LDP (Qin et al., 2017) or node LDP (Ye et al., 2020). However, with some exceptions like (Zhang et al., 2020), node LDP is generally too stringent, making it difficult to release useful information in most applications. As a result, the majority of research in LDP has centered around the more practical edge LDP framework (Imola et al., 2021).

To protect user's information, one widely used technique is randomized response, also known as edge flipping (Warner, 1965; Mangat, 1994; Wang et al., 2016). In this method, before a user sends a bit vector which encodes their list of friends to a central server, each bit in the vector is flipped with a certain probability. The server aggregates the obfuscated adjacency vector to construct an obfuscated version of the graph. Although it is possible to compute various graph statistics from this obfuscated data, the accuracy of these statistics is often reduced. Algorithms designed specifically to publish particular statistics tend to offer more precise and insightful results about the graph (Imola et al., 2021; 2022).

Graph clustering illustrates how analyzing a graph obfuscated by randomized response can lead to inaccurate results. Let n be the number of nodes in the input graph. In (Hehir et al., 2022), the authors demonstrated that spectral clustering (Ng et al., 2001) can yield accurate results with a privacy budget in $O(1)$, provided the input graphs are generated from stochastic block models and have an average degree of $\Theta(\sqrt{n})$ (Holland et al., 1983). For general graphs, (Mukherjee & Suppakitpaisarn, 2023) showed that applying spectral clustering to randomized response data only yields accurate

. **AUTHORERR: Missing \icmlcorrespondingauthor.**

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

results when the privacy budget $\epsilon \in \Omega(\log n)$, which is too large for many real-world applications. Furthermore, even for dense graphs, when $\epsilon \in o(\log n)$, the authors identified a class of graphs for which clustering results are inaccurate.

Although numerous algorithms have been proposed for clustering under differential privacy (Ji et al., 2020; Mohamed et al., 2022; Chen et al., 2023; Imola et al., 2023; Epasto et al., 2024; He et al., 2024), relatively few have been developed specifically for publishing clustering results under edge LDP. Aside from the work mentioned in the previous paragraph, the only other algorithm we are aware of targets node LDP rather than edge LDP (Fu et al., 2023).

1.1. Our Contributions

In this work, we aim to develop a dedicated algorithm for graph clustering under the edge LDP framework. Rather than using non-interactive methods like the randomized response algorithm, we propose an interactive approach, which has been shown to achieve better performance for many edge LDP tasks (Henzinger et al., 2024; Hillebrand et al., 2024).

Specifically, we draw inspiration from the work in (Betzer et al., 2024), where the authors employ multi-round interactive algorithms to compute iterative matrix multiplications for Katz centrality. Since spectral clustering can also be derived through iterative matrix multiplication using the Power Iteration Clustering (PIC) algorithm (Lin & Cohen, 2010; Boutsidis et al., 2015), we propose extending this approach to calculate clusters via the PIC algorithm under the edge LDP framework.

Unfortunately, calculating the PIC algorithm under the edge LDP framework is not straightforward. While the goal is to compute the second eigenvector through the iterative process, the largest component of the result is the first eigenvector. In a non-private setting, the first eigenvector, being a uniform vector, does not interfere with the calculation of the PIC algorithm. However, when protecting users' sensitive information under edge LDP, noise must be added at a magnitude comparable to the largest terms. This causes the noise to dominate the result, especially as the number of iterations increases, leading to a significant loss in accuracy.

We propose a technique to eliminate the largest constant term, enabling the development of an algorithm that achieves accurate results with a constant privacy budget when the minimum degree of the input graph is $\tilde{\Omega}(\sqrt{n})$. Recall that randomized response is proven to yield accurate results for graphs generated by the stochastic block model when the minimum degree is $\tilde{\Omega}(\sqrt{n})$. Our algorithm, however, provides precise results under the same minimum degree condition but applies to general graphs, not limited to those generated by the model. This extends the applicability of our clustering algorithm to a wider range of input graphs.

Our algorithm is computationally efficient. It requires $O(\log n)$ interactions between users and the central server, with each node having a computational complexity of $O(n)$ per iteration. The central server also has a computational complexity of $O(n)$ per iteration. Consequently, the total computation time of our algorithm is $O(n \log n)$. Additionally, the communication cost for each user is also $O(n \log n)$.

Compared to the spectral clustering algorithm applied to the randomized response results (Hehir et al., 2022; Mukherjee & Suppakitpaisarn, 2023), our iterative method is significantly more memory-efficient. In the previous approach, the server requires $\Theta(n^2)$ bits of memory to store the randomized response results (Imola et al., 2022; Hillebrand et al., 2023). In contrast, our algorithm reduces the memory requirement to $\Theta(n)$ for both the server and the users. This improvement enables our method to handle graphs with a large number of nodes, which would be infeasible to process using the earlier algorithm.

We validate our algorithm through experiments on graphs generated using the stochastic block model (Holland et al., 1983) and the Reddit graph (Hamilton et al., 2017). Compared to applying the spectral clustering algorithm to the randomized response results (Hehir et al., 2022), our algorithm produces clustering results that are closer to those of the original spectral clustering algorithm in almost all cases. Notably, there are instances where the previous algorithm yields random outcomes, while our algorithm consistently produces results identical to the original spectral clustering.

2. Preliminaries

2.1. Notation

Throughout this paper, we consider a graph $G = (V, E)$ with n vertices. Let $S \subseteq V$ represent a subset of vertices, and $\overline{S}$ denote its complement $V \setminus S$.

Let S and S' be two disjoint subsets of V (meaning $S \cap S' = \varnothing$). We denote by $e_G(S, S')$ the number of edges in G that have one endpoint in S and the other in S'. For each subset $S \subseteq V$, let $\mathrm{Vol}_G(S)$ denote the number of edges with both endpoints in S. We refer to $\mathrm{Vol}_G(S)$ as the *volume* of S.

For $S, S' \subseteq V$, the quantity $d_{\mathrm{vol}}(S, S')$ is defined as $\min(\mathrm{Vol}_G(S \triangle S') + \mathrm{Vol}_G(\overline{S} \triangle \overline{S'}), \mathrm{Vol}_G(S \triangle \overline{S'}) + \mathrm{Vol}_G(\overline{S} \triangle S'))$. Since $S \triangle S' = \overline{S} \triangle \overline{S'}$, this simplifies to $d_{\mathrm{vol}}(S, S') = \min\left(2\mathrm{Vol}_G(S \triangle S'), 2\mathrm{Vol}_G(S \triangle \overline{S'})\right)$. Two cuts $(S, \overline{S})$ and $(S', \overline{S'})$ are considered similar if $d_{\mathrm{vol}}(S, S')$ is small. We also define the *normalized discrepancy* as

$$d_{\mathrm{norm}}(S, S') = \frac{d_{\mathrm{vol}}(S, S')}{\mathrm{Vol}_G(V)}. \tag{1}$$

Given that $d_{\mathrm{vol}}(S, S') \leq \mathrm{Vol}_G(V)$, normalization ensures that $0 \leq d_{\mathrm{norm}}(S, S') \leq 1$. When S is fixed and nodes are randomly assigned to S' with uniform probability, $d_{\mathrm{norm}}(S, S')$ tends to be close to 1.

Any real symmetric $n \times n$ matrix A has n real eigenvalues. We denote the i-th smallest eigenvalue of A as $\lambda_i(A)$, so that $\lambda_1(A) \geq \lambda_2(A) \geq \cdots \geq \lambda_n(A)$. The eigenvector corresponding to $\lambda_i(A)$ is denoted by $\mathbf{v}_i(A) = [\nu_{i,1}, \ldots, \nu_{i,n}]^\intercal$.

For each $i \in [1, n]$, let $a_i = [a_{i,1}, \ldots, a_{i,n}]^\intercal$ represent the adjacency list of user v_i, where $a_{i,j} = 1$ signifies the existence of an edge between v_i and v_j (i.e., $(v_i, v_j) \in E$), and $a_{i,j} = 0$ indicates no edge. The degree of node v_i, denoted by d_i, reflects the number of edges connected to v_i. In the context of a locally differentially private algorithm, it is assumed that each user v_i is aware only of their own adjacency vector a_i, which contains sensitive personal information.

2.2. Edge Local Differential Privacy

We define two adjacency lists, a and a', as neighboring if they differ by exactly one bit, meaning that one can be transformed into the other by either adding or removing a single edge connected to node v_i. The concept of edge local differential privacy is formalized as follows:

Definition 2.1 (ϵ-Edge LDP Query)**.** Let $\epsilon > 0$. A randomized query $\mathcal{R}$ is said to satisfy ϵ-edge local differential privacy (ϵ-edge LDP) if, for any pair of neighboring adjacency lists a and a', and any possible outcome set S, $\mathbb{P}\left[\mathcal{R}(a) \in S\right] \leq e^{\epsilon}\mathbb{P}\left[\mathcal{R}(a') \in S\right]$.

Definition 2.2 (ϵ-edge LDP Algorithm (Qin et al., 2017))**.** An algorithm $\mathcal{A}$ is said to be ϵ-edge LDP if, for any user v_i, and any sequence of queries $\mathcal{R}_1, \ldots, \mathcal{R}_\kappa$ posed to user v_i, where each query $\mathcal{R}_j$ satisfies ϵ_j-edge local differential privacy (for $1 \leq j \leq \kappa$), the total privacy loss is bounded by $\epsilon_1 + \cdots + \epsilon_\kappa \leq \epsilon$.

If an algorithm $\mathcal{A}$ is ϵ-edge LDP, it is also said to have a privacy budget of ϵ. Next, we introduce a query that satisfies ϵ-edge LDP which designed to estimate a real-valued statistic based on the adjacency vector.

Definition 2.3 (Edge Local Laplacian Query (Hillebrand et al., 2023))**.** Let $f : \{0, 1\}^n \to \mathbb{R}$ be a function defined on adjacency lists, and let $a \sim a'$ represent neighboring adjacency lists. The global sensitivity of f, denoted as Δ_f, is defined as: $\Delta_f = \max_{a \sim a'} |f(a) - f(a')|$.

For any $\epsilon > 0$, a query that returns $f(a) + \mathrm{Lap}(\Delta_f/\epsilon)$ is ϵ-edge LDP. Here, $\mathrm{Lap}(b)$ refers to noise sampled from the Laplace distribution with scale parameter b.

2.3. Spectral Clustering

For a given graph G, the primary objective of clustering techniques is to identify a cut $(S, \overline{S})$ such that the number of edges crossing between S and $\overline{S}$, denoted by $e_G(S, \overline{S})$, is minimized, while most of the edges are concentrated within S or $\overline{S}$. To avoid trivial cuts (such as when S contains only a single vertex), it is common to define the *conductance*, $\phi_G(S) = e_G(S, \overline{S})/\min\{\mathrm{Vol}_G(S), \mathrm{Vol}_G(\overline{S})\}$, and seek cuts that minimize $\phi_G(S)$ (Shi & Malik, 2000). The conductance of the graph, denoted by $\phi(G)$, is given by $\phi(G) = \min_{\varnothing \subsetneq S \subsetneq V} \phi_G(S)$. Unless otherwise stated, we use S^* to denote the subset that achieves the minimum normalized cut, where $\phi_G(S^*) = \phi(G)$.

Let $B = (b_{i,j})_{1 \leq i,j \leq n}$ be the transition-probability matrix of a random walk on G, given by $b_{i,i} = 0$ for all i and $b_{i,j} = a_{i,j}/d_i$ for all $i \neq j$. We have that $-1 \leq \lambda_i(B) \leq 1$ for all i, $\lambda_1(B) = 1$, and $\mathbf{v}_1(B) = [\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}]^\intercal$.

Observe that when I is the identity matrix, the matrix $I - B$ is referred to as the *random walk normalized Laplacian matrix* (Von Luxburg, 2007). The eigenvectors of $I - B$ are identical to those of B. More specifically, it is known that, for all i, $\mathbf{v}_i(I - B) = \mathbf{v}_{n-i}(B)$.

The spectral clustering algorithm (Shi & Malik, 2000) computes the eigenvector $\mathbf{v}_2(B) = [\nu_1, \ldots, \nu_n]^\intercal$, and then produces the cut $S' = \{v_i : \nu_i > 0\}$ as the clustering result. Since $\phi_G(S') \leq 2\sqrt{\phi_G(S^*)}$ (Alon, 1986), it is established that the cut produced by the spectral clustering algorithm achieves a low conductance. Additionally, according to (Peng et al., 2015), we have $d_{\mathrm{vol}}(S', S^*) = O\left(\frac{\phi(G)}{\lambda_3(B)} \cdot \mathrm{Vol}_G(V)\right)$, indicating that S' closely approximates S^* in a graph that is well-clustered.

The normalized Laplacian matrix $L = (\ell_{i,j})_{1 \leq i,j \leq n}$, defined by $\ell_{i,j} = -a_{i,j}/\sqrt{d_i \cdot d_j}$ for $i \neq j$ and $\ell_{i,i} = 1$, is commonly used in spectral clustering algorithms that aim to minimize the conductance. However, in this work, we opt for the random walk normalized Laplacian matrix, as calculating spectral clustering under the normalized Laplacian is more complex in the edge LDP setting. Notably, when the desired number of clusters is two, the results of spectral clustering using the random walk normalized Laplacian matrix are at least as good as those obtained with the normalized Laplacian matrix (Von Luxburg, 2007).

2.4. Power Iteration Clustering

While spectral clustering can produce a cut with a small cut-ratio, it requires computing the eigenvector $\mathbf{v}_2(B)$, which can be computationally expensive. To address this, the power iteration clustering algorithm (Lin & Cohen, 2010) offers a more efficient method for estimating the eigenvector, significantly reducing the computation time.

Let $\mathbf{x}$ be a vector of length n where each element is independently drawn from a Gaussian distribution. It is known that $\mathbf{x}$ can be expressed as $c_1\lambda_1(B)\mathbf{v}_1(B) + \cdots +$

$c_n\lambda_n(B)\mathbf{v}_n(B)$, where $c_1, \ldots, c_n$ are independent random variables also drawn from a Gaussian distribution. Therefore, for a sufficiently large T, applying B^T to $\mathbf{x}$ gives:

$$\begin{aligned} B^T\mathbf{x} &= c_1\lambda_1(B)^T\mathbf{v}_1(B) + \cdots + c_n\lambda_n(B)^T\mathbf{v}_n(B) \\ &= c_1 \cdot \left[\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}\right]^{\mathsf{T}} + c_2\lambda_2(B)^T\mathbf{v}_2(B) \\ &\quad + \cdots + c_n\lambda_n(B)^T\mathbf{v}_n(B). \end{aligned} \tag{2}$$

When $\lambda_3(B) \ll \lambda_2(B)$, the term $B^T\mathbf{x}$ is approximately:

$$B^T\mathbf{x} \approx c_1\left[\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}\right]^{\mathsf{T}} + c_2\lambda_2(B)^T\mathbf{v}_2(B), \tag{3}$$

meaning the order of elements in $B^T\mathbf{x}$ closely mirrors that of $\mathbf{v}_2(B)$. Therefore, clustering can be performed using $B^T\mathbf{x}$ instead of $\mathbf{v}_2(B)$, yielding results similar to those from the spectral clustering algorithm.

2.5. Assumptions

We assume that the input graph has the following properties:
(1) The minimum degree is at least $2\sqrt{n}\log^4 n$,
(2) There exists a constant g such that for all $i \geq 3$, $\lambda_i(B) + 1 \leq \frac{\lambda_2(B)+1}{g}$,
(3) There exists $\delta \approx 1$ and $\gamma < 1$ such that the components of $\mathbf{v}_2(B)$ satisfies $\left|\left\{i : |\nu_i| \geq \frac{\gamma}{\sqrt{n}}\right\}\right| \geq \delta \cdot n$, and
(4) The number of nodes n is larger than a constant C.

Assumption (1) The first assumption is essential for any graph clustering algorithm under edge LDP with a constant privacy budget. Protecting the connections of low-degree nodes requires adding so much noise that their contributions are obscured, resulting in unstable clustering outcomes for these nodes.

Assumption (2) The second assumption is a standard prerequisite for iterative spectral clustering algorithms, such as the one presented in (Boutsidis et al., 2015). This assumption ensures the convergence of the iterative process. A comprehensive technical explanation supporting this assumption is provided in (Boutsidis et al., 2015).

Assumption (3) We demonstrate in Appendix A that the third assumption holds when the graph is well-clustered and most nodes have a degree cluster close to the average degree of the cluster to which they belong.

Specifically, for a node i in cluster $A \subseteq V$, we show in Proposition A.1 that the value of ν_i exceeds $\frac{\sqrt{\sigma}\cdot c}{4} \cdot \sqrt{\frac{d_i}{n \cdot d(A)}} - 2\sqrt{\frac{\phi(G)}{1-\lambda_3(B)}}$, where $d(A)$ represents the average degree of nodes in cluster A, and $c, \sigma \in \mathbb{R}$ satisfy the condition that at least $c|A|$ nodes in cluster A have degrees not less than $\sigma \cdot d(A)$. If the graph is well-clustered, the term $\sqrt{\frac{\phi(G)}{1-\lambda_3(B)}}$ becomes small and can be neglected (Mukherjee & Suppakitpaisarn, 2023). Consequently, we conclude that when $d_i \geq \sigma d(A)$ and there are at least $c|A|$ nodes satisfying this condition, it follows that $\nu_i \geq \frac{\sigma c}{4} \cdot \frac{1}{\sqrt{n}}$. Moreover, if σ and c are constants, there exist at least $c|A|$ nodes i such that $\nu_i = \Omega\left(\frac{1}{\sqrt{n}}\right)$.

We observe that the graphs generated by the stochastic block model have this property. In addition to our mathematical proof in the appendix, it is empirically demonstrated in (Abbe et al., 2020) that most of the values in the eigenvectors is in $\Theta(1/\sqrt{n})$. Additionally, (Balakrishnan et al., 2011) shows that this assumption can be satisfied when B is a node similarity matrix with certain additional properties.

Assumption (4) The final assumption is a common requirement for most differentially private algorithms. A large user base typically allows the added noise, introduced to protect sensitive information, to average out in the results.

3. Our Algorithm

We describe our algorithm in Algorithm 1. One can notice that we almost have $\mathbf{x}^{(t)} = B \cdot \mathbf{x}^{(t-1)}$ and $\mathbf{x}^{(T)} = B^T \cdot \mathbf{x}^{(0)}$ by the calculation at Lines 6 - 7. The only five differences are as follows:

Difference 1: Addition of Laplace Noise We add Laplace noise in Line 6 to protect users' information. Later, we show in Section 4.2 that this noise satisfies the conditions of the edge-local Laplacian query (Definition 2.3). Furthermore, in Section 4.3, we demonstrate that when the minimum degree is sufficiently large, the magnitude of the Laplacian noise becomes negligible compared to other terms in the calculation in Line 6.

Difference 2: Minimum Degree Estimation When B is the normalized random walk Laplacian matrix, calculating $B \cdot \mathbf{x}^{(t-1)}$ does not require knowing the degrees of other nodes. This property simplifies computations within the edge LDP setting and is the main reason we select the normalized random walk Laplacian matrix over the normalized Laplacian matrix in our clustering algorithm.

On the other hands, to bound the sensitivity, which determines the scale of the Laplace noise in Line 6, we need a lower bound on the minimum degree of the graph G. This bound is computed in Line 2 of the algorithm, using degree estimates obtained in Line 1 of the mechanism. In Appendix C, We will show that the estimate in Line 2 overestimates the minimum degree with probability not larger than $\frac{1}{n^2}$ when $\zeta = \frac{1}{n}$. If the estimate exceeds the actual minimum degree, we add edges in Line 3 to ensure that the modified graph meets the estimated minimum degree. In Appendix C, we further show that the variable δ exceeds

Algorithm 1 Private Power Iteration Clustering

```
Input: Graph $G = (V, E)$ where $V = \{v_1, \ldots, v_n\}$ and its adjacency matrix is $A = (a_{i,j})_{1 \leq i,j \leq n}$, privacy budget $\epsilon$, number of iterations $T = \frac{2\log n}{\log g}$, clipping factor $\mathsf{c}$, and parameter $\zeta = \frac{1}{n}$
Output: A cut of $G$ denoted by $S \subset V$
1 [User $i$] Compute the degree of $v_i$, denoted by $d_i$. Broadcast $\tilde{d}_i \leftarrow d_i + \mathrm{Lap}(10/\epsilon)$ to all users and the server.
2 [Server] Calculate $\delta \leftarrow \min_i \tilde{d}_i - \frac{10}{\epsilon}\log\frac{n}{2\zeta}$. Broadcast $\delta$ to all users.
3 [User $i$] If $d_i < \delta$, randomly select $j$ such that $a_{i,j} = 0$, then set $a_{i,j} = 1$ and increment $d_i$ by one. Repeat this process until $d_i \geq \delta$.
4 [Server] Initiate the vector $\mathbf{x}^{(0)} = [x_1^{(0)}, \ldots, x_n^{(0)}]^\intercal$ where $x_i^{(0)}$ is chosen from the Gaussian distribution with expected value 0 and standard deviation 1. Broadcast the vector $\mathbf{x}^{(0)}$ to all users.
5 for $t = 1, \ldots, T$ do
6     [User $i$] Calculate $w_i^{(t)} = \frac{1}{2}x_i^{(t-1)} + \frac{1}{2}\sum_j a_{i,j}\frac{x_j^{(t-1)}}{d_i} - \frac{1}{n}\sum_j x_j^{(t-1)} + \mathrm{Lap}\left(\frac{5\cdot T}{9\cdot\epsilon}\max_j\frac{|x_j^{(t-1)}|}{\delta}\right)$.
7     [User $i$] Let $U = \mathsf{c}\cdot\frac{5\cdot T}{9\cdot\epsilon}\max_j\frac{|x_j^{(t-1)}|}{\delta}$, also let $x_i^{(t)} = U$ if $w_i^{(t)} > U$, $x_i^{(t)} = -U$ if $w_i^{(t)} < -U$, and $x_i^{(t)} = w_i^{(t)}$ otherwise. Calculate and send $x_i^{(t)}$ to the server.
8     [Server] Aggregate the values $x_i^{(t)}$ into a vector $\mathbf{x}^{(t)} = [x_1^{(t)}, \ldots, x_n^{(t)}]^\intercal$, and broadcast this information to all users.
9 [Server] Return $S \leftarrow \{v_i : \mathbf{x}_i^{(T)} > 0\}$.
```

$\sqrt{n}\log^4 n$ with probability at least $1 - \frac{1}{n}$.

Difference 3: Replacing the Random Walk with a Lazy Random Walk Recall that all eigenvalues of the matrix B lie between 1 and -1. In certain networks, such as bipartite graphs, $\lambda_n(B)$ can be close to -1. This causes the final term in Equation (2) to oscillate, preventing the calculation of $B^T\mathbf{x}$ from converging. To address this, we propose replacing B with $W = \frac{1}{2}I + \frac{1}{2}B$. Note that for all i, $\mathbf{v}_i(W) = \mathbf{v}_i(B)$ and $\lambda_i(W) = \frac{1}{2}\lambda_i(B) + \frac{1}{2}$. Consequently, for all i, $0 \leq \lambda_i(W) \leq 1$. By the second assumption in Section 2.5, which is $\lambda_i(W) \leq \frac{\lambda_2(W)}{g}$ for all $i \geq 3$, we can have the approximation (3) even when some $\lambda_i(B)$ are negative. This modification leads to the first two terms of the calculation in Line 6.

Difference 4: Elimination of the Leading Eigenvector Recall Equation (2). Since $\lambda_2(W) < 1$, the term $\alpha_2\lambda_2(W)^T\mathbf{v}_2(W)$ diminishes compared to the leading term as T increases. On the other hand, the size of the Laplace noise added depends on the largest element of $\mathbf{x}^{(t-1)}$, which is determined by the leading term. Hence, for larger T, the noise magnitude dominates over the term $\alpha_2\lambda_2(W)^T\mathbf{v}_2(W)$. This causes $\mathbf{x}^{(T)}$ to deviate significantly from $\mathbf{v}_2(W)$, reducing the accuracy of the results.

To address this, we introduce the matrix $\tilde{W} = (\tilde{w}_{i,j})_{1 \leq i,j \leq n}$, where $\tilde{w}_{i,j} = w_{i,j} - 1/n$ for all i, j. We show in Appendix B that for all $i \geq 1$, $\lambda_i(\tilde{W}) = \lambda_{i+1}(W)$ and $\mathbf{v}_n(\tilde{W}) = \mathbf{v}_1(W)$. Additionally, $\mathbf{v}_n(\tilde{W}) = \mathbf{v}_1(W) = [\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}]^\intercal$ and $\lambda_n(\tilde{W}) = 0$.

With this update, the leading term $\alpha_1 \cdot [\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}}]^\intercal$ from (2) is eliminated. The term $\alpha_2\lambda_2(\tilde{W})^T\mathbf{v}_2(\tilde{W})$ now becomes the leading term, and we can ensure that the Laplace noise (the fourth term of Line 6 in Algorithm 1) is substantially smaller than the new leading term. The subtraction of the third term in the calculation at Line 6 reflects the update from W to $\tilde{W}$.

Difference 5: Clipping At Line 6 of the algorithm, we apply a standard clipping method commonly used in various LDP studies, such as (Imola et al., 2022) and (Betzer et al., 2024). We notice from the proof of Lemma D.6 that when the clipping factor c satisfies $\mathsf{c} \geq \log n \cdot \log g$, it holds with high probability that $-U \leq w_i^{(t)} \leq U$ for all i and t. Consequently, the clipping has no impact on our theoretical results. However, in our experiments, we observed that Algorithm 1 achieves optimal performance when c is set to 5, which is smaller than $\log n \cdot \log g$.

4. Properties of Our Algorithm

4.1. Efficiency

Computation Time The primary computational bottleneck of Algorithm 1 occurs in Line 6. In this step, the per-node computational complexity for each iteration is $O(n)$. To achieve accurate results, the required number of iterations T is given by $2\frac{\log n}{\log g} = \Theta(\log n)$, leading to an overall computational complexity of $O(n\log n)$ per node. In contrast, the central server has minimal computational demands. Its responsibilities are limited to generating the initial vector, receiving calculation results, and distributing them to all users.

Communication Cost While each user uploads only one real number $x_i^{(t)}$ to the server at each iteration, they must download the entire vector $\mathbf{x}^{(t)}$ in Line 8 of the algorithm. This results in a total communication cost of $O(n\log n)$ for each user.

Memory Consumption During iteration t, the central server and all users only need to store two vectors: $\mathbf{x}^{(t-1)}$

and $\mathbf{x}^{(t)}$. As a result, the memory consumption for all parties is $O(n)$. This is a significant improvement compared to the randomized response method. Even for sparse input graphs, the randomized response mechanism flips each relationship with a constant probability, leading to a graph with $\Omega(n^2)$ edges. Storing such a graph, with $\Omega(n^2)$ edges, requires a prohibitive amount of memory on the server, making it infeasible to design an LDP algorithm for large input graphs (Imola et al., 2022). In contrast, our approach requires only $O(n)$ memory, enabling our algorithms to handle input graphs with millions of nodes efficiently.

4.2. Privacy

The following theorem discuss our algorithm's privacy.

Theorem 4.1. *Algorithm 1 is ϵ-edge LDP.*

Proof. We perform $T+1$ edge-local Laplacian queries to all users: one at Line 1 and T queries at Line 6. At Line 1, the degree d_i has a sensitivity of one. Since the Laplace noise is set to $10/\epsilon$, the privacy budget for the publication at Line 1 is $\epsilon/10$.

When any $a_{i,j}$ is changed, the value of $x_i^{(t)}$ calculated at Line 6 changes by at most $\frac{1}{2}\max_j \frac{|x_j^{(t-1)}|}{d_j}$. Therefore, the sensitivity of the publication at Line 6 is $\frac{1}{2}\max_j \frac{|x_j^{(t-1)}|}{d_j} \leq \frac{1}{2}\max_j \frac{|x_j^{(t-1)}|}{\delta}$. The privacy budget for each publication at Line 6 is $\frac{9}{10}\cdot\frac{\epsilon}{T}$. Since there are T publications at Line 6, the total privacy budget of Algorithm 1 is $\frac{\epsilon}{10}+T\cdot\frac{9}{10}\cdot\frac{\epsilon}{T}=\epsilon$. □

4.3. Precision

In this section, we analyze the precision of Algorithm 1. In particular, we demonstrate that the algorithm's results closely resemble those of the spectral clustering algorithm. We provide an outline of our proof sketch here, with the full proof details available in Appendix D.

In Algorithm 1, at iteration t we compute the vector $\mathbf{x}^{(t)} = [x_1^{(t)}, \ldots, x_n^{(t)}]^\intercal$. The output of the algorithm is $S_{\text{alg}} = \{v_i \mid x_i^{(T)} > 0\}$, where $T = \frac{2\log n}{\log g}$.

Let $\mathbf{v}_j(\tilde{W}) = [v_{j,1}, \ldots, v_{j,n}]^\intercal$ be the j'th eigenvector of $\tilde{W}$, and let $c_1, \ldots, c_n \in \mathbb{R}$ be coefficients such that $\mathbf{x}^{(0)} = \sum_{j=1}^{n} c_j \mathbf{v}_j(\tilde{W})$. Additionally, for all t, suppose the noise added during iteration t of the algorithm is $\mathbf{y}^{(t)}$, and that $e_1^{(t)}, \ldots, e_n^{(t)} \in \mathbb{R}$ are coefficients such that $\mathbf{y}^{(t)} = \sum_{j=1}^{n} e_j^{(t)} \mathbf{v}_j(\tilde{W})$. In Lemma D.1, we show that $x_i^{(T)} = \sum_{j=1}^{n} \tilde{c}_j v_{j,i}$, where $\tilde{c}_j$ is given by $\tilde{c}_j = c_j \lambda_j(\tilde{W})^T + \sum_{t=1}^{T} e_j^{(t)} \lambda_j(\tilde{W})^{T-t}$.

In Lemma D.6, we show that the noise generated at Line 6 of the algorithm has a small scale. Specifically, we demonstrate that the noise scale, given by $\frac{5T}{9\epsilon}\max_j \frac{|x_j^{(t-1)}|}{\delta}$, is negligible compared to the magnitude of $\mathbf{x}^{(t)}$. Consequently, the noise term $\mathbf{y}^{(t)}$ does not dominate the calculation. This emphasizes the significance of removing the leading eigenvector and establishing a lower bound for the minimum degree δ.

Due to the lemma, the term $\sum_{t=1}^{T} e_j^{(t)} \lambda_j(\tilde{W})^{T-t}$ is negligible compared to $c_j \lambda_j(\tilde{W})^T$, and we have $\tilde{c}_i \approx c_i \lambda_i(\tilde{W})^T$. Consequently, $x_i^{(T)} \approx \sum_{j=1}^{n} c_j \lambda_j(\tilde{W})^T v_{j,i}$. Using techniques from (Boutsidis et al., 2015), we show that $x_i^{(T)} \approx c_1 \lambda_1(\tilde{W})^T v_{1,i}$ when $\lambda_j(\tilde{W}) \leq \frac{\lambda_1(\tilde{W})}{g}$ for all $j \geq 2$. Specifically, in Theorem D.7, we demonstrate that $\left|c_1 \lambda_1(\tilde{W})^T v_{1,i}\right| > \left|\sum_{t=1}^{T} e_1^T \lambda_1(\tilde{W})^{T-t} v_{1,i} + \sum_{j=2}^{n} \tilde{c}_j v_{j,i}\right|$ with probability at least $0.95 - o(1)$. The term $c_1 \lambda_1(\tilde{W})^T v_{1,i}$ dominates and determines the sign of $x_i^{(T)}$.

Since $\lambda_1(\tilde{W})^T$ is positive, we conclude that when $c_1 v_{1,i} > 0$, $x_i^{(t)} > 0$ with high probability. Recall that the outcome of the spectral clustering algorithm is $S_{\text{orig}} = \{v_i : v_{1,i} > 0\}$. Thus, when $c_1 > 0$, the result S_{alg} closely resembles S_{orig} with high probability. Conversely, when $c_1 < 0$, the result S_{alg} is similar to $V \setminus S_{\text{orig}}$ with high probability. Therefore, our algorithm is likely to produce a small $d_{\text{vol}}(S_{\text{alg}}, S_{\text{orig}})$. In conclusion, the results are comparable to those obtained from the spectral clustering algorithm.

5. Experimental Results

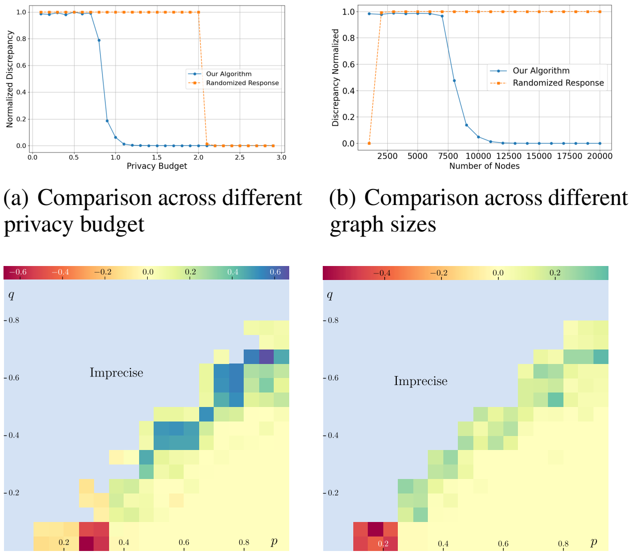
Evaluation Method For all experiments, we use the normalized discrepancy d_{norm}, as defined in (1), to assess precision. Remember that when the normalized discrepancy is small, the outcome closely resembles that of the original spectral clustering algorithm, indicating a high-quality clustering result. The reported values represent the average of 10 experiments, which we consider sufficient, as the variance in precision across each set of experiments is typically small.

Input Graphs We conduct most of our experiments on graphs generated using the stochastic block model (SBM) (Holland et al., 1983). This model is chosen because it ensures that the generated graphs are well-clustered and consist of exactly two clusters. Furthermore, SBM has been widely employed in prior studies to analyze spectral clustering under local differential privacy (Hehir et al., 2022). In this model, the set of n nodes is divided into two clusters of sizes n_1 and n_2, where $n_1 + n_2 = n$. Two nodes within the same cluster are connected with probability p, while nodes from different clusters are connected with probability q. While in most cases $p \gg q$, this paper also considers the scenario where $q > p$.

Parameters Unless otherwise specified, we set $n = 10,000$, $n_1 = n_2 = 5,000$, $p = 0.3$, $q = 0.2$, the clipping factor $\mathsf{c} = 10$, and the privacy budget $\epsilon = 1$.

The value of n is chosen to be $10,000$ due to the memory requirements of the benchmark algorithm, randomized response, which requires $\Omega(n^2)$ bits to store the entire graph for spectral clustering calculations. We believe that graphs of this size are sufficient to effectively demonstrate the empirical properties of our algorithm. Given the constraints of our local computational environment, handling larger graphs is not feasible. We select $p = 0.3$ and $q = 0.2$ because these values are close enough to highlight the precision of our algorithm in distinguishing clusters. We set the clipping factor $\mathsf{c} = 10$, as it is the integer closest to $\log n \log g$ for well-clustered graphs generated using the stochastic block model. Recall that, when $\mathsf{c} = \log n \log g$, the clipping is applied only with small probability. The privacy budget is set to $\epsilon = 1$ as it is a standard value commonly used in experiments of other local differential privacy algorithms (Hillebrand et al., 2023).

Benchmark To the best of our knowledge, only one graph clustering algorithm under local differential privacy has been explored in the literature. This algorithm employs the spectral clustering method on graph processed using randomized response (Hehir et al., 2022). Therefore, we select this algorithm as the benchmark for our study.



(a) Comparison across different privacy budget

(b) Comparison across different graph sizes

(c) Comparison across different graph density when $\epsilon = 1$

(d) Comparison across different graph density when $\epsilon = 1.5$

Figure 1. Comparison of the normalized discrepancy between our algorithm and the randomized response-based algorithm on the graphs generated from the stochastic block model. The results shown in Figures 1(c) and 1(d) represent the differences in the normalized discrepancies between the two algorithms.

Comparison across Different Privacy Budget As illustrated in Figure 1(a), our algorithm consistently outperforms the benchmark algorithm across all privacy budget values (ϵ). The improvement is especially notable in the range $0.8 \leq \epsilon \leq 2$, where the benchmark algorithm yields nearly random results, with a normalized discrepancy close to 1, while our algorithm produces results almost identical to the non-private spectral clustering.

Comparison across Different Graph Size Figure 1(b) presents a comparison with the benchmark algorithm across varying numbers of nodes (n). From the figure, we observe that while our algorithm performs poorly for small n, it achieves results identical to non-private spectral clustering when n becomes sufficiently large. This aligns with our theoretical findings, which indicate that the noise introduced by our algorithm becomes negligible as the input graph size increases.

The plot also reveals that the randomized response-based algorithm performs well only when the input graph size is small. This observation aligns with the theoretical findings of previous work (Mukherjee & Suppakitpaisarn, 2023), which state that the required privacy budget must exceed $\Theta(\log n)$. Consequently, larger values of n demand a higher privacy budget in the prior approach. In summary, our algorithm demonstrates greater precision for larger n, whereas the previous method performs better on very small graphs.

It is worth noting that, for the plot in Figure 1(b) alone, we conducted the experiment on Google Colaboratory. This was necessary because our local computing environment lacked the storage capacity for the randomized response results for graphs of that size. However, we have verified that the precision results remain consistent across different computational environments.

Comparison across Different Edge Density In Figures 1(c) and 1(d), we explore the impact of graph density by varying the probabilities p and q. The experiments are conducted for all pairs $(p, q) \in \{0.05, 0.1, \ldots, 0.95\}^2$ and for $\epsilon \in \{1, 1.5\}$. Due to the large number of experiments, the graph size is reduced to 1000 for this analysis. The results show that when $p > 0.35$, our algorithm consistently outperforms the randomized response-based method, achieving a smaller normalized discrepancy in these cases.

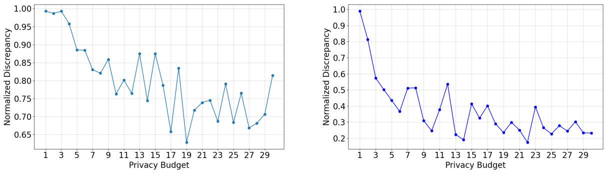
When $p \leq 0.35$, there are instances where our algorithm performs worse than the benchmark algorithm. This occurs because the estimated minimum degree, δ, is relatively small in these cases, resulting in a larger amount of noise added in Algorithm 1. While we have theoretically shown that our algorithm can produce results comparable to original spectral clustering when $\delta \geq \sqrt{n} \log^4 n$ (where n is the number of nodes), this analysis is valid only for large n and does not extend to cases where $n = 1000$. On the other hand, as shown in (Mohamed et al., 2022), the randomized response-based algorithm performs well when $q \leq p$ and p is small. Consequently, in these scenarios, the randomized response method outperforms our algorithm.

We observe that when $q > p$, the results of both algorithms deviate from those of the original spectral clustering algorithm. This outcome arises because the input graphs are not well-clustered, leading to poor performance from both the original spectral clustering method and the two algorithms in these cases.

Computation Time Although our algorithm is designed to be executed in a distributed manner in practice, we were unable to afford the necessary computation units for handling 10,000 nodes in this experiment. As a result, all computations were performed on our server, making the computation environment different from practical scenarios. Consequently, a direct comparison of the computation times between our algorithm and the benchmark algorithm is not feasible. However, even with all computations performed on the server, the computation time for graphs with 20,000 nodes is less than 10 seconds for both algorithms, and for graphs with 1,000,000 nodes, our algorithm completes in under 1 minute. Therefore, we consider computation time to be a manageable factor for both algorithms.

Results on Reddit Graph We also conduct an experiment on the real graph called Reddit graph (Hamilton et al., 2017). We chose this graph because it is one of the largest publicly available social networks and features a clear cluster structure. To ensure that the noise added in our algorithm is not too large, we calculate a 100-core and 500-core decomposition of the graph before giving it as an input of both algorithms. The 100-core decomposition result contains 154,525 nodes and 108,024,958 edges, while the 500-core decomposition result contains 44,586 nodes, 54,984,204 edges.

We were unable to run the randomized response algorithm on this large network, even with the A100 GPU (40GB of GPU RAM) and 83.5GB of system RAM. As a result, we could not directly compare our algorithm with the previous one. Since the Reddit graph contains more than two clusters, we observed that $\lambda_3(B) + 1$ is very close to $\lambda_2(B) + 1$, and the value of g (defined in Section 2.5) must be set as low as 0.005. Consequently, the number of iterations required by the algorithm, calculated as $2 \log n / \log g$, increases significantly to approximately 14,000. Given that the noise size is dependent on the number of iterations, this large iteration count renders the noise size unmanageable. To address this, we limited the number of iterations to 50 for this experiment.



(a) 100-core decomposition (b) 500-core decomposition

Figure 2. The normalized discrepancies of our algorithm for the graph extracted from the Reddit graph

Our results for these graphs are presented in Figure 2. For graphs generated using the SBM, we observe that when an algorithm fails to classify the graph in a particular setting, the normalized discrepancy exceeds 0.99. In contrast, our normalized discrepancy remains below 0.99 when the privacy budget is at least 4 for the 100-core decomposition and at least 1 for the 500-core decomposition. This demonstrates that our algorithm can produce meaningful clustering results under these conditions.

While the normalized discrepancy rapidly converges to 0 in graphs generated by the model, it does not converge to 0 in Figure 2. We attribute this to the Reddit graph containing more than two clusters, which results in a significant number of nodes v_i with small $|\nu_i|$ (as discussed in Assumption 3 in Section 2.5). Consequently, our algorithm is unable to classify these nodes correctly.

Further Experiments In Appendix E, we present experiments to validate the positive impact of the differences discussed in Section 3.

6. Conclusion and Future Work

In this paper, we propose a locally differentially private algorithm for graph clustering that is theoretically proven to work on general graphs. Unlike most prior works, which focus on non-interactive algorithms based on randomized response, we introduce an interactive algorithm leveraging power iterative clustering. Our approach demonstrates both theoretical and experimental improvements over previous methods. By this work, we believe that interactive algorithms have the potential to become a key tool for addressing graph problems under local differential privacy.

Although our algorithm is applicable to sparse graphs, our theoretical guarantees currently hold only for dense graphs. Extending the theory to sparse graphs requires an additional condition: for any eigenvector $\mathbf{v}_i = [v_{i,1}, \dots, v_{i,n}]^\intercal$, the ratio $\max_{j,j'} \frac{v_{i,j}}{v_{i,j'}}$ must be small. This property, known as delocalization, has been studied in several works, such as (Rudelson & Vershynin, 2016). We plan to investigate the potential of incorporating this property into our analysis.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Abbe, E., Fan, J., Wang, K., and Zhong, Y. Entrywise eigenvector analysis of random matrices with low expected rank. *Annals of Statistics*, 48(3):1452, 2020.

Alon, N. Eigenvalues and expanders. *Combinatorica*, 6(2): 83–96, 1986.

Apple's Differential Privacy Team. Learning with privacy at scale. *Apple Machine Learning Research*, 2017.

Balakrishnan, S., Xu, M., Krishnamurthy, A., and Singh, A. Noise thresholds for spectral clustering. *Advances in Neural Information Processing Systems*, 24, 2011.

Betzer, L., Suppakitpaisarn, V., and Hillebrand, Q. Publishing number of walks and Katz centrality under local differential privacy. In *UAI 2024*, 2024.

Boutsidis, C., Kambadur, P., and Gittens, A. Spectral clustering via the power method - provably. In *ICML 2015*, pp. 40–48, 2015.

Chen, H., Cohen-Addad, V., d'Orsi, T., Epasto, A., Imola, J., Steurer, D., and Tiegel, S. Private estimation algorithms for stochastic block models and mixture models. *Advances in Neural Information Processing Systems*, 36: 68134–68183, 2023.

Dwork, C. Differential privacy: A survey of results. In *TAMC 2008*, pp. 1–19, 2008.

Epasto, A., Liu, Q. C., Mukherjee, T., and Zhou, F. The power of graph sparsification in the continual release model. *arXiv preprint arXiv:2407.17619*, 2024.

Erlingsson, U., Pihur, V., and Korolova, A. RAPPOR: Randomized aggregatable privacy-preserving ordinal response. In *SIGSAC 2014*, pp. 1054–1067, 2014.

Fu, N., Ni, W., Zhang, S., Hou, L., and Zhang, D. GC-NLDP: A graph clustering algorithm with local differential privacy. *Computers & Security*, 124:102967, 2023.

Hamilton, W., Ying, Z., and Leskovec, J. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30, 2017.

He, W., Fichtenberger, H., and Peng, P. A differentially private clustering algorithm for well-clustered graphs. In *ICLR 2024*, 2024.

Hehir, J., Slavkovic, A., and Niu, X. Consistent spectral clustering of network block models under local differential privacy. *Journal of Privacy and Confidentiality*, 12 (2), 2022.

Henzinger, M., Sricharan, A., and Zhu, L. Tighter bounds for local differentially private core decomposition and densest subgraph. *arXiv preprint arXiv:2402.18020*, 2024.

Hillebrand, Q., Suppakitpaisarn, V., and Shibuya, T. Unbiased locally private estimator for polynomials of laplacian variables. In *SIGKDD 2023*, pp. 741–751, 2023.

Hillebrand, Q., Suppakitpaisarn, V., and Shibuya, T. Cycle counting under local differential privacy for degeneracy-bounded graphs. *arXiv preprint arXiv:2409.16688*, 2024.

Holland, P. W., Laskey, K. B., and Leinhardt, S. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

Imola, J., Murakami, T., and Chaudhuri, K. Locally differentially private analysis of graph statistics. In *USENIX Security 2021*, pp. 983–1000, 2021.

Imola, J., Murakami, T., and Chaudhuri, K. Communication-efficient triangle counting under local differential privacy. In *USENIX Security 2022*, pp. 537–554, 2022.

Imola, J., Epasto, A., Mahdian, M., Cohen-Addad, V., and Mirrokni, V. Differentially private hierarchical clustering with provable approximation guarantees. In *ICML 2023*, pp. 14353–14375, 2023.

Ji, T., Luo, C., Guo, Y., Wang, Q., Yu, L., and Li, P. Community detection in online social networks: A differentially private and parsimonious approach. *IEEE Transactions on Computational Social Systems*, 7(1):151–163, 2020.

Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S., and Smith, A. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

Li, J. and Tkocz, T. Tail bounds for sums of independent two-sided exponential random variables. In *High Dimensional Probability IX: The Ethereal Volume*, pp. 143–154. Springer, 2023.

Lin, F. and Cohen, W. W. Power iteration clustering. In *ICML 2010*, pp. 655–662, 2010.

Mangat, N. S. An improved randomized response strategy. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(1):93–95, 1994.

Mohamed, M. S., Nguyen, D., Vullikanti, A., and Tandon, R. Differentially private community detection for stochastic block models. In *ICML 2022*, pp. 15858–15894, 2022.

Mohar, B. Isoperimetric numbers of graphs. *Journal of Combinatorial Theory, Series B*, 47(3):274–291, 1989.

Mukherjee, S. and Suppakitpaisarn, V. Robustness for spectral clustering of general graphs under local differential privacy. *arXiv preprint arXiv:2309.06867*, 2023.

Ng, A., Jordan, M., and Weiss, Y. On spectral clustering: Analysis and an algorithm. *NIPS 2001*, 14, 2001.

Peng, R., Sun, H., and Zanetti, L. Partitioning well-clustered graphs: Spectral clustering works! In *COLT 2015*, pp. 1423–1455, 2015.

Qin, Z., Yu, T., Yang, Y., Khalil, I., Xiao, X., and Ren, K. Generating synthetic decentralized social graphs with local differential privacy. In *CCS 2017*, pp. 425–438, 2017.

Rudelson, M. and Vershynin, R. No-gaps delocalization for general random matrices. *Geometric and Functional Analysis*, 26(6):1716–1776, 2016.

Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

Von Luxburg, U. A tutorial on spectral clustering. *Statistics and Computing*, 17:395–416, 2007.

Wang, Y., Wu, X., and Hu, D. Using randomized response for differential privacy preserving data collection. In *EDBT/ICDT Workshops*, 2016.

Warner, S. L. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association*, 60(309):63–69, 1965.

Ye, Q., Hu, H., Au, M. H., Meng, X., and Xiao, X. Towards locally differentially private generic graph metric estimation. In *ICDE 2020*, pp. 1922–1925, 2020.

Zhang, H., Latif, S., Bassily, R., and Rountev, A. Differentially-private control-flow node coverage for software usage analysis. In *USENIX Security 2020*, 2020.

Zhu, T., Li, G., Zhou, W., and Philip, S. Y. Differentially private data publishing and analysis: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 29 (8):1619–1638, 2017.

A. Eigenvector Components

In this section, we analyze the Laplacian matrix of the graph G, defined as $L = I - B$. For each i, let $\lambda_i(L) = 1 - \lambda_i(B)$. It follows that $\lambda_i(L)$ is an eigenvalue of L, and the eigenvalues are ordered as $\lambda_1(L) \leq \cdots \leq \lambda_n(L)$. Moreover, the eigenvector $\mathbf{v}_i(B)$ associated with $\lambda_i(B)$ is also an eigenvector of L corresponding to $\lambda_i(L)$. For simplicity, throughout this section, we denote $\lambda_i(L)$ by λ_i and $\mathbf{v}_i(B)$ by $\mathbf{v}_i = [v_{i,1}, \ldots, v_{i,n}]^\intercal$.

Proposition A.1. *Assume that*

(i) Let $V(G) = A \sqcup B$ be a bipartition of G with $v_{2,j} \geq 0$ for $v_j \in A$, $v_{2,j} \leq 0$ for $v_j \in B$. Then, the cut (A, B) has conductance ϕ satisfying $\phi/\lambda_3 \leq 0.12$.

(ii) Let ϵ and c be a constant. For a subset $S \subseteq V$ and vertex $v_j \in S$, let us call v_j to be (ϵ, S)-average if $d_j \geq \epsilon d(S)$, where $d(S) = \mathrm{Vol}(S)/|S|$ is the average degree of the vertices in S. Let A_ϵ and B_ϵ denote the set of (ϵ, A)-average nodes of A and (ϵ, B)-average nodes of B, respectively. Assume that $|A_\epsilon| \geq c|A|$ and $|B_\epsilon| \geq c|B|$.

Then,

$$|v_{2,j}| \geq \begin{cases} \frac{\epsilon^{1/2} c}{4} \cdot \sqrt{\frac{d_j}{nd(A)}} - 2\sqrt{\frac{\phi}{\lambda_3}}, & v \in A \\ \frac{\epsilon^{1/2} c}{4} \cdot \sqrt{\frac{d_j}{nd(B)}} - 2\sqrt{\frac{\phi}{\lambda_3}}, & v \in B \end{cases} \tag{4}$$

Consequently, for $v_j \in A_\epsilon \cup B_\epsilon$, which is at least c fraction of the vertices of G, we have

$$|v_{2,j}| \geq \frac{\epsilon c}{4} \cdot \frac{1}{\sqrt{n}} - 2\sqrt{\frac{\phi}{\lambda_3}}. \tag{5}$$

Proof. Let us define the normalized indicator variables

$$g_A(j) = \begin{cases} \frac{d_j^{1/2}}{\mathrm{Vol}(A)^{1/2}}, & v_j \in A \\ 0, & v_j \in B \end{cases} \text{ and } g_B(j) = \begin{cases} 0, & v_j \in A \\ \frac{d_j^{1/2}}{\mathrm{Vol}(B)^{1/2}}, & v_j \in B \end{cases}.$$

Let the vector $g_A = [g_A(1), \ldots, g_A(n)]^\intercal$, $g_B = [g_B(1), \ldots, g_B(n)]^\intercal$, and, for any vector $\mathbf{v}$, the Rayleign quotient of $\mathbf{v} = [x_1, \ldots, x_n]^\intercal$, denoted by $\mathcal{R}(\mathbf{v})$, is $\frac{\mathbf{v}^\intercal L \mathbf{v}}{\mathbf{v}^\intercal \mathbf{v}}$. We show the following regarding the Rayleigh quotients $\mathcal{R}_L(g_A)$ and $\mathcal{R}_L(g_B)$.

Claim A.2. $\phi \geq \max\{\mathcal{R}_L(g_A), \mathcal{R}_L(g_B)\}$.

Proof of Claim A.2. Observe that the Rayleigh quotient of L satisfies,

$$\mathcal{R}_L(\mathbf{v}) = \frac{\mathbf{v}^\intercal L \mathbf{v}}{\mathbf{v}^\intercal \mathbf{v}} = 1 - \frac{\sum_{i=1}^n \sum_{j=1}^n \frac{a_{ij}}{d_i} x_i x_j}{\sum_{i=1}^n x_i^2} = 1 - \frac{\sum_{\{i,j\} \in E} \left(\frac{1}{d_i} + \frac{1}{d_j}\right) x_i x_j}{\sum_{i=1}^n x_i^2}. \tag{6}$$

Since $\|g_A\|^2 = 1$, we have

$$\begin{aligned} \mathcal{R}_L(g_A) = 1 - \sum_{\{i,j\} \in E} \left(\frac{1}{d_i} + \frac{1}{d_j}\right) g_A(i) g_A(j) &= 1 - \sum_{\{i,j\} \in E(A)} \left(\frac{1}{d_i} + \frac{1}{d_j}\right) \cdot \frac{\sqrt{d_i d_j}}{\mathrm{Vol}(A)} \\ &\leq 1 - \sum_{\{i,j\} \in E(A)} \frac{2}{\mathrm{Vol}(A)} = \frac{\mathrm{Vol}(A) - 2e(A)}{\mathrm{Vol}(A)} \\ &= \frac{e(A,B)}{\mathrm{Vol}(A)} \leq \phi. \end{aligned}$$

Similarly, we have $\mathcal{R}_L(g_B) \leq \phi$, completing the proof of Claim A.2. ■

For the rest of the proof, let us denote $t := \phi/\lambda_3$. Recall that $\mathbf{v}_1 = [1/\sqrt{n}, \ldots, 1/\sqrt{n}]^\intercal$. We will make use of the following lemmas from the structure theorem (Theorem 3.1) of (Peng et al., 2015), but with a different notation and error estimates.

Lemma A.3. *Let $\hat{g}_A$, $\hat{g}_B$ be the projections of g_A, g_B onto the space spanned by the first two eigenvectors $\{\mathbf{v}_1, \mathbf{v}_2\}$ of L. Then,*

$$\max\{\|\hat{g}_A - g_A\|^2, \|\hat{g}_B - g_B\|^2\} \le t. \tag{7}$$

Proof of Lemma A.3. Let $\mathbf{v}_3, \dots, \mathbf{v}_n$ be normalized eigenvectors of $\lambda_3, \dots, \lambda_n$ of L. Say $g_A = \alpha_1\mathbf{v}_1 + \cdots + \alpha_n\mathbf{v}_n$ and $g_B = \beta_1\mathbf{v}_1 + \cdots + \beta_n\mathbf{v}_n$ are representations of g_A and g_B in the L-eigenbasis. Clearly $\hat{g}_A = \alpha_1\mathbf{v}_1 + \alpha_2\mathbf{v}_2$ and $\hat{g}_B = \beta_1\mathbf{v}_1 + \beta_2\mathbf{v}_2$. Then, note that as $\mathbf{v}_i^\intercal\mathbf{v}_j = 0$ for every $i \ne j$,

$$\mathcal{R}_L(g_A) = \sum_{i=1}^n \alpha_i\mathbf{v}_i^\intercal \cdot L \cdot \sum_{i=1}^n \alpha_i\mathbf{v}_i = \sum_{i=1}^n \alpha_i^2\mathbf{v}_i^\intercal L\mathbf{v}_i = \sum_{i=1}^n \alpha_i^2\lambda_i.$$

But $\lambda_1 = 0$, leading us to $\mathcal{R}_L(g_A) \ge \alpha_2^2\lambda_2 + (\alpha_3^2 + \cdots + \alpha_n^2)\lambda_3 = \alpha_2^2\lambda_2 + \|\hat{g}_A - g_A\|^2\lambda_3 \ge \|\hat{g}_A - g_A\|^2\lambda_3$. Thus, $\|\hat{g}_A - g_A\|^2 \le \mathcal{R}_L(g_A)/\lambda_3 \le \phi/\lambda_3$ by Claim A.2. The proof for $\|\hat{g}_B - g_B\|^2$ is exactly analogous. ■

One of the main ideas used in (Peng et al., 2015) is that if $\hat{g}_A$ and $\hat{g}_B$ are independent, then $\text{Span}(\{\mathbf{v}_1, \mathbf{v}_2\}) = \text{Span}(\{\hat{g}_A, \hat{g}_B\})$, implying that $\mathbf{v}_1$ and $\mathbf{v}_2$ can be written as linear combinations of the projected indicator vectors $\hat{g}_A$ and $\hat{g}_B$, say $\mathbf{v}_2 = \eta_1\hat{g}_A + \eta_2\hat{g}_B$, implying that $\|\mathbf{v}_2 - \eta_1 g_A - \eta_2 g_B\|$ is small.

Let us now continue with the argument.

Claim A.4. $\hat{g}_A$ and $\hat{g}_B$ are linearly independent.

Proof of Claim A.4. By Lemma A.3, we have $\|\hat{g}_A\|^2 \ge 1 - t$ and $\|\hat{g}_B\|^2 \ge 1 - t$. On the other hand,

$$\begin{aligned}
|\langle \hat{g}_A, \hat{g}_B\rangle| &= |\langle \hat{g}_A - g_A + g_A, \hat{g}_B - g_B + g_B\rangle| \\
&\le |\langle \hat{g}_A - g_A, \hat{g}_B - g_B\rangle| + |\langle g_A, \hat{g}_B - g_B\rangle| + |\langle \hat{g}_A - g_A, g_B\rangle| \\
&\le \|\hat{g}_A - g_A\|\|\hat{g}_B - g_B\| + \|\hat{g}_A - g_A\| + \|\hat{g}_B - g_B\| \\
&\le t + 2\sqrt{t}.
\end{aligned} \tag{8}$$

Since $t \le 0.12 < \frac{1}{2}(2 - \sqrt{3})$, we have $t + 2\sqrt{t} < 1 - t$, implying $|\langle \hat{g}_A, \hat{g}_B\rangle| < \|\hat{g}_A\|\|\hat{g}_B\|$. As this implies a strict inequality in the Cauchy-Schwarz inequality, we have $\hat{g}_A \nparallel \hat{g}_B$. ■

As discussed earlier, Claim A.4 implies that there exist $\eta_1, \eta_2 \in \mathbb{R}$ such that $\mathbf{v}_2 = \eta_1\hat{g}_A + \eta_2\hat{g}_B$. Suppose $\mathbf{v}_2' = \eta_1 g_A + \eta_2 g_B$, and $\eta = \|\mathbf{v}_2'\| = \sqrt{\eta_1^2 + \eta_2^2}$. Note that, using (8),

$$\begin{aligned}
1 = \|\mathbf{v}_2\|^2 &\ge \eta_1^2\|\hat{g}_A\|^2 + \eta_2^2\|\hat{g}_B\|^2 - 2|\eta_1\eta_2\langle \hat{g}_A, \hat{g}_B\rangle| \\
&\ge \eta_1^2(1 - t) + \eta_2^2(1 - t) - (\eta_1^2 + \eta_2^2)(t + 2\sqrt{t}) \\
&= \eta^2(1 - 2t - 2\sqrt{t}).
\end{aligned}$$

Moreover, since $t \le 0.12$, we have

$$\eta^2 \le \frac{1}{1 - 2t - 2\sqrt{t}} < 16. \tag{9}$$

Moreover, by the triangle inequality and Cauchy-Schwarz inequality,

$$\begin{aligned}
\|\mathbf{v}_2 - \mathbf{v}_2'\|^2 &= \|\eta_1(\hat{g}_A - g_A) + \eta_2(\hat{g}_B - g_B)\|^2 \\
&\le \left(|\eta_1|\sqrt{t} + |\eta_2|\sqrt{t}\right)^2 \\
&\le 2t\eta^2
\end{aligned} \tag{10}$$

Therefore, we have that

$$2\eta\langle \mathbf{v}_2, \tfrac{1}{\eta}\mathbf{v}_2'\rangle = 2\langle \mathbf{v}_2, \mathbf{v}_2'\rangle = \|\mathbf{v}_2\|^2 + \|\mathbf{v}_2'\|^2 - \|\mathbf{v}_2 - \mathbf{v}_2'\|^2 \ge 1 + \eta^2 - 2t\eta^2,$$

leading us to

$$\langle \mathbf{v}_2, \tfrac{1}{\eta}\mathbf{v}_2'\rangle \ge \tfrac{1+\eta^2}{2\eta} - \eta t \ge 1 - \eta t. \tag{11}$$

Basically, this means that $\mathbf{v}_2$ is closely aligned with the normalized vector $\frac{1}{\eta}\mathbf{v}_2'$. We now show a lemma that relates the components of two such vectors.

Lemma A.5. *Let* $\mathbf{v} = [u_1, \dots, u_n]^\intercal$ *be a unit eigenvector of* L *and* $\mathbf{v}' = [u_1', \dots, u_n']^\intercal$ *be any unit vector with* $\langle \mathbf{v}, \mathbf{v}' \rangle \geq 1 - \epsilon^2$ *for some* $\epsilon > 0$*. Then, for each* $1 \leq j \leq n$*, we have*

$$|u_j'| \leq |u_j| + \epsilon.$$

Proof of Lemma A.5. Let $\{\mathbf{v}, \mathbf{z}_1, \dots \mathbf{z}_{n-1}\}$ be a orthonormal basis of eigenvectors of L, and, for all i, let $\mathbf{z}_i = [z_{i,1}, \dots, z_{i,n}]^\intercal$. Since $\mathbf{v}' = \langle \mathbf{v}, \mathbf{v}' \rangle \cdot \mathbf{v} + \sum_{i=1}^{n-1} \langle \mathbf{v}', \mathbf{z}_i \rangle \cdot \mathbf{z}_i$, this implies that for any $1 \leq j \leq n$,

$$\begin{aligned} |u_j'| &\leq |\langle \mathbf{v}, \mathbf{v}' \rangle| |u_j| + \sum_{i=1}^{n-1} |\langle \mathbf{v}', \mathbf{z}_i \rangle| |z_{i,j}| \\ &\leq |u_j| + \left(\sum_{i=1}^{n-1} \langle \mathbf{v}', \mathbf{z}_i \rangle^2 \right)^{1/2} \left(\sum_{i=1}^{n-1} z_{i,j}^2 \right)^{1/2} \\ &\leq |u_j| + \epsilon, \end{aligned}$$

where the last step follows from the fact that $\sum_{i=1}^{n-1} \langle \mathbf{v}', \mathbf{z}_i \rangle^2 + \langle \mathbf{v}', \mathbf{v} \rangle^2 = \|\mathbf{v}'\|^2 = 1$, and $\sum_{i=1}^{n-1} z_{i,j}^2 + u_j^2 = 1$. ■

Hence, by virtue of Lemma A.5, (9) and (11), we obtain

$$|v_{2,j}| \geq \frac{1}{\eta} |v_{2,j}'| - \sqrt{\eta t} = \frac{1}{\eta} |\eta_1 g_A(j) + \eta_2 g_B(j)| - \sqrt{\eta t} = \begin{cases} \frac{|\eta_1|}{\eta} \cdot \frac{d_j^{1/2}}{\mathrm{Vol}(A)^{1/2}} - \sqrt{\eta t}, & v_j \in A \\ \frac{|\eta_2|}{\eta} \cdot \frac{d_j^{1/2}}{\mathrm{Vol}(B)^{1/2}} - \sqrt{\eta t}, & v_j \in B \end{cases} \tag{12}$$

Finally, we need to show that $\min\{|\eta_1|, |\eta_2|\} \geq \epsilon^{1/2} c$. For this part of the proof, we shall use the assumption (ii) of our proposition.

Claim A.6. $|\eta_1| \geq c \cdot \left(\frac{\epsilon |A|}{n} \right)^{1/2}$ and $|\eta_2| \geq c \cdot \left(\frac{\epsilon |B|}{n} \right)^{1/2}$.

Proof of Claim A.6. Recall from the proof of Lemma A.3 that $\hat{g}_A = \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2$ and $\hat{g}_B = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2$. These equations, along with $\mathbf{v}_2 = \eta_1 \hat{g}_A + \eta_2 \hat{g}_B$, allow us to solve exactly for η_1 and η_2 as,

$$\eta_1 = \frac{\beta_1}{\alpha_2 \beta_1 - \alpha_1 \beta_2} \text{ and } \eta_2 = \frac{-\alpha_1}{\alpha_2 \beta_1 - \alpha_1 \beta_2}.$$

First, we note that $|\alpha_2 \beta_1 - \alpha_1 \beta_2| \leq (\alpha_1^2 + \alpha_2^2)^{1/2} (\beta_1^2 + \beta_2^2)^{1/2} \leq \|g_A\| \|g_B\| = 1$, so it suffices to lower bound $|\alpha_1|$ and $|\beta_1|$. We have that:

$$\begin{aligned} |\alpha_1| = |\langle g_A, \mathbf{v}_1 \rangle| = \frac{1}{\sqrt{n}} \sum_{v_j \in A} \frac{d_j^{1/2}}{\mathrm{Vol}(A)^{1/2}} &\geq \frac{1}{\sqrt{n}} \sum_{v_j \in A_\epsilon} \frac{d_j^{1/2}}{(|A| d(A))^{1/2}} \\ &\geq \frac{1}{\sqrt{n}} \cdot |A_\epsilon| \cdot \left(\frac{\epsilon}{|A|} \right)^{1/2} \\ &\geq c \cdot \left(\frac{\epsilon |A|}{n} \right)^{1/2}. \end{aligned}$$

By a similar argument, we have $|\beta_1| \geq c \cdot \left(\frac{\epsilon |B|}{n} \right)^{1/2}$, finishing the proof of Claim A.6. ■

Claim A.6, (12) and $\eta \leq 4$ leads us to, for $v_j \in A$,

$$|v_{2,j}| \geq \frac{c}{4} \cdot \frac{\epsilon^{1/2} |A|^{1/2}}{n^{1/2}} \cdot \frac{d_j^{1/2}}{\mathrm{Vol}(A)^{1/2}} - 2\sqrt{t} = \frac{c \epsilon^{1/2}}{4} \cdot \sqrt{\frac{d_j}{n d(A)}} - 2\sqrt{t},$$

which proves the inequality (4) for $v_j \in A$. The argument for $v_j \in B$ is analogous.

Finally, the inequality (5) directly follows (4) via the definitions of A_ϵ and B_ϵ.

□

B. Elimination of the Leading Eigenvector

The following proposition shows that the third term in the calculation at Line 6 of Algorithm 1 eliminates the leading eigenvector of W. Consequently, the leading eigenvector of $\tilde{W}$ becomes the second eigenvector of W.

Proposition B.1. *Let $W = \frac{1}{2}(I+D^{-1}A)$ be the lazy random walk matrix for a graph on n vertices. Let $J = (\mathsf{j}_{i,j})_{1\le i,j\le n}$ be a matrix such that $\mathsf{j}_{i,j} = 1$ for all i, j. Define $\tilde{W} = W - \frac{1}{n}J$. Then, for all $i \geq 1$, $\lambda_i(\tilde{W}) = \lambda_{i+1}(W)$ and $\mathbf{v}_n(\tilde{W}) = \mathbf{v}_1(W)$. Additionally, $\mathbf{v}_n(\tilde{W}) = \mathbf{v}_1(W) = [\frac{1}{\sqrt{n}}, \dots, \frac{1}{\sqrt{n}}]^\intercal$ and $\lambda_n(\tilde{W}) = 0$.*

Proof. Recall that $\mathbf{v}_1(W) = \left[\frac{1}{\sqrt{n}}, \dots, \frac{1}{\sqrt{n}}\right]^\intercal$ and $\lambda_1(W) = 1$. We have:

$$\tilde{W} \cdot \mathbf{v}_1(W) \quad = \quad W \cdot \mathbf{v}_1(W) - \frac{1}{n}J\mathbf{v}_1(W) = \mathbf{v}_1(W) - \left[\frac{1}{\sqrt{n}}, \dots, \frac{1}{\sqrt{n}}\right]^\intercal = \mathbf{0}.$$

Therefore, the vector $\left[\frac{1}{\sqrt{n}}, \dots, \frac{1}{\sqrt{n}}\right]^\intercal$ is an eigenvector of $\tilde{W}$ with eigenvalue 0. Since 0 is the minimum eigenvalue of $\tilde{W}$, it follows that $\mathbf{v}_n(\tilde{W}) = \mathbf{v}_1(W)$ and $\lambda_n(\tilde{W}) = 0$.

Next, let us consider $\mathbf{v}_i(W)$ for $i \geq 2$. Since, $\mathbf{v}_i(W) \perp \mathbf{v}_1(W)$, we obtain that the sum of all elements in $\mathbf{v}_i(W)$ is zero. Thus,

$$\tilde{W}\mathbf{v}_i(W) = W\mathbf{v}_i(W) - \frac{1}{n}J\mathbf{v}_i(W) = \lambda_i(W)\mathbf{v}_i(W).$$

This implies that, for all $i \geq 2$, $\mathbf{v}_i(W)$ is also an eigenvector of $\tilde{W}$ with the same eigenvalue. Consequently, as the largest eigenvalue of W becomes the smallest eigenvalue of $\tilde{W}$, we have $\lambda_{i-1}(\tilde{W}) = \lambda_i(W)$ and $\mathbf{v}_{i-1}(\tilde{W}) = \mathbf{v}_i(W)$. □

C. Minimum Degree Estimation

We will now demonstrate that the value of δ computed in Line 2 of Algorithm 1 has a low probability of overestimating the minimum degree of the input graph. This implies that, with large probability, we do not need to modify the input graph in Line 3 of the algorithm.

Proposition C.1. *With probability at least $1 - \zeta$, we have $\delta < \min_i d_i$.*

Proof. We have $\delta > \min_i d_i$ only if there is $\tilde{d}_i$ such that $\tilde{d}_i - \frac{10}{\epsilon}\log\frac{n}{2\zeta} > d_i$. This implies that the value sampled from the Laplace distribution at Line 1, denoted by l_i is larger than $\frac{10}{\epsilon}\log\frac{n}{2\zeta}$. By the property of the Laplace distribution, for all i, we have that:

$$\Pr\left[\mathsf{l}_i > \frac{10}{\epsilon}\log\frac{n}{2\zeta}\right] \quad = \quad \frac{1}{2}\exp\left(-\frac{10}{\epsilon}\log\frac{n}{2\zeta}\Big/\frac{10}{\epsilon}\right) = \zeta/n.$$

Then, by the union bound, the probability that there is an index i such that $\mathsf{l}_i > \frac{10}{\epsilon}\log\frac{n}{2\zeta}$ is not greater than ζ. □

Suppose that $\zeta = \frac{1}{n}$. In the next proposition, we shown that $\delta \geq \sqrt{n}\log^4 n$ with large probability.

Proposition C.2. $\Pr[\delta < \sqrt{n}\log^4 n] \leq \frac{1}{2n}$.

Proof. In Line 2 of Algorithm 1, Laplacian noise with a scale of $\frac{10}{\epsilon}$ is added. It follows that $\tilde{d}_i < d_i - \frac{20}{\epsilon}\log n$ if the noise added to d_i is less than $-\frac{20}{\epsilon}$. This event occurs with probability

$$\frac{1}{2}\exp\left(-\frac{20/\epsilon \cdot \log n}{10/\epsilon}\right) = \frac{1}{2n^2}.$$

Using the union bound, we have:

$$\Pr\left[\min_i \tilde{d}_i < \min_i d_i - \frac{20}{\epsilon}\log n\right] \leq \Pr\left[\tilde{d}_i < d_i - \frac{20}{\epsilon}\log n \text{ for some } i\right] \leq \frac{1}{2n}.$$

Given that $\delta = \min_i \tilde{d}_i - \frac{10}{\epsilon} \log \frac{n}{2\zeta}$, and under the assumption in Section 2.5 that the minimum degree of the network is at least $2\sqrt{n}\log^4 n$, we can bound:

$$\Pr\left[\delta < \sqrt{n}\log^4 n\right] \leq \Pr\left[\delta < \min_i d_i - \frac{20}{\epsilon}\log n - \frac{10}{\epsilon}\log\frac{n}{2\zeta}\right] \leq \frac{1}{2n},$$

for sufficiently large n. □

D. Size of Laplace Noise

In this section, we analyze the effect of adding the Laplace noise at Line 6 of the algorithm. Let the noise added by the node i at the iteration t is $y_i^{(t)}$. Define the vector $\mathbf{y}^{(t)}$ as $[y_1^{(t)}, \ldots, y_n^{(t)}]^\intercal$. Also, for all i, t, let $e_i^{(t)}$ be a real number such that $\mathbf{y}^{(t)} = e_1^{(t)}\mathbf{v}_1(\tilde{W}) + \cdots + e_n^{(t)}\mathbf{v}_n(\tilde{W})$.

Let the initial vector denoted by $\mathbf{x}^{(0)} = c_1\mathbf{v}_1(\tilde{W}) + \cdots + c_n\mathbf{v}_n(\tilde{W})$, and the final vector is denoted by $\mathbf{x}^{(T)}$. We obtain the following lemma by the notation.

Lemma D.1. *Let $\tilde{c}_1, \ldots, \tilde{c}_n$ be numbers such that $\mathbf{x}^{(T)} = \tilde{c}_1\mathbf{v}_1(\tilde{W}) + \cdots + \tilde{c}_n\mathbf{v}_n(\tilde{W})$. We obtain that $\tilde{c}_i = c_i\lambda_i(\tilde{W})^T + e_i^{(1)}\lambda_i(\tilde{W})^{T-1} + \cdots + e_i^{(T)}$.*

Proof. To prove the statement, let $\mathsf{c}_i^{(t)} = c_i\lambda_i(\tilde{W})^t + e_i^{(1)}\lambda_i(\tilde{W})^{t-1} + \cdots + e_i^{(t)}$. We proceed by induction on t to show that, for all $t \geq 0$, $\mathbf{x}^{(t)} = \mathsf{c}_1^{(t)}\mathbf{v}_1(\tilde{W}) + \cdots + \mathsf{c}_n^{(t)}\mathbf{v}_n(\tilde{W})$. When $t = 0$, $\mathsf{c}_i^{(0)} = c_i$, so the statement holds directly by the definition of the notation. Assume the statement is true for $t-1$; that is, $\mathbf{x}^{(t-1)} = \mathsf{c}_1^{(t-1)}\mathbf{v}_1(\tilde{W}) + \cdots + \mathsf{c}_n^{(t-1)}\mathbf{v}_n(\tilde{W})$. Then, for $\mathbf{x}^{(t)}$, we have

$$\mathbf{x}^{(t)} = \tilde{W} \cdot \mathbf{x}^{(t-1)} + \mathbf{y}^{(t)}.$$

Expanding this using the induction hypothesis gives

$$\mathbf{x}^{(t)} = (\mathsf{c}_1^{(t-1)}\lambda_1(\tilde{W}) + e_1^{(t)})\mathbf{v}_1(\tilde{W}) + \cdots + (\mathsf{c}_n^{(t-1)}\lambda_n(\tilde{W}) + e_n^{(t)})\mathbf{v}_n(\tilde{W}).$$

Thus, we obtain $\mathbf{x}^{(t)} = \mathsf{c}_1^{(t)}\mathbf{v}_1(\tilde{W}) + \cdots + \mathsf{c}_n^{(t)}\mathbf{v}_n(\tilde{W})$, completing the induction. □

From now, let $\mathbf{v}_i(\tilde{W}) = [v_{i,1}, \ldots, v_{i,n}]^\intercal$. We will now calculate the size of each variable. Recall from Line 4 of Algorithm 1 that $x_i^{(0)}$ is sampled from the Gaussian distribution with expected value 0 and standard deviation 1.

Lemma D.2. *For each i, the variable c_i is a normal random variable with mean 0 and standard deviation 1. Furthermore, for $i \neq j$, c_i is independent to c_j.*

Proof. Since, for all i, the eigenvector $\mathbf{v}_i(\tilde{W})$ is a unit vector and $c_i = \langle \mathbf{x}^{(0)}, \mathbf{v}_i(\tilde{W})\rangle$, we have that $c_i = \sum_j v_{i,j}x_j^{(0)}$. Because c_i is a linear combination of normal random variables, c_i is a normal random variable. Furthermore,

$$\mathbb{E}[c_i] = v_{i,1}\mathbb{E}[x_1^{(0)}] + \cdots + v_{i,n}\mathbb{E}[x_n^{(0)}] = 0,$$

and

$$\mathrm{Var}(c_i) = v_{i,1}^2\mathrm{Var}[x_1^{(0)}] + \cdots + v_{i,n}^2\mathrm{Var}[x_n^{(0)}] = v_{i,1}^2 + \cdots v_{i,n}^2 = 1.$$

Since $\mathbf{v}_i(\tilde{W})$ is orthogonal to $\mathbf{v}_j(\tilde{W})$ for $i \neq j$, the coefficients c_i and c_j, which are the dot products of $\mathbf{x}^{(0)}$ with $\mathbf{v}_i(\tilde{W})$ and $\mathbf{v}_j(\tilde{W})$ respectively, are independent of each other. □

Next, we give analyze the variables $e_i^{(t)}$. We observe that, although the random variable is a linear combination of the Laplace variables $y_j^{(t)}$, it is not itself Laplace-distributed.

Lemma D.3. *For all t and i, we have $\mathbb{E}[e_i^{(t)}] = 0$. Furthermore, for all t and all $i \neq j$, $\mathrm{Cov}(e_i^{(t)}, e_j^{(t)}) = 0$.*

Proof. According to Line 6 of Algorithm 1, for all t and $i \neq j$, the variables $y_i^{(t)}$ and $y_j^{(t)}$ are independent, with $\mathbb{E}(y_i^{(t)}) = \mathbb{E}(y_j^{(t)}) = 0$ and $\mathrm{Var}(y_i^{(t)}) = \mathrm{Var}(y_j^{(t)})$. The variable $e_i^{(t)}$ is defined as the dot product between $\mathbf{v}_i(\tilde{W})$ and $\mathbf{y}^{(t)}$. Specifically, if $\mathbf{v}_i(\tilde{W}) = [v_{i,1}, \ldots, v_{i,n}]^\intercal$, then $e_i^{(t)} = \sum_j v_{i,j} y_j^{(t)}$. Consequently, $\mathbb{E}(e_i^{(t)}) = \sum_j v_{i,j} \mathbb{E}[y_j^{(t)}] = 0$.

Next, for $i \neq j$, we examine the covariance between $e_i^{(t)}$ and $e_j^{(t)}$, denoted as $\mathrm{Cov}(e_i^{(t)}, e_j^{(t)})$. Since $\mathbb{E}(e_i^{(t)}) = \mathbb{E}(e_j^{(t)}) = 0$, $\{y_1^{(t)}, \ldots, y_n^{(t)}\}$ are independent with mean 0, and $\mathbf{v}_i$ is orthogonal to $\mathbf{v}_j$, we have:

$$\begin{aligned}\mathrm{Cov}(e_i^{(t)}, e_j^{(t)}) = \mathbb{E}\left[\sum_{i',j'} v_{i,i'} y_{i'}^{(t)} v_{j,j'} y_{j'}^{(t)}\right] = \sum_{i',j'} v_{i,i'} v_{j,j'} \mathbb{E}[y_{i'}^{(t)} y_{j'}^{(t)}] = \sum_k v_{i,k} v_{j,k} \mathbb{E}[(y_k^{(t)})^2] = \mathbb{E}[(y_1^{(t)})^2] \cdot \sum_k v_{i,k} v_{j,k} \\ = 0.\end{aligned}$$

□

Let C_t represent the scale of the Laplace noise in Line 6 during the t-th iteration of Algorithm 1. By definition, $\mathrm{Var}(y_i^{(t)}) = 2C_t^2$ for every i. The variance of $e_i^{(t)}$ is discussed in the following lemma. Our proof draws on ideas from the paper (Li & Tkocz, 2023).

Lemma D.4. *For all i and t, the variance of* $e_i^{(t)}$ *is* $2 \cdot C_t^2$. *Furthermore,* $\Pr[e_i^{(t)} \geq \sqrt{2} C_t \log n] \leq \frac{e}{n}$.

Proof. Based on the argument in the proof of Lemma D.3, we have $e_i^{(t)} = \sum_j v_{i,j} y_j^{(t)}$. Consequently, $\mathrm{Var}(e_i^{(t)}) = \sum_j v_{i,j}^2 \mathrm{Var}(y_j^{(t)})$ for all i and t. Since $y_j^{(t)}$ is a Laplace variable with scale C_t and each $\mathbf{v}_i(\tilde{W})$ is a unit vector, it follows that $\mathrm{Var}(e_j^{(t)}) = 2 \cdot C_t^2$.

Using the Chernoff bound and the moment generating function of the Laplacian distribution, we obtain that

$$\begin{aligned}\Pr[e_i^{(t)} \geq \sqrt{2} C_t \log n] \leq e^{-\log n} \cdot \mathbb{E}\left[\exp\left(\frac{\sum_j v_{i,j} y_j^{(t)}}{\sqrt{2} C_t}\right)\right] &= \frac{1}{n} \prod_j \mathbb{E}\left[\exp\left(\frac{v_{i,j} y_j^{(t)}}{\sqrt{2} C_t}\right)\right] \\ &= \frac{1}{n} \prod_j \mathbb{E}\left[\exp\left(\frac{v_{i,j}}{\sqrt{2}} \cdot \mathrm{Lap}\,(0,1)\right)\right] \\ &= \frac{1}{n} \prod_j \frac{1}{1 - \frac{1}{2} v_{i,j}^2} \\ &\leq \frac{1}{n} \exp \sum_j v_{i,j}^2 = \frac{e}{n}.\end{aligned}$$

□

Let h be a positive integer. We discuss the property of the vector $\tilde{W}^h \mathbf{y}^{(t)} := [\gamma_1^{(h,t)}, \ldots, \gamma_n^{(h,t)}]^\intercal$ in the next lemma.

Lemma D.5. *For all* i, h, t, *the probability that* $|\gamma_i^{(h,t)}| \geq 3\sqrt{2} \cdot \lambda_1(\tilde{W})^h \cdot C_t \cdot \log n$ *is at most* $2e/n^3$.

Proof. From the definition of $\gamma_i^{(h,t)}$ and the argument in Lemma D.1, we find that $\gamma_i^{(h,t)} = \sum_j \lambda_j(\tilde{W})^h \cdot v_{j,i} \cdot e_j^{(t)}$. According to Lemma D.3, $\mathrm{Cov}(e_j^{(t)}, e_{j'}^{(t)}) = 0$ for $j \neq j'$. Therefore, by Lemma D.4,

$$\mathrm{Var}(\gamma_i^{(h,t)}) = \sum_j \lambda_j(\tilde{W})^{2h} \cdot v_{j,i}^2 \cdot \mathrm{Var}(e_j^{(t)}) = 2C_t^2 \cdot \sum_j \lambda_j(\tilde{W})^{2h} \cdot v_{j,i}^2 \leq 2C_t^2 \cdot \lambda_1(\tilde{W})^{2h} \cdot \sum_j v_{j,i}^2 = 2C_t^2 \cdot \lambda_1(\tilde{W})^{2h}.$$

Since $e_j^{(t)}$ is a linear combination of Laplace variables, $\gamma_i^{(h,t)}$ is also a linear combination of the Laplace variable $y_j^{(t)}$. Let $\mathsf{a}_1, \ldots, \mathsf{a}_n$ be real numbers such that $\gamma_i^{(h,t)} = \sum_j \mathsf{a}_j y_j^{(t)}$. We obtain that $\mathrm{Var}(\gamma_i^{(h,t)}) = 2 \cdot C_t^2 \sum_j \mathsf{a}_j^2 \leq 2C_t^2 \cdot \lambda_1(\tilde{W})^{2h}$, and $\sum_j \mathsf{a}_j^2 \le \lambda_1(\tilde{W})^{2h}$. Using the Chernoff bound, we obtain that

$$\begin{aligned}
\Pr[\gamma_i^{(h,t)} \ge 3\sqrt{2}\cdot\lambda_1(\tilde{W})^h\cdot C_t\cdot\log n] &\le e^{-3\log n}\cdot\mathbb{E}\left[\exp\left(\frac{\gamma_i^{(h,t)}}{\sqrt{2}\cdot\lambda_1(\tilde{W})^h\cdot C_t}\right)\right]\\
&\le \frac{1}{n^3}\mathbb{E}\left[\exp\left(\frac{\sum_j \mathsf{a}_j\cdot\mathrm{Lap}(0,1)}{\sqrt{2}\cdot\lambda_1(\tilde{W})^h}\right)\right]\\
&= \frac{1}{n^3}\prod_j\frac{1}{1-\frac{\mathsf{a}_j^2}{2\lambda_1(\tilde{W})^{2h}}}\\
&\le \frac{1}{n^3}\exp\left(\frac{1}{2\lambda_1(\tilde{W})^{2h}}\sum_j \mathsf{a}_j^2\right)\le\frac{1}{n^3}\exp(1).
\end{aligned}$$

The lemma statement follows from the fact that the probability distribution of $\gamma_{h,t}$ is symmetric about 0. □

In the next lemma, we analyze the size of the noise added in the algorithm. Recall that the variable δ is the noisy minimum degree published at Line 2 of Algorithm 1. In Proposition C.2, we show that $\delta\ge\sqrt{n}\log^4 n$ with probability at least $1-\frac{1}{n}$. We denote the event that $\delta\ge\sqrt{n}\log^4 n$ by $\mathcal{E}_\delta$.

Lemma D.6. *Recall that C_t is the scale of the noise added at Line 6 of Algorithm 1. Then,*

$$\Pr\left[C_t\le\frac{10}{9\epsilon}\cdot\frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n}\ \textit{for all}\ 1\le t\le T\ \mid\mathcal{E}_\delta\right]\ge 1-\frac{8eT^2}{n^2}.$$

Proof. Since $x_i^{(0)}$ is drawn from a Gaussian distribution with mean 0 and standard deviation 1, it follows from the properties of a normal random variable that $\Pr[|x_i^{(0)}|\ge\log n\cdot\log g]\le\frac{1}{n^3}$. By applying the union bound, we then have $\Pr[\max_i|x_i^{(0)}|\ge\log n\cdot\log g]\le\frac{1}{n^2}$.

We will prove this lemma by induction on the number of iterations t. For $t=1$, recall from Line 6 of the algorithm that the noise $y_i^{(t)}$ is drawn from a Laplace distribution with scale parameter $\frac{5\cdot T}{9\cdot\epsilon}\cdot\frac{\max_i|x_i^{(t-1)}|}{\delta}$, where ϵ is the privacy budget and δ is the minimum degree of the input graph. In the event $\mathcal{E}_\delta$, the variable $\delta\ge\sqrt{n}\log^4 n$. Recall that we set $T=2\frac{\log n}{\log g}$ in our algorithm. Consequently, the noise scale in the first iteration is larger than $\frac{10}{9\epsilon}\frac{\log n}{\log g}\cdot\frac{\log n\cdot\log g}{\sqrt{n}\log^4 n}=\frac{10}{9\epsilon\cdot\sqrt{n}\log^2 n}$ with probability not larger than $1/n^2$ when n is large enough.

Next, assume that, in the event $\mathcal{E}_\delta$, with probability not smaller than $1-\frac{2e\cdot(2t-2)^2}{n^2}$, for all $t'<t$, the noise (denoted by $y_i^{(t')}$) is sampled from a Laplace distribution with a scale no more than $\frac{10}{9\epsilon}\cdot\frac{\lambda_1(\tilde{W})^{t'-1}}{\sqrt{n}\log^2 n}$. From our previous calculations, it follows that $\mathbf{x}^{(t)}=\tilde{W}^t\mathbf{x}^{(0)}+\tilde{W}^{t-1}\mathbf{y}^{(1)}+\cdots+\mathbf{y}^{(t)}$. Let $\tilde{W}^t\mathbf{x}^{(0)}=[\mathsf{x}_1^{(t)},\ldots,\mathsf{x}_n^{(t)}]^\intercal$ and, for all $t'\le t$, $\tilde{W}^{t-t'}\mathbf{y}^{(t')}=[\mathsf{y}_1^{(t,t')},\ldots,\mathsf{y}_n^{(t,t')}]^\intercal$. The value of $\max_i|x_i^{(t-1)}|$, which decides the noise scale of $\mathbf{y}^{(t)}$, is equal to $\max_i\left|\mathsf{x}_i^{(t-1)}+\sum_{t'=1}^{t-1}\mathsf{y}_i^{(t-1,t')}\right|$.

Let us first consider the vector $[\mathsf{x}_1^{(t-1)},\ldots,\mathsf{x}_n^{(t-1)}]^\intercal$. Recall that $\mathbf{v}_i(\tilde{W})=[v_{i,1},\ldots,v_{i,n}]^\intercal$. By the notation, we have $\mathsf{x}_i^{(t-1)}=\sum_j\lambda_j(\tilde{W})^{t-1}v_{j,i}c_j$.

Since, by Lemma D.2, c_j and $c_{j'}$ are independent for $j\ne j'$, we obtain:

$$\begin{aligned}
\mathbb{E}[\mathsf{x}_i^{(t-1)}] &= \sum_j\lambda_j(\tilde{W})^{t-1}v_{j,i}\cdot\mathbb{E}[c_j]=0,\\
\mathrm{Var}[\mathsf{x}_i^{(t-1)}] &= \sum_j\lambda_j(\tilde{W})^{2t-2}v_{j,i}^2\mathrm{Var}[c_j]\le\lambda_1(\tilde{W})^{2t-2}\mathrm{Var}\left[\sum_j v_{j,i}c_j\right]=\lambda_1(\tilde{W})^{2t-2}\mathrm{Var}\left[x_i^{(0)}\right]=\lambda_1(\tilde{W})^{2t-2}.
\end{aligned}$$

Also, since $\mathrm{x}_i^{(t-1)}$ is a linear combination of the normal random variable c_j, we can conclude that $\mathrm{x}_i^{(t-1)}$ is also normal. By the property of the normal variable, we have $\Pr\left[|\mathrm{x}_i^{(t-1)}| \geq \frac{1}{2}\log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1}\right] \leq \frac{1}{n^3}$ for all i. By the union bound, $\Pr\left[\max_i |\mathrm{x}_i^{(t-1)}| \geq \frac{1}{2}\log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1}\right] \leq \frac{1}{n^2}$.

Let us reconsider the variable $\gamma_i^{(h,t)}$ from Lemma D.5. Note that $\mathrm{y}_i^{(t-1,t')} = \gamma^{(t-t'-1,t')}$. Let $\mathcal{E}$ denote the event that $C_{t'} \leq \frac{10}{9\epsilon} \cdot \frac{\lambda_1^{t'-1}(\tilde{W})}{\sqrt{n}\log^2 n}$ for all $t' < t$. In the event $\mathcal{E}$ and $\mathcal{E}_\delta$, Lemma D.5 implies that, for all i, t, t',

$$\left|\mathrm{y}_i^{(t-1,t')}\right| \quad \geq \quad 3\sqrt{2} \cdot \lambda_1^{t-t'-1}(\tilde{W}) \cdot \frac{10}{9\epsilon} \cdot \frac{\lambda_1^{t'-1}(\tilde{W})}{\sqrt{n}\log^2 n} = \frac{30\sqrt{2}}{9\epsilon} \cdot \frac{\lambda_1^{t-2}(\tilde{W})}{\sqrt{n}\log^2 n}$$

with probability at most $\frac{2e}{n^3}$.

By applying the union bound, we deduce that for all t, t',

$$\max_i |\mathrm{y}_i^{(t-1,t')}| \geq \frac{30\sqrt{2}}{9\epsilon} \cdot \frac{\lambda_1^{t-2}(\tilde{W})}{\sqrt{n}\log^2 n}$$

with probability at most $\frac{2e}{n^2}$. By Lemma 4.4 of (Mohar, 1989), we have that $\lambda_2(B) \geq 0$ and $\lambda_1(\tilde{W}) \geq \frac{1}{2}$. When n is sufficiently large, it follows that, for all t, t',

$$\max_i |\mathrm{y}_i^{(t-1,t')}| \geq \frac{\lambda_1^{t-1}(\tilde{W})\log g}{4\log n} \geq \frac{30\sqrt{2}}{9\epsilon}\frac{\lambda_1^{t-2}(\tilde{W})}{\sqrt{n}\log^2 n}$$

with probability at most $\frac{2e}{n^2}$. We finally obtain

$$\Pr\left[\sum_{t' \leq t-1} \max_i |\mathrm{y}_i^{(t-1,t')}| \geq \frac{1}{2} \cdot \lambda_1^{t-1}(\tilde{W}) \mid \mathcal{E}, \mathcal{E}_\delta\right] \leq \frac{2et}{n^2}.$$

Because, for all i and t, the variables $\mathrm{x}_i^{(t-1)}$ do not depends on the scale of the Laplacian noise and the event $\mathcal{E}$, we obtain that:

$$\begin{aligned}
& \Pr\left[\max_i \left|x_i^{(t-1)}\right| \geq \log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1} \mid \mathcal{E}, \mathcal{E}_\delta\right] \\
= \quad & \Pr\left[\max_i \left|\mathrm{x}_i^{(t-1)} + \sum_{t' \leq t-1} \mathrm{y}_i^{(t-1,t')}\right| \geq \log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1} \mid \mathcal{E}, \mathcal{E}_\delta\right] \\
\leq \quad & \Pr\left[\max_i \left|\mathrm{x}_i^{(t-1)}\right| + \sum_{t' \leq t-1} \max_i \left|\mathrm{y}_i^{(t-1,t')}\right| \geq \log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1} \mid \mathcal{E}, \mathcal{E}_\delta\right] \\
\leq \quad & \Pr\left[\max_i \left|\mathrm{x}_i^{(t-1)}\right| \geq \frac{1}{2}\log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1}\right] + \Pr\left[\sum_{t' \leq t-1} \max_i \left|\mathrm{y}_i^{(t-1,t')}\right| \geq \frac{1}{2}\lambda_1(\tilde{W})^{t-1} \mid \mathcal{E}, \mathcal{E}_\delta\right] \\
\leq \quad & \frac{(2et+1)}{n^2}.
\end{aligned}$$

In the event $\mathcal{E}$ and $\mathcal{E}_\delta$, $\max_i |x_i^{(t-1)}| \geq \log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1}$ with probability at most $\frac{2et+1}{n^2}$. In the event of $\mathcal{E}$ and $\mathcal{E}_\delta$, the noise scale at the iteration t, denoted by C_t, is at most $\frac{10}{9\epsilon}\frac{2\log n}{\log g}\frac{\log n \cdot \log g \cdot \lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^4 n} = \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n}$ with probability at least $1 - \frac{2et+1}{n^2}$. As a result,

$$\begin{aligned}\Pr\left[C_t \geq \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n} \text{ or } \bar{\mathcal{E}} \mid \mathcal{E}_\delta\right] &\leq \Pr\left[C_t \geq \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n} \text{ and } \mathcal{E} \mid \mathcal{E}_\delta\right] + \Pr[\bar{\mathcal{E}} \mid \mathcal{E}_\delta] \\ &\leq \Pr\left[C_t \geq \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n} \mid \mathcal{E}, \mathcal{E}_\delta\right] + \frac{2e(2t-2)^2}{n^2} \\ &\leq \frac{2et+1}{n^2} + \frac{2e(2t-2)^2}{n^2} \leq \frac{2e(2t)^2}{n^2}.\end{aligned}$$

This completes the induction step. We can conclude that, for all $t \in \{1, \ldots, T\}$, $C_{t'} \leq \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t'-1}}{\sqrt{n}\log^2 n}$ for all $t' \leq t$ with probability at least $1 - \frac{2e(2t)^2}{n^2}$ when $\delta \geq \sqrt{n}\log^4 n$. □

We will leverage the previous lemma to demonstrate that the outcome of Algorithm 1 closely aligns with the results obtained through spectral clustering. Recall Lemma D.1 that the final vector $x_i^{(T)} = \sum_{j=1}^n \tilde{c}_j v_{j,i}$ when $\tilde{c}_j = c_j \lambda_j(\tilde{W})^T + \sum_{t=1}^T e_j^{(t)} \lambda_j(\tilde{W})^{T-t}$.

Theorem D.7. *For any node i such that $|v_{1,i}| \geq \frac{\gamma}{\sqrt{n}}$. For large enough n, we obtain that*

$$\Pr\left[\left|c_1\lambda_1(\tilde{W})^T v_{1,i}\right| > \left|\sum_{t=1}^T e_1^T \lambda_1(\tilde{W})^{T-t} v_{1,i} + \sum_{j=2}^n \tilde{c}_j v_{j,i}\right|\right] \geq 0.95 - o(1).$$

Proof. We first obtain that

$$\begin{aligned}&\Pr\left[\left|c_1\lambda_1(\tilde{W})^T v_{1,i}\right| > \left|\sum_{t=1}^T e_1^T \lambda_1(\tilde{W})^{T-t} v_{1,i} + \sum_{j=2}^n \tilde{c}_j v_{j,i}\right|\right] \\ \geq\ &\Pr\left[\left|c_1\lambda_1(\tilde{W})^T v_{1,i}\right| > \left|\sum_{t=1}^T e_1^T \lambda_1(\tilde{W})^{T-t} v_{1,i}\right| + \left|\sum_{j=2}^n \tilde{c}_j v_{j,i}\right|\right] \\ \geq\ &\Pr\left[|v_{1,i}|\left(|c_1\lambda_1(\tilde{W})^T| - \left|\sum_{t=1}^T e_1^{(t)} \lambda_1(\tilde{W})^{T-t}\right|\right) > \left|\sum_{j=2}^n c_j v_{j,i} \lambda_j(\tilde{W})^T\right| + \left|\sum_{j=2}^n \sum_{t=1}^T e_j^{(t)} \lambda_j(\tilde{W})^{T-t} v_{j,i}\right|\right].\end{aligned}$$

Recall from Lemma D.2 that c_i is a normal random variable with mean 0 and standard deviation 1. We obtain that:

$$\Pr\left[\left|c_1\lambda_1(\tilde{W})^T\right| \geq \frac{\lambda_1(\tilde{W})^T}{16}\right] > 0.95. \tag{13}$$

Recall from Lemma D.4 that $\Pr\left[e_1^{(t)} \geq \sqrt{2}C_t \log n\right] \leq \frac{e}{n}$. Let $\mathcal{E}$ be the event that $\max_t C_t \leq \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n}$ and $\mathcal{E}_\delta$ be the event that $\delta \geq \sqrt{n}\log^4 n$. We obtain that $\Pr\left[e_1^{(t)} \geq \frac{10\sqrt{2}}{9\epsilon}\frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log n}|\mathcal{E}, \mathcal{E}_\delta\right] \leq \frac{e}{n}$. Denote $\sum_{t=1}^T e_j^{(t)} \lambda_1(\tilde{W})^{T-t}$ by η. By the union bound,

$$\Pr\left[\eta \geq \frac{\lambda_1(\tilde{W})^T}{32}|\mathcal{E}\right] \leq \Pr\left[\eta \geq \frac{10\sqrt{2}\cdot T}{9\epsilon}\frac{\lambda_1(\tilde{W})^{T-1}}{\sqrt{n}\log n}|\mathcal{E}\right] \leq \frac{eT}{n},$$

for sufficiently large n. Recall the event $\mathcal{E}_\delta$, which is the event when $\delta \geq \sqrt{n}\log^4 n$. By Lemma D.6, we know that

$\Pr[\mathcal{E} \mid \mathcal{E}_\delta] \geq 1 - \frac{8eT^2}{n^2}$. Also, by Proposition C.2, we know that $\Pr[\bar{\mathcal{E}}_\delta] \leq \frac{1}{2n}$. As a result, when n is large enough,

$$\begin{aligned}
\Pr\left[\eta \geq \frac{\lambda_1(\tilde{W})^T}{32}\right] &\leq \Pr\left[\eta \geq \frac{\lambda_1(\tilde{W})^T}{32} \text{ and } \mathcal{E} \text{ and } \mathcal{E}_\delta\right] + \Pr[\bar{\mathcal{E}} \text{ and } \mathcal{E}_\delta] + \Pr[\bar{\mathcal{E}}_\delta] \\
&\leq \Pr\left[\eta \geq \frac{\lambda_1(\tilde{W})^T}{32} \mid \mathcal{E}, \mathcal{E}_\delta\right] + \Pr[\bar{\mathcal{E}} \mid \mathcal{E}_\delta] + \frac{1}{2n} = o(1).
\end{aligned}$$

Since the distribution of η is symmetric around 0, we have that

$$\Pr\left[|\eta| \geq \frac{\lambda_1(\tilde{W})^T}{32}\right] = o(1). \tag{14}$$

By combining (13) and (14) and using the fact that $|v_{1,i}| \geq \frac{\gamma}{\sqrt{n}}$, we obtain that:

$$\Pr\left[|v_{1,i}|\left(|c_1\lambda_1(\tilde{W})^T| - |\eta|\right) \geq \frac{\gamma\lambda_1(\tilde{W})^T}{32\sqrt{n}}\right] > 0.95 - o(1) \tag{15}$$

By the assumption that $\lambda_2(\tilde{W}) \leq \frac{\lambda_1(\tilde{W})}{g}$, we have that $\left|\sum_{j=2}^n c_j v_{j,i}\lambda_j(\tilde{W})^T\right| \leq \left|\frac{\lambda_1(\tilde{W})^T}{g^T} \cdot \sum_{j=2}^n c_j\right|$. Since $\sum_{j=2}^n c_j v_{j,i}\lambda_j(\tilde{W})^T$ is a normal random variable with mean 0 and standard deviation at most $\frac{\sqrt{n}\cdot\lambda_1(\tilde{W})^T}{g^T}$, we have the following.

$$\Pr\left[\left|\sum_{j=2}^n c_j v_{j,i}\lambda_j(\tilde{W})^T\right| \geq \frac{\sqrt{n}\log n \cdot \lambda_1(\tilde{W})^T}{g^T}\right] < \frac{1}{n^2}.$$

When $T = \frac{2\log n}{\log g}$ and n is large enough, we obtain that $\frac{\sqrt{n}\log n}{g^T} < \frac{\gamma}{65\sqrt{n}}$. Hence,

$$\Pr\left[\left|\sum_{j=2}^n c_j v_{j,i}\lambda_j(\tilde{W})^T\right| \geq \frac{\gamma \cdot \lambda_1(\tilde{W})^T}{65\sqrt{n}}\right] < \frac{1}{n^2}. \tag{16}$$

Consider the summation $\sum_{j=2}^n \sum_{t=1}^T e_j^{(t)}\lambda_j(\tilde{W})^{T-t}v_{j,i}$. Denote the summation as β_t. By Lemma D.3, we know that $\mathbb{E}[e_j^{(t)}] = 0$ for all j, t. As a result, $\mathbb{E}[\beta_t] = \mathbb{E}\left[\sum_{j=2}^n e_j^{(t)}\lambda_j(\tilde{W})^{T-t}v_{j,i}\right] = 0$ for all t. Furthermore, by $\lambda_j(\tilde{W}) \leq \frac{\lambda_1(\tilde{W})}{g}$ for all $j \geq 2$,

$$\begin{aligned}
\mathrm{Var}\left[\beta_t\right] &\leq \sum_{j=2}^n \lambda_j(\tilde{W})^{2T-2t}v_{j,i}^2\mathrm{Var}(e_j^{(t)}) \\
&\leq \sum_{j=1}^n \frac{\lambda_1(\tilde{W})^{2T-2t}}{g^{2T-2t}}v_{j,i}^2\mathrm{Var}(e_j^{(t)}) \\
&= \frac{\lambda_1(\tilde{W})^{2T-2t}}{g^{2T-2t}}\mathrm{Var}\left[\sum_j v_{j,i}e_j^{(t)}\right] \\
&= \frac{\lambda_1(\tilde{W})^{2T-2t}}{g^{2T-2t}}\mathrm{Var}\left[y_i^{(t)}\right] \\
&= 2\frac{\lambda_1(\tilde{W})^{2T-2t}}{g^{2T-2t}}C_t^2.
\end{aligned} \tag{17}$$

We observe that both $e_j^{(t)}$ and $\beta_t = \sum_{j=2}^n e_j^{(t)}\lambda_j(\tilde{W})^{T-t}v_{j,i}$ can be written as linear combinations of $y_1^{(t)}, \ldots, y_n^{(t)}$. Let $\mathsf{b}_1^{(t)}, \ldots, \mathsf{b}_n^{(t)} \in \mathbb{R}$ be such that $\beta_t = \sum_{j=1}^n \mathsf{b}_j^{(t)} y_j^{(t)}$. By (17), $\mathrm{Var}\left[\beta_t\right] = 2\sum_{j=1}^n \mathsf{b}_j^2 C_t^2 \leq 2\frac{\lambda_1(\tilde{W})^{2T-2t}}{g^{2T-2t}}C_t^2$ and

$\sum_{j=1}^{n} \mathsf{b}_j^2 \leq \frac{\lambda_1(\tilde{W})^{2T-2t}}{g^{2T-2t}}$. Using the Chernoff bound and the moment generating function of the Laplacian distribution, we obtain that

$$\begin{aligned}
\Pr\left[\beta_t \geq \sqrt{2}\log n \cdot \frac{\lambda_1^{T-t}(\tilde{W})}{g^{T-t}} \cdot C_t\right] &\leq e^{-\log n} \cdot \mathbb{E}\left[\exp\left(\frac{\beta_t}{\sqrt{2} \cdot \frac{\lambda_1^{T-t}(\tilde{W})}{g^{T-t}} \cdot C_t}\right)\right] \\
&\leq \frac{1}{n}\mathbb{E}\left[\exp\left(\frac{\sum_j \mathsf{b}_j \mathrm{Lap}(0,1)}{\sqrt{2} \cdot \frac{\lambda_1^{T-t}(\tilde{W})}{g^{T-t}}}\right)\right] \\
&= \frac{1}{n}\prod_j \frac{1}{1 - \mathsf{b}_j^2 / \left(2 \cdot \frac{\lambda_1^{2T-2t}(\tilde{W})}{g^{2T-2t}}\right)} \\
&\leq \frac{1}{n}\exp\left(\frac{g^{2T-2t}}{\lambda_1^{2T-2t}(\tilde{W})}\sum_j \mathsf{b}_j^2\right) \leq \frac{e}{n}.
\end{aligned}$$

Let $\mathcal{E}$ be the event that $\max_t C_t \leq \frac{10}{9\epsilon} \cdot \frac{\lambda_1(\tilde{W})^{t-1}}{\sqrt{n}\log^2 n}$. For large n such that $\log n \geq \frac{650\sqrt{2}}{9\epsilon} \cdot \frac{g}{g-1} \cdot \frac{1}{\gamma}$,

$$\Pr\left[\beta_t \geq \frac{\gamma\lambda_1(\tilde{W})^T}{65 \cdot \frac{g}{g-1}\sqrt{n}g^{T-t}} \mid \mathcal{E}\right] \leq \Pr\left[\beta_t \geq \frac{10\sqrt{2}}{9\epsilon} \cdot \frac{\lambda_1^{T-1}(\tilde{W})}{\log n\sqrt{n}g^{T-t}} \mid \mathcal{E}\right] \leq \frac{e}{n}.$$

Recall that $\mathcal{E}_\delta$ is the event such that $\delta \geq \sqrt{n}\log^4 n$. By Lemma D.6, we know that $\Pr[\mathcal{E} \mid \mathcal{E}_\delta] = 1 - \frac{8eT^2}{n^2}$, and, by Proposition C.2, we know that $\Pr[\bar{\mathcal{E}}_\delta] \leq \frac{1}{2n}$. As a result, when n is large enough,

$$\begin{aligned}
\Pr\left[\beta_t \geq \frac{\gamma\lambda_1(\tilde{W})^T}{65 \cdot \frac{g}{g-1}\sqrt{n}g^{T-t}}\right] &\leq \Pr\left[\beta_t \geq \frac{\gamma\lambda_1(\tilde{W})^T}{65 \cdot \frac{g}{g-1}\sqrt{n}g^{T-t}} \text{ and } \mathcal{E} \text{ and } \mathcal{E}_\delta\right] + \Pr[\bar{\mathcal{E}} \text{ and } \mathcal{E}_\delta] + \Pr[\mathcal{E}_\delta] \\
&\leq \Pr\left[\beta_t \geq \frac{\gamma\lambda_1(\tilde{W})^T}{65 \cdot \frac{g}{g-1}\sqrt{n}g^{T-t}} \mid \mathcal{E}\right] + \Pr\left[\bar{\mathcal{E}} \mid \mathcal{E}_\delta\right] + \frac{1}{2n} = O\left(\frac{1}{n}\right).
\end{aligned}$$

By the union bound and by $\sum_{t=1}^{T} \frac{1}{g^{T-t}} = \frac{1}{g^T}\sum_{t=1}^{T} g^t = \frac{1}{g^T}\frac{g^{T+1}-1}{g-1} = \frac{gn^2-1}{gn^2-n^2} \leq \frac{g}{g-1}$, we obtain that

$$\Pr\left[\sum_{t=1}^{T}\beta_t \geq \frac{\gamma\lambda_1(\tilde{W})^T}{65\sqrt{n}}\right] \leq \Pr\left[\sum_{t=1}^{T}\beta_t \geq \sum_{t=1}^{T}\frac{\gamma\lambda_1(\tilde{W})^T}{65 \cdot \frac{g}{g-1} \cdot \sqrt{n}g^{T-t}}\right] = O\left(\frac{\log n}{n}\right).$$

As $\sum_{t=1}^{T}\beta_t$ is a linear combination of Laplacian random variable, we know that the distribution of the summation is symmetric around 0. Hence,

$$\Pr\left[\left|\sum_{j=2}^{n}\sum_{t=1}^{T} e_j^{(t)}\lambda_j(\tilde{W})^{T-t}v_{j,i}\right| \geq \frac{\gamma\lambda_1(\tilde{W})^T}{65\sqrt{n}}\right] = o(1). \tag{18}$$

We then obtain the lemma statement by combining (15) with inequalities (16) and (18). □

E. Further Experiments

In this appendix, we present additional experimental results, specifically demonstrating that each modification introduced in Algorithm 1 contributes to improved precision.

E.1. Results of using the lazy random walk matrix in the iterative spectral clustering (Difference 3)

In this subsection, we analyze the effect of performing power iteration with the lazy random walk matrix $W_\alpha = \alpha I + (1-\alpha)D^{-1}A$ instead of the usual random walk matrix $D^{-1}A$ used during PIC (Lin & Cohen, 2010; Boutsidis et al., 2015).

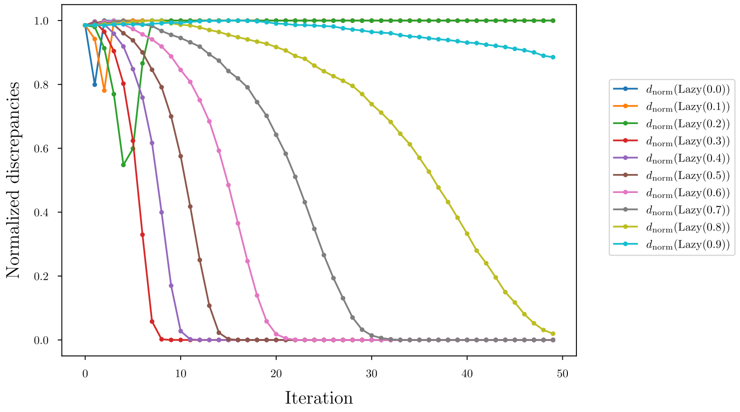


Figure 3. Power iteration on $\mathrm{BSBM}(1000, 1000, 1000, 1000, 0.5, 0.2)$ for lazy random walk matrices W_α with $\alpha \in \{0, 0.1, \ldots, 0.9\}$.

We start with an n-dimensional standard normal variable $\mathbf{x}^{(0)}$, and iteratively obtain $\mathbf{x}^{(t)} = W_\alpha \cdot \mathbf{x}^{(t-1)} - \frac{1}{n}\sum_i \mathbf{x}_i^{(t-1)}$. This is equivalent to the PIC algorithm with $k = 2$ initial vectors.

For bipartite graphs, the random walk matrix $W_0 = D^{-1}A$ has -1 as an eigenvalue. Thus, for bipartite 2-clustered graphs, the performance of PIC is not good unless more initial vectors are selected. We demonstrate this by introducing a *Bipartite Stochastic Block Model* graph with two clusters, defined as follows. Given integers a_i, b_i and probabilities p and q, a graph $G \sim \mathrm{BSBM}(a_1, a_2, b_1, b_2, p, q)$ has node set $A_1 \sqcup A_2 \sqcup B_1 \sqcup B_2$ with $|A_i| = a_i$ and $|B_i| = b_i$, such that every pair of nodes between A_i and B_i is added with probability p, and A_i and B_j are added with probability q. This graph is bipartite with independent sets $A_1 \cup A_2$ and $B_1 \cup B_2$, and when $p \gg q$, admits a clear cluster structure given by the node clusters $A_1 \cup B_1$ and $A_2 \cup B_2$.

Figure 3 shows that for certain BSBM's, the produced clusters by iteratively multiplying W_α always have discrepancy close to 1 when α is close to 0. On the other hand, when $\alpha \approx 1$, the procedure is too slow to converge since $W_\alpha \approx I$. Therefore, selecting a lazy factor of $\alpha = \frac{1}{2}$ seems to be a natural choice in general when no additional information about the input graph is available.

E.2. Result of leading eigenvector elimination (Difference 4)

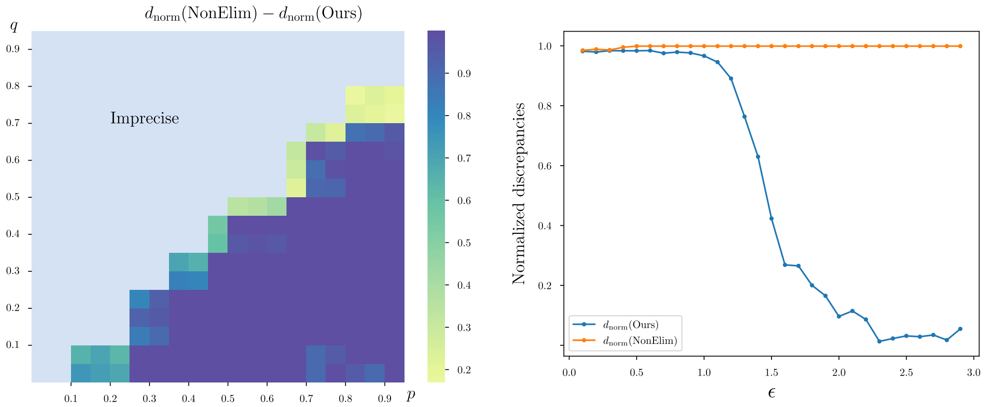


Figure 4. (Left): Heatmap of average $d_{\mathrm{norm}}(\mathrm{NonElim}) - d_{\mathrm{norm}}(\mathrm{Ours})$ over 20 SBMs with $n_1 = n_2 = 1000$, with varying probabilities $p, q \in \{0.05, 0.1, \ldots, 0.95\}$, and privacy budget $\epsilon = 2.0$. (Right): Discrepancy with increasing ϵ for 20 SBMs with $p = 0.3, q = 0.2$.

Now, we perform an experiment to investigate the effect of elimination of the leading eigenvalue of the lazy random walk matrix, which is a procedure changing the matrix W to the matrix $\widetilde{W}$ described in Difference 4 of Section 3. For this experiment, we select $\varepsilon = 2.0$ and generate 20 SBM's of cluster sizes 1000 for pairs of probabilities (p, q).

We present our results in Figure 4. The figure illustrates that our algorithm, without the leading vector elimination (referred to as NonElim), consistently fails to recover the original clusters. Although not depicted in the graph, we observed that the NonElim algorithm fails to successfully identify the clusters even when the privacy budget ϵ is set as high as 20. In contrast, our proposed method successfully identifies these clusters with minimal discrepancy.