Improved Large-Scale Graph Learning through Ridge Spectral Sparsification

Daniele Calandriello [1 2] **Alessandro Lazaric** [3] **Ioannis Koutis** [4] **Michal Valko** [1]

Abstract

The representation and learning benefits of methods based on graph Laplacians, such as *Laplacian smoothing* or *harmonic function solution for semi-supervised learning* (SSL), are empirically and theoretically well supported. Nonetheless, the exact versions of these methods scale poorly with the number of nodes n of the graph. In this paper, we combine a spectral sparsification routine with Laplacian learning. Given a graph $\mathcal{G}$ as input, our algorithm computes a sparsifier in a *distributed* way in $\mathcal{O}(n\log^3(n))$ time, $\mathcal{O}(m\log^3(n))$ work and $\mathcal{O}(m\log(n))$ memory, using only $\log(n)$ rounds of communication. Furthermore, motivated by the regularization often employed in learning algorithms, we show that constructing sparsifiers that preserve the spectrum of the Laplacian *only up to* the regularization level may drastically reduce the size of the final graph. By constructing a spectrally-similar graph, we are able to bound the error induced by the sparsification for a variety of downstream tasks (e.g., SSL). We empirically validate the theoretical guarantees on Amazon co-purchase graph and compare to the state-of-the-art heuristics.

1. Introduction

Graphs are a very effective data structure to represent relationships between entities (e.g., social and collaboration networks, influence graphs). Over the years, many machine learning problems have been defined and solved exploiting the graph representation, such as *graph-regularized least squares* (LapRLS, Belkin et al. 2005), *Laplacian smoothing* (LapSmo, Sadhanala et al. 2016) graph *semi-supervised learning* (SSL, Chapelle et al. 2010; Zhu et al. 2003), *laplacian embedding* (LE, Belkin & Niyogi 2001, and *spectral clustering* (SC, Von Luxburg 2007). The intuition behind graph-based learning is that the information expressed by the graph helps to capture the underlying structure of the problem (e.g., a manifold), thus improving the learning. For instance, LapSmo and SSL rely on the assumption that nodes that are *close* in the graph are more likely to have similar labels. Similarly, LE and SC try to find a low-dimensional representation of the nodes using the eigenvectors of the Laplacian of the graph. In general, given a graph $\mathcal{G}$ of n nodes and m edges, most of graph-based learning tasks require computing the minimum of a cost function based on the associated $n \times n$ Laplacian matrix $\mathbf{L}_\mathcal{G}$, which contains m non-zero entries. Solving *exactly* such optimization problems amounts to $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space complexity in the worst case and they become infeasible even for mildly large/dense graphs.

A complete review of the literature on large-scale graph learning is beyond the scope of this paper and we only consider methods that reduce learning space and time complexity starting from a given graph received as input.[1] We identify mainly three possible approaches. We can (1) reduce runtime replacing the pseudo-inverse operator $\mathbf{L}_\mathcal{G}^+$ with an *iterative solver*, (2) reduce time and space complexity replacing the large graph $\mathcal{G}$ with a sparser approximation $\mathcal{H}$, or (3) reduce runtime and increase memory capacity by *distributing* the computation across multiple machines.

Iterative solvers. Iterative methods can solve a number of learning problems without explicitly constructing $\mathbf{L}_\mathcal{G}^+$ (e.g., gradient descent, GD, for LapSmo, iterative averaging for SSL, and the power method for SC). In this case we only need $\mathcal{O}(m)$ time per iteration. Unfortunately, all simple iterative methods (e.g., GD) converge in a number of iterations proportional to the condition number of the Laplacian, $\kappa = \lambda_{\max}(\mathbf{L}_\mathcal{G})/\lambda_{\min}(\mathbf{L}_\mathcal{G})$, which may grow linearly with the number of nodes n, thus removing the advantage of the iterative method, whose complexity tends to $\mathcal{O}(n^3)$ in the worst case. Advanced iterative methods, such as the *preconditioned conjugate gradient,* use preconditioning to find an accurate solution in a number of iterations independent of κ. Koutis et al. (2011) gives a nearly-linear solver for Laplacians or *strongly diagonally dominant* (SDD) matri-

[1] SequeL team, INRIA Lille - Nord Europe, France [2] LCSL, IIT, Italy, and MIT, USA. [3] Facebook AI Research, Paris, France [4] New Jersey Institute of Technology, USA. Correspondence to: Daniele Calandriello <daniele.calandriello@iit.it>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018. Copyright 2018 by the author(s).

[1] Many algorithms reduce the complexity of graph learning *at construction* time but they cannot be applied to *natural* graphs (e.g., social graphs) and therefore we do not review them.

ces, that using a chain of preconditioners, converges in only $\mathcal{O}(m\log(n))$ time. As space and time costs scale with the number of edges, a natural desire is to reduce m by sparsifying and distributing the graph.

Graph sparsification. The objective of sparsification methods is to remove *redundant* edges, so that the resulting sparse sub-graph can be easily stored in memory and efficiently manipulated to compute final solutions. A simple *graph-sparsification* technique is to sample $n\overline{q}$ (with $\overline{q} > 1$) edges from $\mathcal{G}$ with probabilities proportional to the edge weights with replacement. While computationally very efficient, uniform sampling requires sampling a number of edges proportional to $\mathcal{O}(n\mu(\mathcal{G}))$ (i.e., $\overline{q} \propto \mu(\mathcal{G})$), where $\mu(\mathcal{G})$ is the *coherence* of the Laplacian matrix, and it can grow as large as n when the graph is highly structured (e.g., if there is a single edge e connecting two components of the graph we need to sample all of the edges of the graph—potentially $\mathcal{O}(n^2)$—to guarantee that we do not exclude e and generate an inappropriate $\mathcal{H}$). A more refined approach is the k-neighbors (KN) sparsifier (Sadhanala et al., 2016), which performs local sparsifications node-by-node by keeping all edges at nodes with degree smaller than $\overline{q}$, and samples them proportionally to their weights whenever the degree is bigger than $\overline{q}$. While in certain structured graphs, this method may perform much better than uniform (Von Luxburg et al., 2014), in the general case $\overline{q}$, still needs to scale with the coherence $\mu(\mathcal{G})$. A more effective method is to sample edges proportionally to their *effective resistance*, which intuitively measures the importance of an edge in preserving the minimum distance between two nodes. As a result, only relevant edges are kept and the sparsified graph could be reduced to $\mathcal{O}(n\,\mathrm{polylog}(n))$ edges. Nonetheless, computing effective resistances also requires the pseudo-inverse $\mathbf{L}_\mathcal{G}^+$, thus being as expensive as solving any graph-Laplacian learning problem.

Distributed computing. When the number of edges m is too large to fit the whole graph in a single machine, we are forced to distribute the edges across multiple machines. At the same time, if the sparsifier construction or the downstream inference can be parallelized, we can also reduce their runtime. Unfortunately, distributing data and computation across multiple machines can cause large communication costs. For example, simple GD or label propagation methods require $\mathcal{O}(\kappa)$ iterations (and communication rounds) to converge and access to non-local (e.g., neighbors in graph) data. While preconditioned solvers reduce the number of iterations, almost none of their memory access is local, thus making difficult to have efficient distributed implementations.

Contribution. In this paper, we propose a new approach that aims at integrating the benefits of the three different methods above. Using the large memory and computational capacity of distributed computing and leveraging the sequential sparsification methods of Kelner & Levin (2013) and Calandriello et al. (2017), we show how to compute an accurate sparsifier $\mathcal{H}$ of graph $\mathcal{G}$ in $\mathcal{O}(n\log^3(n))$ time, $\mathcal{O}(n\log^2(n))$ work and $\mathcal{O}(n\log(n))$ memory, using only $\log(n)$ rounds of communication. Afterwards, learning tasks can be solved directly on $\mathbf{L}_\mathcal{H}$ on a single machine using near-linear time solvers, resulting in an overall $\mathcal{O}(n\log^3(n))$ runtime. Moreover, we show that the regularization used in some graph-based learning algorithms allows using even sparser graphs. In particular, we introduce the notion of *ridge* effective resistance to obtain sparsifiers that are better adapted to solve Laplacian-regularized learning tasks (e.g., LAPSMO, SSL) and are smaller than standard spectral sparsifiers without compromising the performance of downstream tasks.

2. Background

We use lowercase letters a for scalars, bold lowercase letters $\mathbf{a}$ for vectors and uppercase bold letters $\mathbf{A}$ for matrices. We use $\mathbf{A} \preceq \mathbf{B}$ to denote that $\mathbf{B} - \mathbf{A}$ is positive semi-definite (PSD), $[\mathbf{A}]_{i,j}$ to indicate the (i,j)-th entry of $\mathbf{A}$, and ordered the eigenvalues as $\lambda_1(\mathbf{A}) \leq \ldots \leq \lambda_n(\mathbf{A})$.

2.1. Graphs and graph Laplacian

We denote with $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, an undirected weighted graph with n nodes $\mathcal{V}$ and m edges $\mathcal{E}$. Each edge $e_{i,j} \in \mathcal{E}$ has a weight $a_{e_{i,j}}$ measuring the "similarity" between nodes i and j. Given graphs $\mathcal{G}$ and $\mathcal{G}'$ over the same set of nodes $\mathcal{V}$, $\mathcal{G} + \mathcal{G}'$ denotes the graph obtained by summing the weights of their edges. For graph $\mathcal{G}$, we introduce the weighted adjacency matrix $\mathbf{A}_\mathcal{G}$ with entries $[\mathbf{A}_\mathcal{G}]_{i,j} = a_{e_{i,j}}$, the total weights $A = \sum_e a_e$, and the diagonal degree matrix $\mathbf{D}_\mathcal{G}$ with entries $[\mathbf{D}_\mathcal{G}]_{i,i} \triangleq \sum_j a_{e_{i,j}}$. The Laplacian of $\mathcal{G}$ is the PSD matrix $\mathbf{L}_\mathcal{G} = \mathbf{D}_\mathcal{G} - \mathbf{A}_\mathcal{G}$. Furthermore, we assume that $\mathcal{G}$ is connected and thus $\mathbf{L}_\mathcal{G}$ has only one eigenvalue equal to 0 and $\mathrm{Ker}(\mathbf{L}_\mathcal{G}) = \mathbf{1}$. Let $\mathbf{L}_\mathcal{G}^+$ be the pseudoinverse of $\mathbf{L}_\mathcal{G}$ and $\mathbf{L}_\mathcal{G}^{-1/2} = (\mathbf{L}_\mathcal{G}^+)^{1/2}$. For any node $i = 1, \ldots, n$, we denote with $\chi_i \in \mathbb{R}^n$, the indicator vector, so that $\mathbf{b}_e = \sqrt{a_e}(\chi_i - \chi_j)$ is the "edge" vector. If we denote with $\mathbf{B}_\mathcal{G}$ the $m \times n$ signed edge-vertex incidence matrix, then the Laplacian can be written as $\mathbf{L}_\mathcal{G} = \sum_e \mathbf{b}_e\mathbf{b}_e^\mathsf{T} = \mathbf{B}_\mathcal{G}^\mathsf{T}\mathbf{B}_\mathcal{G}$.

2.2. Learning on graphs

Given graph $\mathcal{G}$ and its Laplacian $\mathbf{L}_\mathcal{G}$, we denote with $\mathbf{f} \in \mathbb{R}^n$, a *labeling* of its nodes, where $[\mathbf{f}]_i$ is the value associated with the i-th node. Many graph learning algorithms assume that the optimal labeling $\mathbf{f}^\star$ is *smooth* w.r.t. the graph, i.e., the quantity $\sum_e a_e([\mathbf{f}^\star]_{e_i} - [\mathbf{f}^\star]_{e_j})^2 = \mathbf{f}^{\star\mathsf{T}}\mathbf{L}_\mathcal{G}\mathbf{f}^\star$ is small. In the following, we review examples from the supervised, semi-supervised and unsupervised learning with graphs.

Laplacian smoothing (LAPSMO) with Gaussian noise. Given a graph $\mathcal{G}$ on n nodes, let $\mathbf{y} \triangleq \mathbf{f}^\star + \xi$ be a noisy measurement of $\mathbf{f}^*$ with $[\xi]_i \sim \mathcal{N}(0, \sigma^2)$. The goal of LAPSMO is to find a vector $\widehat{\mathbf{f}}$ that accurately reconstructs $\mathbf{f}^*$ under the graph smoothness assumption by solving

$$\begin{aligned}\widehat{\mathbf{f}} &\triangleq \arg\min_{\mathbf{f}\in\mathbb{R}^n} (\mathbf{f}-\mathbf{y})^\mathsf{T}(\mathbf{f}-\mathbf{y}) + \lambda \mathbf{f}^\mathsf{T}\mathbf{L}_\mathcal{G}\mathbf{f}\\ &= (\lambda\mathbf{L}_\mathcal{G} + \mathbf{I})^{-1}\mathbf{y}, \end{aligned} \tag{1}$$

where λ is a regularization parameter.

Graph semi-supervised learning (SSL). In SSL, the input $\mathbf{f}_l$ is a partial observation of the labels $\mathbf{f}^\star$ for a subset $\mathcal{S} \subset [n]$ of nodes. The goal is to predict the labels $\mathbf{f}_u$ of the unrevealed nodes. The *harmonic function solution* (HFS) by Zhu et al. (2003) solves the optimization problem

$$\begin{aligned}\widehat{\mathbf{f}}_{\text{HFS}} &\triangleq \arg\min_{\mathbf{f}\in\mathbb{R}^n} \tfrac{1}{\ell}(\mathbf{f}-\mathbf{y})^\mathsf{T}\ell_\mathcal{S}(\mathbf{f}-\mathbf{y}) + \lambda \mathbf{f}^\mathsf{T}\mathbf{L}_\mathcal{G}\mathbf{f}\\ &= (\lambda\ell\mathbf{L}_\mathcal{G} + \mathbf{I}_\mathcal{S})^{+}\mathbf{y}_\mathcal{S}, \end{aligned} \tag{2}$$

where $\ell \triangleq |\mathcal{S}|$ is the number of labeled nodes received as input, $\mathbf{I}_\mathcal{S} \in \mathbb{R}^{n\times n}$ is the identity matrix with zeros at nodes not in $\mathcal{S}$, and $\mathbf{y}_\mathcal{S} \triangleq \mathbf{I}_\mathcal{S}\mathbf{y} \in \mathbb{R}^n$. Similarly, in *local transductive regression* (LTR) (Cortes et al., 2008), the optimization problem is

$$\begin{aligned}\widehat{\mathbf{f}}_{\text{LTR}} &\triangleq \arg\min_{\mathbf{f}\in\mathbb{R}^n} (\mathbf{f}-\mathbf{y})^\mathsf{T}\mathbf{C}(\mathbf{f}-\mathbf{y}) + \mathbf{f}^\mathsf{T}(\mathbf{L}_\mathcal{G} + \lambda\mathbf{I})\mathbf{f}\\ &= (\mathbf{C}^{-1}(\mathbf{L}_\mathcal{G} + \lambda\mathbf{I}) + \mathbf{I})^{-1}\mathbf{y}_\mathcal{S}, \end{aligned} \tag{3}$$

where $\mathbf{C}$ is a diagonal matrix with entries c_ℓ for nodes in $\mathcal{S}$, c_u for entries not in $\mathcal{S}$, and $c_\ell \geq c_u > 0$.

Spectral clustering (SC). Applying the Laplacian smoothness assumption, the goal of SC is to find k disjoint subset assignments such that the clusters are smooth w.r.t. the Laplacian. Let $\{\mathbf{f}_c\}_{c=1}^k$ be the cluster indicator vectors such that $[\mathbf{f}_c]_i \triangleq 1$ if node i is in the c-th cluster and $[\mathbf{f}_c]_i \triangleq 0$ otherwise. Denote with $\mathbf{F} \in \mathbb{R}^{n\times k}$, the matrix containing the assignments, and let $\mathcal{C}$ be the space of feasible clustering, such that all $\mathbf{f}_c$ are binary and each row of $\mathbf{F}$ contains only one non-zero entry. Since computing the minimum ratio-cut is NP-hard (Von Luxburg, 2007; Lee et al., 2014), even under constraints (Cucuringu et al., 2016), SC defines instead the relaxed problem

$$\widehat{\mathbf{F}} \triangleq \arg\min_{\mathbf{F}:\mathbf{F}^\mathsf{T}\mathbf{F}=\mathbf{I}_k, \mathbf{f}_c\perp\mathbf{1}} \operatorname{Tr}(\mathbf{F}^\mathsf{T}\mathbf{L}\mathbf{F}). \tag{4}$$

Once the relaxed solution is computed, we can use different heuristics to recover the clustering, such as thresholding or performing a k-means clustering on the $\widehat{\mathbf{F}}$ matrix.

Computational complexity. The problems above require either to compute an eigendecomposition of the Laplacian $\mathbf{L}_\mathcal{G}$ or to solve a linear system involving $\mathbf{L}_\mathcal{G}$. Computing these exactly is not feasible when the number of nodes n and edges m grows. In particular, (a) storing $\mathbf{L}_\mathcal{G}$ in memory requires $\mathcal{O}(m)$ space, and it is not feasible when m is large, (b) even if $\mathbf{L}_\mathcal{G}$ is sparse and m is small, the pseudo-inverse $\mathbf{L}_\mathcal{G}^+$ might be dense, and thus computing and storing $\mathbf{L}_\mathcal{G}^+$ exactly requires up to $\mathcal{O}(n^3)$ time and $\mathcal{O}(n^2)$ space.

3. Distributed Spectral Sparsification

In this section, we describe a new, sequential, distributed, and efficient algorithm for graph sparsification that can be used as a preprocessing step to solve a large variety of downstream learning tasks, without significantly affecting their performance. We point out that while distributing data-agnostic sparsifiers (e.g. uniform sampling) is straightforward, distributing the computation of sparsifiers based on effective resistances requires a careful merging procedure to guarantee satisfactory memory vs. accuracy tradeoff, which is what we provide in this section.

3.1. (ε, γ)-Spectral Sparsifiers

We start with the introduction of the notion of (ε, γ)-sparsifier that is adapted for the learning tasks that use sparsified graph Laplacian.

Definition 1. *A (ε, γ)-spectral sparsifier of $\mathcal{G}$ is a reweighted sub-graph $\mathcal{H} \subseteq \mathcal{G}$ whose Laplacian $\mathbf{L}_\mathcal{H}$ satisfies*

$$(1-\varepsilon)\mathbf{L}_\mathcal{G} - \varepsilon\gamma\mathbf{I} \preceq \mathbf{L}_\mathcal{H} \preceq (1+\varepsilon)\mathbf{L}_\mathcal{G} + \varepsilon\gamma\mathbf{I}. \tag{5}$$

For $\gamma = 0$, this definition reduces to the standard notion of ε-spectral sparsifier (Spielman & Teng, 2011). The main difference is that an (ε, γ)-spectral sparsifier allows for an extra *additive* error of order $\varepsilon\gamma$. This change is directly motivated by the fact that the sparsifier $\mathcal{H}$ may be used in learning tasks whose solution may not be sensitive to small (additive) errors. As a result, (ε, γ)-spectral sparsifiers are able to further reduce the size of $\mathcal{H}$ w.r.t. $(\varepsilon, 0)$-sparsifiers, without significantly affecting the final learning performance. Formally, an ε-sparsifier preserves all the quadratic forms up to a small multiplicative (constant) error, and thus can be used to provide an accurate approximation to many important quantities such as graph cuts or eigenvalues. In fact, for all $i \in [n]$, an ε-sparsifier guarantees that $(1-\varepsilon)\lambda_i(\mathbf{L}_\mathcal{G}) \leq \lambda_i(\mathbf{L}_\mathcal{H}) \leq (1+\varepsilon)\lambda_i(\mathbf{L}_\mathcal{G})$. Nonetheless, in many learning tasks (e.g., LTR) the noise level in the signal $\mathbf{f}$ requires regularizing the solution so that the Laplacian $\mathbf{L}_\mathcal{G}$ itself is eventually replaced by $\mathbf{L}_\mathcal{G} + \lambda\mathbf{I}$ (e.g., Eq. 1). This corresponds to *soft-thresholding* the eigenvalues of the Laplacian, so that eigenvalues below λ are partially ignored. If λ is properly tuned w.r.t. the noise, the regularization increases stability and improves the learning performance. Therefore, constructing a sparsifier that accurately reconstructs *all* eigenvalues of $\mathbf{L}_\mathcal{G}$ may be wasteful, as it may require keeping most of the edges. As a result, in tasks where $\mathbf{L}_\mathcal{G}$ is regularized, it is better to use (ε, γ)-sparsifiers, as their additive error $\gamma\mathbf{I}$ is homogeneous with the regularization $\lambda\mathbf{I}$ and their smaller size allows scaling to larger problems.[2]

We now extend the results of Spielman & Srivastava (2011) for the construction of ε-spectral sparsifiers to the general case of (ε,γ)-sparsifiers. We redefine the edge effective resistance to account for the regularization.

Definition 2. *The γ-effective resistance of an edge e in graph $\mathcal{G}$ is defined as*

$$r_e(\gamma) \triangleq \mathbf{b}_e^\mathsf{T}\left(\mathbf{L}_\mathcal{G} + \gamma\mathbf{I}\right)^{-1}\mathbf{b}_e. \tag{6}$$

The "effective dimension" of the graph is the total sum of the γ-effective resistances, $d_{\mathit{eff}}(\gamma) = \sum_e r_e(\gamma)$.

We can now construct a sparsifier $\mathcal{H}$ by sampling $\overline{q}$ times each edge with a probability proportional to its γ-effective resistance. More formally, the resulting (random) graph contains $q_e \sim \mathcal{B}(r_e(\lambda);\overline{q})$ copies of each edge, where $\mathcal{B}$ is the Binomial distribution, and its associated Laplacian is $\mathbf{L}_\mathcal{H} = \sum_{e\in\mathcal{H}} q_e/(\overline{q}r_e(\gamma))\mathbf{b}_e\mathbf{b}_e^\mathsf{T}$, which is an unbiased estimator of $\mathbf{L}_\mathcal{G}$. We can then apply existing results from sketching of PSD matrices (Alaoui & Mahoney, 2015) to prove that $\mathcal{H}$ is a valid (ε,γ)-sparsifier.

Proposition 1 (Cohen et al. 2017)**.** *Let $\varepsilon > 0$ and $\gamma \ge 0$ be the accuracy parameters and $0 \le \delta \le 1$ the probability of error. Let $\mathcal{H}$ be the graph obtained by sampling edges in $\mathcal{G}$ with a probability proportional to their γ-effective resistances. If $\overline{q} \ge 4\log(4n/\delta)/\varepsilon^2$, then w.p. $1-\delta$, $\mathcal{H}$ is an (ε,γ)-sparsifier with $\mathcal{O}(d_{\mathit{eff}}(\gamma)\overline{q})$ edges.*

We first notice that this result reduces to the one of Spielman & Srivastava (2011) for $\gamma = 0$. In fact, $d_{\text{eff}}(0) = n-1$ for all graphs, thus matching the space requirement $\overline{q}$ for ε-sparsifiers. Nonetheless, as γ increases, the size of $\mathcal{H}$ reduces significantly. Using $\mathbf{L}_\mathcal{G} = \mathbf{B}_\mathcal{G}^\mathsf{T}\mathbf{B}_\mathcal{G}$, the effective dimension $d_{\text{eff}}(\gamma)$ can be conveniently rewritten as

$$d_{\text{eff}}(\gamma) \triangleq \operatorname{Tr}\left(\mathbf{B}_\mathcal{G}^\mathsf{T}\mathbf{B}_\mathcal{G}(\mathbf{B}_\mathcal{G}^\mathsf{T}\mathbf{B}_\mathcal{G}+\gamma\mathbf{I})^{-1}\right) = \sum_{i=2}^{n} \frac{\lambda_i(\mathbf{L}_\mathcal{G})}{\lambda_i(\mathbf{L}_\mathcal{G})+\gamma},$$

thus showing that $d_{\text{eff}}(\gamma)$ is the "soft" rank of the Laplacian, where γ significantly reduces the contribution of small eigenvalues to the total sum. While in the worst case $d_{\text{eff}}(\gamma)$ can be as large as $n-1$, for a variety of graphs with rapidly decaying spectrum (Jamakovic & Mieghem, 2006; Samukhin et al., 2008; Zhan et al., 2010; Akoglu et al., 2015), $d_{\text{eff}}(\gamma)$ may be significantly smaller than $n-1$, thus reducing the number of edges $\overline{q}$ required to obtain an (ε,γ)-sparsifier.

3.2. The algorithm

As pointed out in the introduction, the main limitation of effective-resistance-based sparsification is that the computation of r_e requires inverting the Laplacian matrix, thus resulting in a computational cost that already matches the cost of the learning tasks themselves. Moreover large graphs cannot be stored in memory, and multiple passes over the graph would result in a disk access overhead larger than the computational cost. In order to avoid these problem, we adapt previous work (Calandriello et al., 2017) in online graph sparsification and randomized linear algebra (see a thorough discussion and comparison at the end of the section) to obtain the distributed sequential resparsification (DiSRe) algorithm in Alg. 1.[3]

Algorithm 1 The DiSRe algorithm.

```
Input: $\mathcal{G}$
Output: $\mathcal{H}_\mathcal{G}$
Partition $\mathcal{G}$ into $k$ sub-graphs $\mathcal{H}_{1,l} = \mathcal{G}_l = \{(e_{i,j}, q_e = 1, \widetilde{p}_{1,e} = 1)\}$
Initialize set $\mathcal{S}_1 = \{\mathcal{H}_{1,l}\}_{l=1}^k$
for $h = 1, \dots, k-1$ do
    Pick two sparsifiers $\mathcal{H}_{h,i'}, \mathcal{H}_{h,i'}$ from $\mathcal{S}_h$
    $\overline{\mathcal{H}}$ = Merge-Resparsify$(\mathcal{H}_{h,i}, \mathcal{H}_{h,i'})$
    Place $\overline{\mathcal{H}}$ back into $\mathcal{S}_{h+1}$
end for
Return $\mathcal{H}_\mathcal{G}$, the last sparsifier in $\mathcal{S}_k$
```

Algorithm 2 Merge-Resparsify

```
Require: $(\varepsilon,\gamma)$-sparsifiers $\mathcal{H}_{h,i}, \mathcal{H}_{h,i'}$ of graphs $\mathcal{G}_{h,i}, \mathcal{G}_{h,i'}$
Ensure: $\overline{\mathcal{H}}$, a $(\varepsilon,\gamma)$ sparsifier of $\mathcal{G}_{h,i} + \mathcal{G}_{h,i'}$
Initialize $\overline{\mathcal{H}} = \mathcal{H}_{h,i} + \mathcal{H}_{h,i'}$
For all $e \in \overline{\mathcal{H}}$ use a fast SDD solver to compute
    $\widetilde{r}_{h+1,e}(\gamma) = (1-\varepsilon)\mathbf{b}_e^\mathsf{T}(\mathbf{L}_{\overline{\mathcal{H}}} + (1+\varepsilon)\gamma\mathbf{I})^{-1}\mathbf{b}_e$
Set probabilities $\widetilde{p}_{h+1,e} = \min\{\widetilde{r}_{h+1,e}(\gamma), \widetilde{p}_{h,e}\}$
Sample $q_{h+1,e}$ from $\mathcal{B}(\widetilde{p}_{h+1,e}/\widetilde{p}_{h,e}, q_{h,e})$
Return $\overline{\mathcal{H}} = \{(e_{i,j}, q_{h+1,e}, \widetilde{p}_{h+1,e})\}$ for all $q_{h+1,e} > 0$
```

The structure. We represent a sparsifier $\mathcal{H}$ as a collection of weighted edges $\mathcal{H} = \{(e_{i,j}, q_e, \widetilde{p}_e)\}$, and the Laplacian can be reconstructed as $\mathbf{L}_\mathcal{H} = \sum_{e\in\mathcal{H}} 1/\widetilde{p}_e \frac{q_e}{\overline{q}}\mathbf{b}_e\mathbf{b}_e^\mathsf{T}$. Intuitively, each edge e has an associated weight based on its probability $\widetilde{p}_e$, and a number of included copies q_e. Keeping multiple copies of each edge helps the random $\mathbf{L}_\mathcal{H}$ to concentrate towards $\mathbf{L}_\mathcal{G}$, where the maximum number of copies $\overline{q}$ for an edge trades-off success probability and the size of $\mathcal{H}$. We assume we have k machines are available. DiSRe begins by partitioning the graph $\mathcal{G}$ into k sub-graphs $\mathcal{G}_l$ on n vertices and $m_l \ge n$ edges, such that $\mathcal{G} = \{\mathcal{G}_l\}_{i=l}^k$ In other words, it splits the matrix $\mathbf{B}_\mathcal{G}$ into submatrices $\mathbf{B}_{\mathcal{G}_i}$ by arbitrarily selecting a subset of rows. The sub-graphs are

[2] Whenever no regularization is required in the learning task (i.e., HFS,SC), we set $\gamma = 0$ and consider "standard" ε-sparsifiers.

[3] Whenever the original graph contains $m \le d_{\text{eff}}(\gamma)$.... edges, there is no need to run DiSRe as the (ε,γ)-sparsifiers would not reduce the size of the graph.

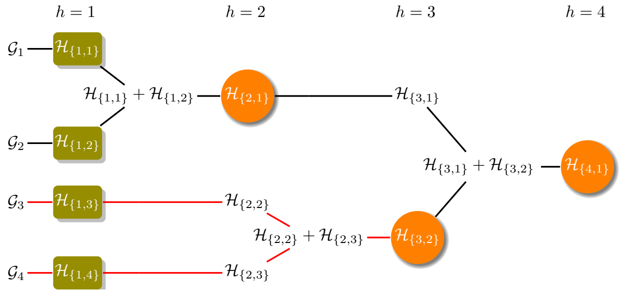


Figure 1. Merge tree for Algorithm 1.

small enough that they can be stored in memory,[4] and they are also obviously sparsifiers of themselves, therefore we can define an initial set of sparsifiers $\mathcal{S}_1 = \{\mathcal{H}_{1,l}\}_{l=1}^{k}$, with $\mathcal{H}_{1,l} = \{(e_{i,j}, q_{1,e} = \overline{q}, \widetilde{p}_{1,e} = 1)\}_{e \in \mathcal{G}_l}$. With this definition, $\mathcal{H}_{1,l}$ contains edges $e_{i,j}$ with unit weight $\widetilde{p}_{1,e} = 1$ and $\mathcal{H}_{1,l} = \mathcal{G}_l$. Starting from these initial sparsifiers, DISRE proceeds through a sequence of *merge* and *sparsify* operations where two sparsifiers are first combined and then sparsified again to keep having manageable-size graphs at each step. While DISRE can run on any arbitrary sequence of merges, we consider the most (computationally) effective scheme, where sparsifiers are merged two-by-two in parallel, thus inducing a *balanced* full binary merge tree (see Fig. 2). For notational convenience, we consider that at each iteration h, the inner loop of Alg. 1 only merges two arbitrary sparsifiers from the pool of available sub-graphs $\mathcal{S}_h$ and merges them into a new sparsifier. In practice, multiple merge-and-sparsify operations can be executed in a parallel and asynchronous way. The size of $\mathcal{S}_h$ (number of sparsifiers present at layer h) is $|\mathcal{S}_h| = k-h+1$. Therefore, a node in the tree corresponding to a sparsifier is uniquely identified by two indices $\{h, l\}$ where h is the height of the layer and $l \le |\mathcal{S}_h|$ is the index of the node in the layer. We also define the graph $\mathcal{G}_{\{h,l\}}$ as the union of all sub-graphs $\mathcal{G}_{l'}$ that are reachable from node $\{h, l\}$ as leaves (descendent of $\{h, l\}$). For example, in Fig. 2, sparsifier $\mathcal{H}_{3,1}$ in node $\{3, 1\}$ approximates the graph $\mathcal{G}_{\{3,2\}} = \mathcal{G}_3 + \mathcal{G}_4$, where we highlight in red the descendent tree.

The re-sparsification. In Alg. 2 we detail how two arbitrary sparsifiers are combined to obtain a temporary graph $\overline{\mathcal{H}}$. While the merge operation simply combines $\mathcal{H}_{h,i}$ and $\mathcal{H}_{h,i'}$ by summing their weights, the resparsification aims at generating a valid sparsifier from the "original" subgraph $(\mathcal{G}_{h,i} + \mathcal{G}_{h,i'})$, as if it was directly sparsified at the beginning. We first compute estimates $\widetilde{r}(\gamma)$ of the γ-effective resistance by using fast solvers to invert the strongly diagonal dominant $L_{\overline{\mathcal{H}}} + \gamma\mathbf{I}$ matrix. Instead of sampling edges in $\overline{\mathcal{H}}$ directly proportionally to $\widetilde{r}(\gamma)$ (more precisely $\widetilde{p}_{h+1,e}$), we perform a "resampling" scheme where an edge e is preserved with a "reweighted" probability $\widetilde{p}_{h+1,e}/\widetilde{p}_{h,e}$. Intuitively, the overall sequence of resampling guarantees that at each step $h+1$, an edge $e \in (\mathcal{G}_{h,i} + \mathcal{G}_{h,i'})$ has the "correct" probability $\widetilde{p}_{h+1,e}$ of being included in the sparsifier.

Performance. We now study the performance of DISRE and its complexity. *Time* complexity refers to the amount of time necessary to compute the final solution and *work* complexity refers to the total amount of operations carried out by all machines to compute the final solution.

Theorem 1. *Let $\varepsilon > 0$ be the accuracy parameter, $0 \le \delta \le 1$ the probability of error, and $\rho = (1+3\varepsilon)/(1-\varepsilon)$. Given an arbitrary graph $\mathcal{G}$ and an arbitrary merge tree structure, if* DISRE *is run with parameter $\overline{q} = 26\rho\log(3n/\delta)/\varepsilon^2$, then each sub-graphs $\mathcal{H}_{\{h,l\}}$ is a (ε, γ)-sparsifier of $\mathcal{G}_{\{h,l\}}$ with at most $3\overline{q}d_{\mathit{eff}}(\gamma)$ edges with probability $1-\delta$. Whenever the merge tree is balanced and k is big enough such that $m/k \le 3\overline{q}d_{\mathit{eff}}(\gamma)$,[5] then merge operations can be run in parallel across the machines with an overall time complexity of $\mathcal{O}(d_{\mathit{eff}}(\gamma)\log^3(n))$, a total work $\mathcal{O}(m\log^3(n))$, and $\mathcal{O}(\log(n))$ rounds of communication.*

Discussion. Kelner & Levin (2013) proposed a sequential algorithm for graph sparsification that closely emulates the batch sampling of Spielman & Srivastava (2011) in a semi-streaming setting and incrementally constructs an ε-sparsifier. Kyng et al. (2016) resolved some issues in the original proof of Kelner & Levin (2013), and showed that a slightly modified algorithm can construct a sparsifier with $\mathcal{O}(n\log(n)/\varepsilon^2)$ edges in $\mathcal{O}(m\log^2(n)/\varepsilon^2)$ time, matching the space complexity of batch sampling. The method proposed by Kyng et al. (2016) can be further improved by parallelizing its computation over multiple machines. Using the parallel sparsification algorithm of Koutis & Xu (2016), the time complexity can be reduced up to $\widetilde{\mathcal{O}}(\log^6(n))$. Nonetheless, since these methods requires *random access to the edges*, they cannot be easily distributed (it would have $\mathcal{O}(m\,\mathrm{polylog}(n))$ communication cost) and scaled to graphs that cannot be stored on a single machine. Furthermore, the algorithm of Kyng et al. (2016) focused on accurately reconstructing the whole spectrum of the Laplacian, which leads to sparsifiers whose number of edges scales linearly with n. On the other hand, in regularized learning tasks, the presence of multiplicative and additive spectral error allows creating smaller sparsifiers whose size scales with $d_{\mathrm{eff}}(\gamma)$. Notice that, for γ large enough, this possibly means sparsifiers with less than $n-1$ edges, necessarily leading to disconnected graphs. Finally note that, merging two traditional ε-sparsifiers gives an ε-sparsifier, merging two (γ, ε)-sparsifiers produces a less accurate $(2\gamma, \varepsilon)$-sparsifier. Therefore simple merge-and-reduce strategies

[4] Whenever this is not possible (i.e., m/k is too large to be stored on a single machine), we can simply apply the same merging scheme of DISRE by loading *small enough* chunks of the graph and sparsifying them sequentially.

[5] This implies that there are enough machines so that the leaves in the merge tree already have relatively sparse sub-graphs.

(Feldman et al., 2013), which address every re-sparsification as independent, would either cumulate errors or require multiple passes over the data. Similarly to Kyng et al. (2016), DISRE's sequential MERGE-RESPARSIFY solves this problem.

Mixed additive-multiplicative reconstruction is studied more extensively in randomized matrix algebra (Drineas & Mahoney, 2017). Cohen et al. (2016) developed an efficient method to spectrally sparsify generic matrices up to $(1 \pm \varepsilon)$ multiplicative and γ additive errors using an incremental sampling method based on *ridge leverage scores* (i.e., the analog of γ-effective resistances for matrices). If applied to graph Laplacians, their method adds edges incrementally and returns a (ε, γ)-sparsified graph with $\mathcal{O}(d_{\text{eff}}(\gamma) \log^2(n))$ edges in $\mathcal{O}(m \log(n))$ time. Nonetheless, Cohen et al. (2016) focused only on ε-sparsifiers,suggesting to set γ as small as possible, and did not explore the advantages possible in the ML setting. Moreover, no existing (ε, γ)-sparsifier construction method can leverage both distribution and fast solvers. In (Cohen et al., 2016) edges can only be added (and not removed as in DISRE), preventing repeated merge-and-resparsify. Other streaming RLS sampling methods, such as Cohen et al. (2017), use dense intermediate sketches, such as Frequent Directions (Ghashami et al., 2016), that are not Laplacians of a sub-graph and cannot be easily paired with near-linear solvers for Laplacians. To solve both these problems DISRE modifies SQUEAK (Calandriello et al., 2017), a parallel dictionary learning algorithm, to operate on graphs. Therefore, it can be seen as a generalization of Cohen et al. (2016) that allows edge removal.

4. Downstream Guarantees

We now show how the spectral reconstruction guarantees provided by (ε, γ)-sparsifiers translate into guarantees on the quality of the approximate solutions computed using $\mathcal{H}$ instead of $\mathcal{G}$. We first introduce a result for ε-sparsifiers in SSL, and then show how for regularized problems (ε, γ)-sparsification can further improve computational performance without loss in accuracy in LAPSMO.

4.1. Generalization Bounds for SSL

Given the closed form solutions of HFS (Eq. 2) and LTR (Eq. 3), we simply replace $\mathbf{L}_{\mathcal{G}}$ with $\mathbf{L}_{\mathcal{H}}$, and then run a nearly-linear time solver to obtain approximate solutions $\widetilde{\mathbf{f}}_{\text{HFS}}$ and $\widetilde{\mathbf{f}}_{\text{LTR}}$. We compare approximate solutions to their exact counterparts in the context of algorithmic stability.

Definition 3. *Let $\mathcal{L}$ be a transductive learning algorithm. We denote by $\mathbf{f}$ and $\mathbf{f}'$ the solutions obtained by running $\mathcal{L}$ on datasets $\mathcal{V} = (\mathcal{S}, \mathcal{T})$ and $\mathcal{V} = (\mathcal{S}', \mathcal{T}')$ respectively. $\mathcal{L}$ is uniformly β-stable w.r.t. the squared loss if there exists $\beta \geq 0$ such that for any two partitions $(\mathcal{S}, \mathcal{T})$ and $(\mathcal{S}', \mathcal{T}')$ that differ by exactly one training (and test) point and for all $i \in [n]$,$|([\mathbf{f}]_i - [\mathbf{y}]_i)^2 - ([\mathbf{f}']_i - [\mathbf{y}]_i)^2| \leq \beta$.*

The stability of LTR was proven by Cortes et al. (2008). On the other hand, the singularity of the Laplacian may lead to unstable behavior in HFS due to the $(\gamma l \mathbf{L}_{\mathcal{G}} + \mathbf{I}_{\mathcal{S}})^{+}$ pseudo-inverse, with drastically different results for small perturbations to the dataset. For this reason, we focus on the STABLE-HFS algorithm proposed in (Belkin et al., 2004) where an additional regularization term is introduced to restrict the space of admissible solutions to the space $\mathcal{F} = \{\mathbf{f} : \langle \mathbf{f}, \mathbf{1} \rangle = 0\}$ of solutions orthogonal to the null space of $\mathbf{L}_{\mathcal{G}}$ (i.e., centered functions). As shown in (Belkin et al., 2004), to satisfy the constraint it is sufficient to set an additional regularization parameter μ to $\mu = ((\gamma l \mathbf{L}_{\mathcal{G}} + \mathbf{I}_{\mathcal{S}})^{+} \mathbf{y}_{\mathcal{S}})^{\mathsf{T}} \mathbf{1} / ((\gamma l \mathbf{L}_{\mathcal{G}} + \mathbf{I}_{\mathcal{S}})^{+} \mathbf{1})^{\mathsf{T}} \mathbf{1}$, and compute the solution $\widehat{\mathbf{f}}_{\text{STA}}$ as $\widehat{\mathbf{f}}_{\text{STA}} = (\gamma l \mathbf{L}_{\mathcal{G}} + \mathbf{I}_{\mathcal{S}})^{+} (\mathbf{y}_{\mathcal{S}} - \mu \mathbf{1})$. While STABLE-HFS is more stable and thus more suited for theoretical analysis, its space and time requirement remains $\mathcal{O}(m)$, and cannot be applied to graphs with a large number of edges. Therefore, we again replace $\widehat{\mathbf{f}}_{\text{STA}}$ with an approximate solution $\widetilde{\mathbf{f}}_{\text{STA}}$ computed using $\mathbf{L}_{\mathcal{H}}$. Define $\widehat{R}(\mathbf{f}) = \frac{1}{l} \sum_{i=1}^{l} (\mathbf{f}(x_i) - \mathbf{y}(x_i))^2$ as the empirical error and $R(\mathbf{f}) = \frac{1}{u} \sum_{i=1}^{u} (\mathbf{f}(x_i) - \mathbf{y}(x_i))^2$ as the generalization.

Theorem 2. *Let $\mathcal{G}$ be a fixed (connected) graph with eigenvalues $0 = \lambda_1(\mathcal{G}) < \lambda_2(\mathcal{G}) \leq \ldots \leq \lambda_n(\mathcal{G})$, and $\mathcal{H}$ an ε-sparsifier of $\mathcal{G}$. Let $\mathbf{y} \in \mathbb{R}^n$ be the labels of the nodes in $\mathcal{G}$ with $|\mathbf{y}(x)| \leq c$ and $\mathcal{F}$ be the set of centered functions such that $|\mathbf{f}(x) - \mathbf{y}(x)| \leq 2c$. Let $\mathcal{S} \subset \mathcal{V}$ be a random subset of labeled nodes, if the labels $\mathbf{y}_{\mathcal{S}}$ are centered, then w.p. at least $1 - \delta$ (w.r.t. the random generation of the sparsifier $\mathcal{H}$ and the random subset of labeled points $\mathcal{S}$) the resulting* STABLE-HFS *solution satisfies,*

$$R(\widetilde{\mathbf{f}}) \leq \widehat{R}(\widehat{\mathbf{f}}) + \beta + \left(2\beta + \frac{4c^2(l+u)}{lu}\right) \sqrt{\frac{\pi(l,u) \ln \frac{1}{\delta}}{2}} + \frac{1}{1-\varepsilon} \left(\frac{2(1+\varepsilon)\varepsilon l \gamma \lambda_2(\mathcal{G}) c}{((1-\varepsilon) l \gamma \lambda_2(\mathcal{G}) - 1)^2} \right)^2, \qquad (7)$$

where $\widetilde{\mathbf{f}}$ and $\widehat{\mathbf{f}}$ are computed on $\mathcal{H}$ and $\mathcal{G}$ and,

$$\pi(l,u) = \frac{lu}{l+u-0.5} \frac{2\max\{l,u\}}{2\max\{l,u\}-1}, \quad \textit{and}$$

$$\beta \leq \frac{3c\sqrt{l}}{((1-\varepsilon) l \gamma \lambda_2(\mathcal{G}) - 1)^2} + \frac{4c}{(1-\varepsilon) l \gamma \lambda_2(\mathcal{G}) - 1}.$$

Thm. 2 shows how approximating $\mathcal{G}$ with $\mathcal{H}$ impacts the generalization error as the number of labeled samples l increases. If we set $\varepsilon = 0$, we recover the exact case bound by Cortes et al. (2008), which depends only on $\widehat{R}(\widehat{\mathbf{f}})$ and β. When $\varepsilon > 0$, we see from Eq. (7) that the two terms already present in the exact case are either unchanged ($\widehat{R}(\widehat{\mathbf{f}})$) or increase only by a constant factor (β). Because of the approximation, a new error term (the last one in Eq. 7) is added to the bound, but we can see that it is negligible compared to β. In fact, it converges to zero as $O(\varepsilon^2/l^2(1-\varepsilon)^4)$ as l grows and it is dominated by β for any constant value of ε. This means that increasing ε corresponds to a constant increase in the bound, regardless of the size of the problem. Consequently, ε can be freely chosen to trade off accuracy and space complexity (Thm. 1) depending on the problem constraints. Finally, because the eigenvalues present in the bound are the ones of the original graph, any additional knowledge on the spectral properties of the input graph can be easily included in the analysis. Therefore it is straightforward to provide stronger guarantees for Sparse-HFS when combined with assumptions on the graph generating model. Finally, we remark the level of generality of this result that holds for the integration between HFS and any ε-accurate spectral sparsification method. We will postpone computational considerations to the following subsection.

4.2. Generalization Bounds for LapSmo

Starting from the closed form solution of LapSmo (Eq. 1) we can replace the $\mathbf{L}_\mathcal{G}$ matrix with a sparsified Laplacian $\mathbf{L}_\mathcal{H}$ and compute an approximate solution $\widetilde{\mathbf{f}} = (\lambda\mathbf{L}_\mathcal{H} + \mathbf{I})^{-1}\mathbf{y}$ in $\mathcal{O}(n\log^2(n))$ time and $\mathcal{O}(n\log(n))$ space using a fast linear solver. Finally, we can decompose the error as $\|\mathbf{f}^* - \widetilde{\mathbf{f}}\|_2^2 \leq \|\mathbf{f}^* - \widehat{\mathbf{f}}\|_2^2 + \|\widehat{\mathbf{f}} - \widetilde{\mathbf{f}}\|_2^2$. The first term can be bounded using classical results from empirical process theory (Bühlmann & Van De Geer, 2011). We can bound the second term as follows

Theorem 3. *For an arbitrary graph $\mathcal{G}$ and its (ε,γ)-sparsifier, let $\widehat{\mathbf{f}}$ be the* LapSmo *solution computed using $\mathbf{L}_\mathcal{G}$ and $\widetilde{\mathbf{f}}$ the solution computed using $\mathbf{L}_\mathcal{H}$. Then*

$$\|\widetilde{\mathbf{f}} - \widehat{\mathbf{f}}\|_2^2 \leq \frac{\varepsilon^2}{1-\varepsilon}\,(1/4+\lambda\gamma)\left(\lambda\widehat{\mathbf{f}}^\mathsf{T}\mathbf{L}_\mathcal{G}\widehat{\mathbf{f}} + \lambda\gamma\|\widehat{\mathbf{f}}\|_2^2\right)$$

where λ is the regularization of LapSmo.

For ε-sparsifiers Sadhanala et al. (2016) derive a similar bound $\|\widetilde{\mathbf{f}} - \widehat{\mathbf{f}}\|_2^2 \leq \mathcal{O}(\lambda\widehat{\mathbf{f}}^\mathsf{T}\mathbf{L}_\mathcal{G}\widehat{\mathbf{f}})$. Setting $\gamma = 0$, we recover their bound up to constants. When $\gamma > 0$ instead, additional error terms emerge due to the introduced bias. In particular, the term $\lambda\gamma\|\widehat{\mathbf{f}}\|_2^2$ depends on the norm of the exact solution $\widehat{\mathbf{f}}$, which in turn depends on the value of λ. Nonetheless, when $\|\mathbf{f}^*\|_2^2$ is small, as is the case in our experiments, setting $\gamma = 1/\lambda$ makes this term a constant, which is reflected by the good empirical performance. Computationally, for both Stable-HFS and LapSmo, passing from computing a solution on the full graph to computing a solution on the sparsifier can reduce the number of edges, which drives memory and runtime, significantly. Moreover, carefully distributing the sparsification process across multiple machines allows us to compute a final solution in a time *independent* from the number of edges, since the preprocessing sparsification step takes only $\mathcal{O}(n\log^3(n))$ time, and the solution step only $\mathcal{O}(n\log^2(n))$. Up to logarithmic terms, this results in an overall $\widetilde{\mathcal{O}}(n)$ near-linear runtime, without any assumptions on the input graph. For graphs with a particularly favorable spectrum, and problems with enough regularization, this can be reduced to $\widetilde{\mathcal{O}}(d_{\text{eff}}(\gamma))$, resulting in a potentially sub-linear runtime. This result, only possible due to particular structure of learning problems, opens up unexplored possibilities that would not be possible for general graph problems.

4.3. Bounds for other problems

Many other problems can be well approximated using (ε,γ)-sparsifiers. For example, the cost of a SC solution evaluated on $\mathbf{L}_\mathcal{H}$ is very close to the cost evaluated on $\mathbf{L}_\mathcal{G}$.

Proposition 2. *For any rank k orthogonal projection $\mathbf{F}^\mathsf{T}\mathbf{F}$, if $\mathcal{H}$ is a (ε,γ)-sparsifier of $\mathcal{G}$ we have*

$$\mathrm{Tr}(\mathbf{F}^\mathsf{T}\mathbf{L}_\mathcal{H}\mathbf{F}) \leq (1+\varepsilon)\,\mathrm{Tr}(\mathbf{F}^\mathsf{T}\mathbf{L}_\mathcal{G}\mathbf{F}) + \varepsilon\gamma k.$$

Therefore, a clustering that well separates the sparsifier will also separate well the true graph. Similarly we can obtain strong approximation guarantees for a variety of other Laplacian-based algorithms. Regularized problems such as LTR (Cortes et al., 2008), Laplacian Regularized Least Squares, and Laplacian SVM (Belkin et al., 2005), are of particular interest since the additive γ error is absorbed by the regularization, and it is possible to provide strong generalization guarantees.

5. Experiments

We empirically validate our theoretical findings by testing how (ε,γ)-sparsifiers can improve computational complexity without sacrificing final accuracy.

Dataset. We run experiments on the Amazon co-purchase graph (Sadhanala et al., 2016). This graph fits our setting: it cannot be generated from vectorial data, and is only artificially sparse, since the crawler that created it had no access to the true private co-purchase network held by Amazon. To compensate, Gleich & Mahoney (2015) use a densification procedure that given the graph adjacency matrix $\mathbf{A}_\mathcal{G}$, computes all k-step neighbours $\mathbf{A}_{\mathcal{G},k} = \sum_{s=1}^{k}\mathbf{A}_\mathcal{G}^s$. We make the graph unweighted for numerical stability. The final graph has $n = 334,863$ nodes and $m = 98,465,352$ edges, with an average degree of 294. We followed an approach similar to Sadhanala et al. (2016), and introduce a hand-designed smooth signal as a target. We then perform 2000 iterations of the power method to compute an approximation of the smallest eigenvector $\mathbf{v}_{\min}$, which is used as a smooth function over the graph.

Baselines. For all setups, we compute an "exact" solution (up to convergence error) using a fast linear solver. Computing this EXACT baseline requires $\mathcal{O}(m\log(n))$ time

Alg.	*Parameters*	$\lvert\mathcal{E}\rvert$ *(x10^6)*	*Err.* SSL$(l=346)$	*Err.* SSL$(l=672)$	*Err.* $D(\widetilde{\mathbf{f}})(\sigma=10^{-3})$	*Err.* $D(\widetilde{\mathbf{f}})$ $(\sigma=10^{-2})$
EXACT		98.5	0.312 ± 0.022	0.286 ± 0.010	0.067 ± 0.0004	0.756 ± 0.006
KN	$k = 60$	15.7	0.329 ± 0.0143	0.311 ± 0.027	0.172 ± 0.0004	0.822 ± 0.002
KN	$k = 90$	21.2	0.334 ± 0.024	0.311 ± 0.024	0.125 ± 0.0002	0.811 ± 0.003
DISRE	$\gamma=0, \overline{q}=100$	15	0.314 ± 0.0165	0.296 ± 0.015	0.068 ± 0.0003	0.758 ±0.005
DISRE	$\gamma=0, \overline{q}=150$	22.8	0.314 ± 0.0158	0.310 ± 0.024	0.068 ± 0.0004	0.756 ± 0.005
DISRE	$\gamma=10^3, \overline{q}=100$	7.3	–	–	0.072 ± 0.0003	0.789 ± 0.005
DISRE	$\gamma=10^2, \overline{q}=100$	11.8	–	–	0.068 ± 0.0002	0.772 ± 0.004
DISRE	$\gamma=10, \overline{q}=100$	14.4	–	–	0.068 ± 0.0004	0.760 ± 0.004

Table 1. Results for the SSL and the smoothing problems.

and $\mathcal{O}(m)$ space, and achieves the absolute best performance possible. Afterwards, we compare three different sparsification procedures to evaluate if they can accelerate computation while preserving accuracy. We run DISRE with different values of γ depending on the setting. For empirically strong heuristics, we attempted to uniformly subsample the edges, but at the sparsity level achieved by the other methods the uniformly sampled sparsifier is disconnected and highly inaccurate. Instead, we compare to the state-of-the-art k-neighbours (KN) heuristic by Sadhanala et al. (2016), which is just as fast as uniform sampling and more accurate in practice.

Experimental procedure. We repeat each experiment 10 times with different sparsifiers and report the average performance of $\widetilde{\mathbf{f}}$ on the specific task and its standard deviation. More details on experiments are given in the appendix.

5.1. Laplacian smoothing with Gaussian noise

We set $\mathbf{f}^* = \mathbf{v}_{\min}$ and test different levels of noise $\log_{10}(\sigma) \in \{-3, -2, -1, 0\}$. After constructing the sparsifier $\mathcal{H}$, we compute an approximate solution $\widetilde{\mathbf{f}}$ using LAPSMO (Eq. 2) with $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$. We measure the performance by the squared error $D(\widetilde{\mathbf{f}}) = \|\mathbf{f}^* - \widetilde{\mathbf{f}}\|_2^2$. As $\|\mathbf{f}^*\|_2^2 = \|\mathbf{v}_{\min}\|_2^2 = 1$, good values of $D(\widetilde{\mathbf{f}})$ should be below 1.

Accuracy. In the interest of space, in Tab. 1, we report results for $\sigma = \{0.001, 0.01\}$ and the best regularization λ for each method. We first notice that all sparsifiers are considerably smaller than the original graph, keeping only a small fraction of its edges. The smallest sparsifiers are obtained by DISRE when γ is large. The comparison with DISRE with $\gamma = 0$ (i.e., ε-sparsifier) confirms that the additive error translates into an extra compression of the resulting sparsifier. This also impacts the accuracy which degrades as γ increases. Nonetheless, we notice that while ε-sparsifiers perfectly match the accuracy of the exact method, even for large γ (and thus much smaller graph), DISRE still outperforms KN, which has a significantly worse accuracy. Finally, we note that for $\gamma = 0$, the impact of $\overline{q}$ is as expected: increasing $\overline{q}$ increases the size of the sparsifier and slightly improves the performance.

Computational complexity. All algorithms require 90s to load the graph from disk. The preprocessing phase of KN takes slightly less than 1`min`, while DISRE's takes 12`min` on 4 machines. For the solving step, EXACT is unsurprisingly the slowest, requiring 12`min` to compute an $\widehat{\mathbf{f}}$ solution. Both KN and (ε, γ)-sparsifiers require 1–2`min`, depending on the number of edges preserved. Overall, preprocessing the graph with DISRE before computing a solution does not introduce any overhead compared to EXACT (both take roughly 12`min`). We notice that while KN is overall faster, the time for DISRE could be easily reduced by increasing the number of parallel processes when computing effective resistances, and with a better network topology allowing point-to-point communication. Moreover, once we have access to an accurate ε-sparsifier, it is easier to solve problem repeatedly, e.g., to cross-validate regularization. For example, computing a solution for 4 different value of λ (see the appendix) is crucial for good performance, and requires 48`min` for EXACT and only 20`min` for DISRE. Finally, memory usage is reduced by a factor 3, as EXACT requires over 30GB of memory to execute while DISRE never exceeds 10GB. We expect these advantages to only grow larger as we scale to larger graphs.

5.2. SSL with harmonic function solution

We also test DISRE on a SSL problem. The labels are generated taking the sign of $\mathbf{f}^* = \mathbf{v}_{\min}$ and $l \in \{20, 346, 672, 1000\}$ labels are revealed. The labeled nodes are chosen at random so that 0 and 1 labels are balanced in the dataset. We run STABLE-HFS with $\lambda \in \{10^{-6}, 10^{-4}, 10^{-2}, 1\}$. In Tab. 1, we report results for $l = \{346, 672\}$ and the best λ for each method. We run DISRE with $\gamma = 0$ as STABLE-HFS does not have any regularization and ε-sparsifiers are preferable. The average size of the sparsifiers is the same as before, as they are agnostic to the learning task. Similar to the smoothing case, DISRE achieves a performance that closely approximates the exact solution, despite the significant compression of the original graph. Furthermore, the effectiveness of the ε-sparsifier returned by DISRE is confirmed by its comparison with KN, whose error is significantly worse. Finally, we notice that the computational analysis in the previous section holds for SSL as well. In fact, although the learning task is different, we use the same SSD solver to compute the HFS and thus the running time are comparable in the two tasks.

Acknowledgements: Experiments presented in this paper were carried out using the Grid'5000 testbed, supported by a scientific interest group hosted by Inria and including CNRS, RENATER and several Universities as well as other organizations (see https://www.grid5000.fr). The research presented was also supported by European CHIST-ERA project DELTA, French Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council, Inria and Otto-von-Guericke-Universität Magdeburg associated-team north-european project Allocate, and French National Research Agency projects ExTra-Learn (n.ANR-14-CE24-0010-01) and BoB (n.ANR-16-CE23-0003).

References

Akoglu, Leman, Tong, Hanghang, and Koutra, Danai. Graph based anomaly detection and description: a survey. *Data Mining and Knowledge Discovery*, 29(3): 626–688, may 2015. ISSN 1384-5810. doi: 10.1007/s10618-014-0365-y.

Alaoui, Ahmed El and Mahoney, Michael W. Fast randomized kernel methods with statistical guarantees. In *Neural Information Processing Systems*, 2015.

Belkin, Mikhail and Niyogi, Partha. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pp. 585–591, 2001.

Belkin, Mikhail, Matveeva, Irina, and Niyogi, Partha. Regularization and Semi-Supervised Learning on Large Graphs. In *Proceedings of COLT*, 2004.

Belkin, Misha, Niyogi, Partha, and Sindhwani, Vikas. On manifold regularization. In *AISTATS*, pp. 1, 2005.

Bühlmann, Peter and Van De Geer, Sara. *Statistics for high-dimensional data: methods, theory and applications*. Springer Science & Business Media, 2011.

Calandriello, Daniele, Lazaric, Alessandro, and Valko, Michal. Distributed sequential sampling for kernel matrix approximation. In *AISTATS*, 2017.

Chapelle, Olivier, Schlkopf, Bernhard, and Zien, Alexander. *Semi-Supervised Learning*. The MIT Press, 2010.

Cohen, Michael B., Musco, Cameron, and Pachocki, Jakub. Online Row Sampling. *arXiv:1604.05448 [cs]*, April 2016. URL http://arxiv.org/abs/1604.05448. arXiv: 1604.05448.

Cohen, Michael B, Musco, Cameron, and Musco, Christopher. Input sparsity time low-rank approximation via ridge leverage score sampling. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1758–1777. SIAM, 2017.

Cortes, Corinna, Mohri, Mehryar, Pechyony, Dmitry, and Rastogi, Ashish. Stability of transductive regression algorithms. In *Proceedings of ICML*, 2008.

Cucuringu, Mihai, Koutis, Ioannis, Chawla, Sanjay, Miller, Gary, and Peng, Richard. Simple and scalable constrained clustering: a generalized spectral method. In Gretton, Arthur and Robert, Christian C. (eds.), *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics*, volume 51 of *Proceedings of Machine Learning Research*, pp. 445–454, Cadiz, Spain, 09–11 May 2016. PMLR. URL http://proceedings.mlr.press/v51/cucuringu16.html.

Drineas, Petros and Mahoney, Michael W. Lectures on randomized numerical linear algebra. *CoRR*, abs/1712.08880, 2017. URL http://arxiv.org/abs/1712.08880.

Feldman, Dan, Schmidt, Melanie, and Sohler, Christian. Turning big data into tiny data: Constant-size coresets for k-means, pca and projective clustering. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pp. 1434–1453, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics. ISBN 978-1-611972-51-1. URL http://dl.acm.org/citation.cfm?id=2627817.2627920.

Ghashami, Mina, Liberty, Edo, Phillips, Jeff M, and Woodruff, David P. Frequent directions: Simple and deterministic matrix sketching. *SIAM Journal on Computing*, 45(5):1762–1792, 2016.

Gleich, David F. and Mahoney, Michael W. Using local spectral methods to robustify graph-based learning algorithms. In *SIGKDD*, pp. 359–368, 2015.

Jamakovic, A and Mieghem, P Van. The Laplacian spectrum of complex networks. *European Conference on Complex Systems*, pp. 1–6, 2006. URL http://repository.tudelft.nl/assets/uuid:abe61d93-4e25-41ab-90d4-2a55cf2982f5/TheLaplacianSpectrumofComplexNetworks.pdf.

Kelner, Jonathan A. and Levin, Alex. Spectral Sparsification in the Semi-streaming Setting. *Theory of Computing Systems*, 53(2):243–262, 2013.

Koutis, Ioannis and Xu, Shen Chen. Simple parallel and distributed algorithms for spectral graph sparsification. *ACM Transactions on Parallel Computing (TOPC)*, 3(2): 14, 2016.

Koutis, Ioannis, Miller, Gary L., and Peng, Richard. A nearly-m log n time solver for SDD linear systems. In *IEEE 52nd Annual Symposium on Foundations of Computer Science, FOCS*, pp. 590–598, 2011.

Kyng, Rasmus and Sachdeva, Sushant. Approximate gaussian elimination for laplacians-fast, sparse, and simple. In *Foundations of Computer Science (FOCS), 2016 IEEE 57th Annual Symposium on*, pp. 573–582. IEEE, 2016.

Kyng, Rasmus, Pachocki, Jakub, Peng, Richard, and Sachdeva, Sushant. A framework for analyzing resparsification algorithms. In *STOC*, 2016.

Lee, James R., Gharan, Shayan Oveis, and Trevisan, Luca. Multiway spectral partitioning and higher-order cheeger inequalities. *J. ACM*, 61(6):37:1–37:30, December 2014. ISSN 0004-5411. doi: 10.1145/2665063. URL http://doi.acm.org/10.1145/2665063.

Levy, Haim. *Stochastic dominance: Investment decision making under uncertainty*. Springer, 2015.

Sadhanala, Veeru, Wang, Yu-Xiang, and Tibshirani, Ryan. Graph sparsification approaches for laplacian smoothing. In *Artificial Intelligence and Statistics*, pp. 1250–1259, 2016.

Samukhin, A. N., Dorogovtsev, S. N., and Mendes, J. F. F. Laplacian spectra of, and random walks on, complex networks: Are scale-free architectures really important? *Physical Review E*, 77(3):036115, mar 2008. ISSN 1539-3755. doi: 10.1103/PhysRevE.77.036115. URL http://arxiv.org/abs/0706.1176http://dx.doi.org/10.1103/PhysRevE.77.036115https://link.aps.org/doi/10.1103/PhysRevE.77.036115.

Spielman, Daniel A. and Srivastava, Nikhil. Graph sparsification by effective resistances. *SIAM Journal on Computing*, 40(6), 2011.

Spielman, Daniel A. and Teng, Shang-Hua. Spectral sparsification of graphs. *SIAM Journal on Computing*, 40 (4):981–1025, 2011. URL http://epubs.siam.org/doi/abs/10.1137/08074489X.

Tropp, Joel A. Freedman's inequality for matrix martingales. *Electron. Commun. Probab*, 16:262–270, 2011.

Tropp, Joel A. An introduction to matrix concentration inequalities. *Foundations and Trends® in Machine Learning*, 8(1-2):1–230, 2015. ISSN 1935-8237. doi: 10.1561/2200000048.

Von Luxburg, Ulrike. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007.

Von Luxburg, Ulrike, Radl, Agnes, and Hein, Matthias. Hitting and commute times in large random neighborhood graphs. *Journal of Machine Learning Research*, 15(1): 1751–1798, 2014.

Zhan, Choujun, Chen, Guanrong, and Yeung, Lam F. On the distributions of Laplacian eigenvalues versus node degrees in complex networks. *Physica A: Statistical Mechanics and its Applications*, 389(8):1779–1788, 2010.

Zhu, Xiaojin, Ghahramani, Zoubin, and Lafferty, John. Semi-Supervised Learning Using Gaussian Fields and Harmonic Functions. In *Proceedings of ICML*, 2003.

A. Further Details on Experiments

Software and hardware All our code is implemented in Julia, and runs on a cluster of 4 machines with 128 GB of RAM and a 10-core Xeon E5-2630. The distributed computation is achieved using a sub-optimal but simple producer-consumer queue.

Linear solver We use Kyng & Sachdeva (2016)'s approximate Gaussian elimination scheme, provided by the `Laplacians.jl`[6] package.

Details on DiSRe. To compute all (ε, γ) sparsifiers we use DiSRe after splitting the input graph in 8 sub-graphs[7], resulting in 3 rounds of resparsifications using 4 machines. For each resparsification, we compute the effective resistance in parallel on each machine using 10 processes[8]

Additional results. For completeness, we also provide two OpenDocument spreadsheets containing all the combinations of hyper-parameters $(\varepsilon, \gamma, \overline{q}, k, \sigma, l)$ used, one for the Laplacian smoothing experiment (`extra_results_smoothing.ods`) and one for the SSL experiment (`extra_results_ssl,ods`).

B. Proofs for Sec. 4

We will often use the following reformulation of 1

Proposition 3. *A sub-graph $\mathcal{H}$ is a (ε, γ)-sparsifier of $\mathcal{G}$ iff*

$$(1-\varepsilon)\mathbf{L}_{\mathcal{G}} - \varepsilon\gamma\mathbf{I} \preceq \mathbf{L}_{\mathcal{H}} \preceq (1+\varepsilon)\mathbf{L}_{\mathcal{G}} + \varepsilon\gamma\mathbf{I} \qquad \Leftrightarrow \qquad \|(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{-1/2}(\mathbf{L}_{\mathcal{H}} - \mathbf{L}_{\mathcal{G}})(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{-1/2}\|_2^2 \leq \varepsilon$$

Proof. Proof of Thm. 3 We need to bound the distance between $\widetilde{\mathbf{f}}$ and $\widehat{\mathbf{f}}$. Using the definition, the fact that $\mathbf{A}^{-1} - \mathbf{B}^{-1} = \mathbf{B}^{-1}(\mathbf{B} - \mathbf{A})\mathbf{A}^{-1}$ and collecting $(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{1/2}$ we have

$$\begin{aligned}
\|\widetilde{\mathbf{f}} - \widehat{\mathbf{f}}\|_2^2 &= \|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1} - \lambda\mathbf{L}_{\mathcal{G}} + \mathbf{I})^{-1})\mathbf{y}\|_2^2 = \|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}(\lambda\mathbf{L}_{\mathcal{H}} - \lambda\mathbf{L}_{\mathcal{G}})(\lambda\mathbf{L}_{\mathcal{G}} + \mathbf{I})^{-1})\mathbf{y}\|_2^2 \\
&= \lambda^2\|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}(\lambda\mathbf{L}_{\mathcal{H}} - \mathbf{L}_{\mathcal{G}})(\lambda\mathbf{L}_{\mathcal{G}} + \mathbf{I})^{-1})\mathbf{y}\|_2^2 \\
&= \lambda^2\|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{1/2}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{-1/2}(\mathbf{L}_{\mathcal{H}} - \mathbf{L}_{\mathcal{G}})(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{-1/2}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{1/2}\widehat{\mathbf{f}}\|_2^2
\end{aligned}$$

Then using Prop. 3

$$\begin{aligned}
&\lambda^2\|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{1/2}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{-1/2}(\mathbf{L}_{\mathcal{H}} - \mathbf{L}_{\mathcal{G}})(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{-1/2}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{1/2}\widehat{\mathbf{f}}\|_2^2 \\
&\leq \varepsilon^2\lambda^2\|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})^{1/2}\|_2^2\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \\
&= \frac{\varepsilon^2}{1-\varepsilon}\lambda^2\|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}((1-\varepsilon)\mathbf{L}_{\mathcal{G}} - \varepsilon\gamma\mathbf{I} + \gamma\mathbf{I})(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}\|_2\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \\
&\leq \frac{\varepsilon^2}{1-\varepsilon}\lambda^2\|(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}(\mathbf{L}_{\mathcal{H}} + \gamma\mathbf{I})(\lambda\mathbf{L}_{\mathcal{H}} + \mathbf{I})^{-1}\|_2\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \\
&= \frac{\varepsilon^2}{1-\varepsilon}\lambda^2\max_i\left\{\frac{\lambda_i(\mathbf{L}_{\mathcal{H}}) + \gamma}{(\lambda\lambda_i(\mathbf{L}_{\mathcal{H}}) + 1)^2}\right\}\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \\
&= \frac{\varepsilon^2}{1-\varepsilon}\lambda\max_i\left\{\frac{\lambda\lambda_i(\mathbf{L}_{\mathcal{H}})}{(\lambda\lambda_i(\mathbf{L}_{\mathcal{H}}) + 1)^2} + \frac{\lambda\gamma}{(\lambda\lambda_i(\mathbf{L}_{\mathcal{H}}) + 1)^2}\right\}\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \\
&\leq \frac{\varepsilon^2}{1-\varepsilon}\lambda\left(\max_i\left\{\frac{\lambda\lambda_i(\mathbf{L}_{\mathcal{H}})}{(\lambda\lambda_i(\mathbf{L}_{\mathcal{H}}) + 1)^2}\right\} + \max_i\left\{\frac{\lambda\gamma}{(\lambda\lambda_i(\mathbf{L}_{\mathcal{H}}) + 1)^2}\right\}\right)\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \\
&\leq \frac{\varepsilon^2}{1-\varepsilon}\lambda\left(1/4 + \lambda\gamma\right)\widehat{\mathbf{f}}^{\mathsf{T}}(\mathbf{L}_{\mathcal{G}} + \gamma\mathbf{I})\widehat{\mathbf{f}} \leq \frac{\varepsilon^2}{1-\varepsilon}((1/4 + \lambda\gamma)\,\lambda\widehat{\mathbf{f}}^{\mathsf{T}}\mathbf{L}_{\mathcal{G}}\widehat{\mathbf{f}} + (1/4 + \lambda\gamma)\,\lambda\gamma\mathbf{I}),
\end{aligned}$$

which concludes the proof. □

[6] `http://github.com/danspielman/Laplacians.jl`

[7] to guarantee that each of the sub-graphs is defined on the same set of nodes, we pre-construct a spanning tree of $\mathcal{G}$ and include it in each of the sub-graphs. Note that the analysis holds even if each sub-graph is disconnected from the others. We choose this approach to avoid complicating the code with additional searches of connected components in the sparsifiers

[8] `Laplacian.jl` is strictly single-threaded, so we use multiple processes on a single machine. Faster runtime and lower memory usage could be achieved by sharing memory with threads.

Proof of Thm. 2. **Step 1 (generalization of stable algorithms).** Let β be the stability of STABLE-HFS when using the sparsified Laplacian $L_\mathcal{H}$ in place of $L_\mathcal{G}$. Then using the result in (Cortes et al., 2008), we have that with probability at least $1-\delta$ (w.r.t. the randomness of the labeled set $\mathcal{S}$) the solution $\widetilde{\mathbf{f}}$ satisfies

$$R(\widetilde{\mathbf{f}}) \leq \widehat{R}(\widetilde{\mathbf{f}}) + \beta + \Big(2\beta + \frac{c^2(l+u)}{lu}\Big)\sqrt{\frac{\pi(l,u)\log(1/\delta)}{2}}.$$

In order to obtain the final result we first derive an upper bound on the stability β, and relate the empirical error of $\widetilde{\mathbf{f}}_{\text{STA}}$ to the one of $\widehat{\mathbf{f}}_{\text{STA}}$.

Furthermore, it can be shown that if we center the vector of labels $\widetilde{\mathbf{y}}_\mathcal{S} = \mathbf{y}_\mathcal{S} - \overline{\mathbf{y}}_\mathcal{S}$, with $\overline{\mathbf{y}} = \frac{1}{l}\mathbf{y}_\mathcal{S}^\mathsf{T}\mathbf{1}$, then the solution of STABLE-HFS can be rewritten in closed form as $\widehat{\mathbf{f}}_{\text{STA}} = (\gamma l\mathbf{L}_\mathcal{G} + \mathbf{I}_\mathcal{S})^+(\widetilde{\mathbf{y}}_\mathcal{S} - \mu\mathbf{1}) = \big(\mathbf{P}(\gamma l\mathbf{L}_\mathcal{G} + \mathbf{I}_\mathcal{S})\big)^+\widetilde{\mathbf{y}}_\mathcal{S}$.

Step 2 (stability). The bound on the stability follows similar steps as in the analysis of STABLE-HFS in (Belkin et al., 2004) integrated with the properties of spectral sparsifiers reported in Definition 1. Let $\mathcal{S}$ and $\mathcal{S}'$ be two labeled sets only differing by one element and $\widetilde{\mathbf{f}}$ and $\widetilde{\mathbf{f}}'$ be the solutions obtained by running STABLE-HFS using $L_\mathcal{H}$ and $\mathcal{S}$ and $\mathcal{S}'$ respectively. Without loss of generality, we assume that $\mathbf{I}_\mathcal{S}(l,l) = 1$ and $\mathbf{I}_\mathcal{S}(l+1,l+1) = 0$, and the opposite for $\mathbf{I}_{\mathcal{S}'}$. The original proof in (Cortes et al., 2008) showed that the stability β can be bounded as $\beta \leq \|\widetilde{\mathbf{f}} - \widetilde{\mathbf{f}}'\|$. In the following we show that the difference between the solutions $\widetilde{\mathbf{f}}$ and $\widetilde{\mathbf{f}}'$, and thus the stability of the algorithm, is strictly related to eigenvalues of the sparse graph $\mathcal{H}$. Let $\mathbf{A} = \mathbf{P}(l\gamma\mathbf{L}_\mathcal{H} + \mathbf{I}_\mathcal{S})$ and $\mathbf{B} = \mathbf{P}(l\gamma\mathbf{L}_\mathcal{H} + \mathbf{I}_{\mathcal{S}'})$, we remind that if the labels are centered, the solutions of STABLE-HFS can be conveniently written as $\widetilde{\mathbf{f}} = \mathbf{A}^{-1}\widetilde{\mathbf{y}}_\mathcal{S}$ and $\widetilde{\mathbf{f}}' = \mathbf{B}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'}$. As a result, the difference between the solutions can be written as

$$\|\widetilde{\mathbf{f}} - \widetilde{\mathbf{f}}'\| = \|\mathbf{A}^{-1}\widetilde{\mathbf{y}}_\mathcal{S} - \mathbf{B}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'}\| \leq \|\mathbf{A}^{-1}(\widetilde{\mathbf{y}}_\mathcal{S} - \widetilde{\mathbf{y}}_{\mathcal{S}'})\| + \|\mathbf{A}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'} - \mathbf{B}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'}\|. \tag{8}$$

Let us consider any vector $\mathbf{f} \in \mathcal{F}$, since the null space of a Laplacian $\mathbf{L}_\mathcal{H}$ is the one vector $\mathbf{1}$ and $\mathbf{P} = \mathbf{L}_\mathcal{H}\mathbf{L}_\mathcal{H}^+$, then $\mathbf{P}\mathbf{f} = \mathbf{f}$. Thus we have

$$\|\mathbf{P}(l\gamma\mathbf{L}_\mathcal{H} + \mathbf{I}_\mathcal{S})\mathbf{f}\| \overset{(1)}{\geq} \|\mathbf{P}l\gamma\mathbf{L}_\mathcal{H}\mathbf{f}\| - \|\mathbf{P}\mathbf{I}_\mathcal{S}\mathbf{f}\| \overset{(2)}{\geq} \|\mathbf{P}l\gamma\mathbf{L}_\mathcal{H}\mathbf{f}\| - \|\mathbf{f}\| \overset{(3)}{\geq} (l\gamma\lambda_2(\mathcal{H}) - 1)\|\mathbf{f}\| \tag{9}$$

where (1) follows from the triangle inequality and (2) follows from the fact that $\|\mathbf{P}\mathbf{I}_\mathcal{S}\mathbf{f}\| \leq \|\mathbf{f}\|$ since the largest eigenvalue of the project matrix $\mathbf{P}$ is one and the norm of $\mathbf{f}$ restricted on $\mathcal{S}$ is smaller than the norm of $\mathbf{f}$. Finally (3) follows from the fact that $\|\mathbf{P}\mathbf{L}_\mathcal{H}\mathbf{f}\| = \|\mathbf{L}_\mathcal{H}\mathbf{L}_\mathcal{H}^+\mathbf{L}_\mathcal{H}\mathbf{f}\| = \|\mathbf{L}_\mathcal{H}\mathbf{f}\|$ and since $\mathbf{f}$ is orthogonal to the null space of $\mathbf{L}_\mathcal{H}$ then $\|\mathbf{L}_\mathcal{H}\mathbf{f}\| \geq \lambda_2(\mathcal{H})\|\mathbf{f}\|$, where $\lambda_2(\mathcal{H})$ is the smallest non-zero eigenvalue of $\mathbf{L}_\mathcal{H}$. At this point we can exploit the spectral guarantees of the sparsified Laplacian $L_\mathcal{H}$ and we have that $\lambda_2(\mathcal{H}) \geq (1-\varepsilon)\lambda_2(\mathcal{G})$. As a result, we have an upper-bound on the spectral radius of the inverse operator $(\mathbf{P}(l\gamma\mathbf{L}_\mathcal{H} + \mathbf{I}_\mathcal{S}))^{-1}$ and thus

$$\|\mathbf{A}^{-1}(\mathbf{y}_\mathcal{S} - \mathbf{y}_{\mathcal{S}'})\| \leq \frac{4M}{l\gamma(1-\varepsilon)\lambda_2(\mathcal{G}) - 1},$$

where the first step follows from Eq. 9 since both $\widetilde{\mathbf{y}}_\mathcal{S}$ and $\widetilde{\mathbf{y}}_{\mathcal{S}'}$ are centered and thus $(\mathbf{y}_\mathcal{S} - \mathbf{y}_{\mathcal{S}'}) \in \mathcal{F}$, and the second step is obtained by bounding $\|\widetilde{\mathbf{y}}_\mathcal{S} - \widetilde{\mathbf{y}}_{\mathcal{S}'}\| \leq \|\mathbf{y}_\mathcal{S} - \mathbf{y}_{\mathcal{S}'}\| + \|\overline{\mathbf{y}}_\mathcal{S} - \overline{\mathbf{y}}_{\mathcal{S}'}\| \leq 4M$. The second term in Eq. 8 can be bounded as

$$\begin{aligned}\|\mathbf{A}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'} - \mathbf{B}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'}\| &= \|\mathbf{B}^{-1}(B-A)\mathbf{A}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'}\| \\ &= \|\mathbf{B}^{-1}\mathbf{P}(\mathbf{I}_\mathcal{S} - \mathbf{I}_{\mathcal{S}'})\mathbf{A}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}'}\| \leq \frac{1.5M\sqrt{l}}{(l\gamma(1-\varepsilon)\lambda_2(\mathcal{G}) - 1)^2},\end{aligned}$$

where we used $\|\widetilde{\mathbf{y}}_{\mathcal{S}'}\| \leq \|\mathbf{y}_{\mathcal{S}'}\| + \|\overline{\mathbf{y}}_{\mathcal{S}'}\| \leq 2M\sqrt{l}$, $\|\mathbf{P}(\mathbf{I}_\mathcal{S} - \mathbf{I}_{\mathcal{S}'})\| \leq \sqrt{2} < 1.5$ and we applied Eq. 9 twice. Putting it all together we obtain the stated bound.

Step 3 (empirical error). The other element effected by the sparsification is the empirical error $\widehat{R}(\widetilde{\mathbf{f}})$. We first recall that $\mathbf{P} = \mathbf{L}_\mathcal{G}^+\mathbf{L}_\mathcal{G} = \mathbf{L}_\mathcal{G}^{-1/2}\mathbf{L}_\mathcal{G}\mathbf{L}_\mathcal{G}^{-1/2}$ (and equivalently with $\mathcal{G}$ replaced by $\mathcal{H}$) and we introduce $\widetilde{\mathbf{P}} = \mathbf{L}_\mathcal{G}^{-1/2}\mathbf{L}_\mathcal{H}\mathbf{L}_\mathcal{G}^{-1/2}$ Let

$\widetilde{\mathbf{A}} = \mathbf{P}(l\gamma\mathbf{L}_{\mathcal{H}} + \mathbf{I}_{\mathcal{S}})$, $\widehat{\mathbf{A}} = \mathbf{P}(l\gamma\mathbf{L}_{\mathcal{G}} + \mathbf{I}_{\mathcal{S}})$, then rewrite the empirical error as

$$\begin{aligned}
\widehat{R}(\widetilde{\mathbf{f}}) &= \frac{1}{l}\|\mathbf{I}_{\mathcal{S}}\widetilde{\mathbf{f}} - \mathbf{I}_{\mathcal{S}}\widehat{\mathbf{f}} + \mathbf{I}_{\mathcal{S}}\widehat{\mathbf{f}} - \widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\leq \tfrac{1}{l}\|\mathbf{I}_{\mathcal{S}}\widehat{\mathbf{f}} - \widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 + \tfrac{1}{l}\|\mathbf{I}_{\mathcal{S}}\widetilde{\mathbf{f}} - \mathbf{I}_{\mathcal{S}}\widehat{\mathbf{f}}\|^2 \\
&\leq \widehat{R}(\widehat{\mathbf{f}}) + \tfrac{1}{l}\|\mathbf{I}_{\mathcal{S}}(\widetilde{\mathbf{A}}^{-1} - \widehat{\mathbf{A}}^{-1})\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\leq \widehat{R}(\widehat{\mathbf{f}}) + \tfrac{1}{l}\|\widehat{\mathbf{A}}^{-1}(\widehat{\mathbf{A}} - \widetilde{\mathbf{A}})\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&= \widehat{R}(\widehat{\mathbf{f}}) + \tfrac{l^2\gamma^2}{l}\|\widehat{\mathbf{A}}^{-1}(\mathbf{P}(\mathbf{L}_{\mathcal{G}} - \mathbf{L}_{\mathcal{H}}))\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&= \widehat{R}(\widehat{\mathbf{f}}) + l\gamma^2\|\widehat{\mathbf{A}}^{-1}(\mathbf{P}(\mathbf{L}_{\mathcal{G}} - \mathbf{L}_{\mathcal{H}})\mathbf{P})\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2
\end{aligned}$$

Where in the last passage we use $\mathbf{L}_{\mathcal{G}}\mathbf{P} = \mathbf{L}_{\mathcal{G}}$ and $\mathbf{L}_{\mathcal{H}}\mathbf{P} = \mathbf{L}_{\mathcal{H}}$. To bound the second term we derive

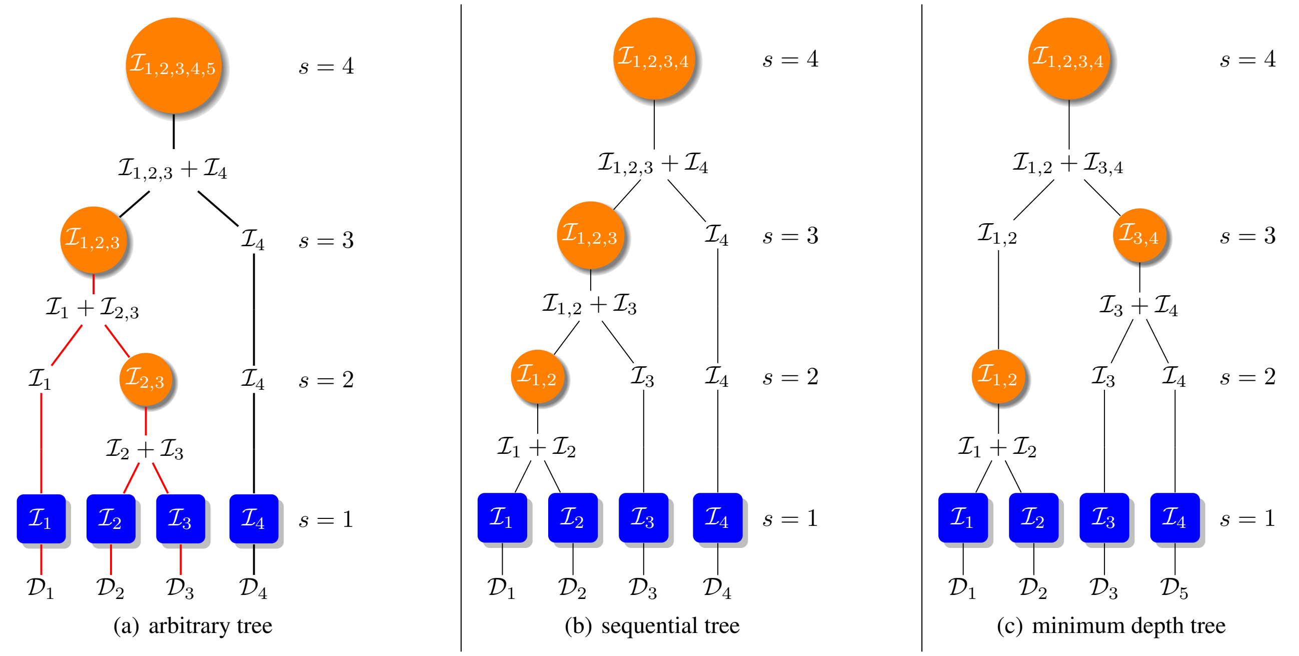
$$\begin{aligned}
&\|\widehat{\mathbf{A}}^{-1}\mathbf{P}(\mathbf{L}_{\mathcal{G}} - \mathbf{L}_{\mathcal{H}})\mathbf{P}\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\overset{(1)}{=} \|\widehat{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{L}_{\mathcal{G}}^{-1/2}(\mathbf{L}_{\mathcal{G}} - \mathbf{L}_{\mathcal{H}})\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{L}_{\mathcal{G}}^{1/2}\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\overset{(2)}{=} \|\widehat{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{P}\mathbf{L}_{\mathcal{G}}^{-1/2}(\mathbf{L}_{\mathcal{G}} - \mathbf{L}_{\mathcal{H}})\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{P}\mathbf{L}_{\mathcal{G}}^{1/2}\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\overset{(3)}{=} \|\widehat{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{P}(\mathbf{P} - \widetilde{\mathbf{P}})\mathbf{P}\mathbf{L}_{\mathcal{G}}^{1/2}\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\leq \|\widehat{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{P}\|^2\|\mathbf{P} - \widetilde{\mathbf{P}}\|^2\|\mathbf{P}\mathbf{L}_{\mathcal{G}}^{1/2}\widetilde{\mathbf{A}}^{-1}\|^2\|\widetilde{\mathbf{y}}_{\mathcal{S}}\|^2 \\
&\overset{(4)}{\leq} lM^2\varepsilon^2\|\widehat{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{P}\|^2\|\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{P}\|^2
\end{aligned}$$

where in (1) and (2) we use the definition of $\mathbf{P}$, in (3) we use the definition of $\widetilde{\mathbf{P}}$, while in (4) we use the fact that Def. 1 implies that $(1-\varepsilon)\mathbf{P} \preceq \widetilde{\mathbf{P}} \preceq (1+\varepsilon)\mathbf{P}$ and thus the largest eigenvalue of $\mathbf{P} - \widetilde{\mathbf{P}}$ is ε and $\|\mathbf{P} - \widetilde{\mathbf{P}}\|^2 \leq \varepsilon^2$. We need now to bound $\|\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{P}\|^2 = \|\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\|^2$. From the definition of spectral norm

$$\begin{aligned}
\|\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\|^2 &= \max_{\|\mathbf{x}\|=1} \mathbf{x}^{\mathsf{T}}\mathbf{L}_{\mathcal{G}}^{1/2}\widetilde{\mathbf{A}}^{-1}\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}^{1/2}\mathbf{x} = \max_{\|\mathbf{x}\|=1} \mathbf{x}^{\mathsf{T}}\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{G}}\widetilde{\mathbf{A}}^{-1}\mathbf{x} \\
&\leq \frac{1}{1-\varepsilon}\max_{\|\mathbf{x}\|=1} \mathbf{x}^{\mathsf{T}}\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{H}}\widetilde{\mathbf{A}}^{-1}\mathbf{x} = \frac{1}{1-\varepsilon}\|\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{H}}^{1/2}\|^2 = \frac{1}{1-\varepsilon}\|\widetilde{\mathbf{A}}^{-1}\mathbf{L}_{\mathcal{H}}^{1/2}\mathbf{P}\|^2
\end{aligned}$$

From the definition of spectral norm, we have Similarly to Eq. 9 finding a lower bound on $\|\widetilde{\mathbf{A}}\mathbf{L}_{\mathcal{H}}^{-1/2}\mathbf{P}\mathbf{x}\|$ for all $\mathbf{x}$ is equivalent to find a lower bound for all $\mathbf{f} \in \mathcal{F}$ to

$$\begin{aligned}
&\|\mathbf{P}(l\gamma\mathbf{L}_{\mathcal{H}} + \mathbf{I}_{\mathcal{S}})\mathbf{L}_{\mathcal{H}}^{-1/2}\mathbf{f}\| \\
&\geq \|\mathbf{P}l\gamma\mathbf{L}_{\mathcal{H}}\mathbf{L}_{\mathcal{H}}^{-1/2}\mathbf{f}\| - \|\mathbf{P}\mathbf{I}_{\mathcal{S}}\mathbf{L}_{\mathcal{H}}^{-1/2}\mathbf{f}\| \\
&\geq \|\mathbf{P}l\gamma\mathbf{L}_{\mathcal{H}}\mathbf{L}_{\mathcal{H}}^{-1/2}\mathbf{f}\| - \|\mathbf{L}_{\mathcal{H}}^{-1/2}\mathbf{f}\| \\
&\geq \left(l\gamma\sqrt{\lambda_2(\mathcal{H})} - \frac{1}{\sqrt{\lambda_2(\mathcal{H})}}\right)\|\mathbf{f}\| \\
&\geq \frac{1}{\sqrt{\lambda_2(\mathcal{H})}}\left(l\gamma\lambda_2(\mathcal{H}) - 1\right)\|\mathbf{f}\| \\
&\geq \frac{1}{\sqrt{(1+\varepsilon)\lambda_2(\mathcal{G})}}\left(l\gamma(1-\varepsilon)\lambda_2(\mathcal{G}) - 1\right)\|\mathbf{f}\|
\end{aligned}$$


$\mathcal{I}_{1,2,3,4,5}$
$s=4$
$\mathcal{I}_{1,2,3}+\mathcal{I}_4$
$\mathcal{I}_{1,2,3}$
$\mathcal{I}_4$
$s=3$
$\mathcal{I}_1+\mathcal{I}_{2,3}$
$\mathcal{I}_1$
$\mathcal{I}_{2,3}$
$\mathcal{I}_4$
$s=2$
$\mathcal{I}_2+\mathcal{I}_3$
$\mathcal{I}_1$ $\mathcal{I}_2$ $\mathcal{I}_3$ $\mathcal{I}_4$
$s=1$
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_4$
$\mathcal{I}_{1,2,3,4}$
$s=4$
$\mathcal{I}_{1,2,3}+\mathcal{I}_4$
$\mathcal{I}_{1,2,3}$
$\mathcal{I}_4$
$s=3$
$\mathcal{I}_{1,2}+\mathcal{I}_3$
$\mathcal{I}_{1,2}$
$\mathcal{I}_3$
$\mathcal{I}_4$
$s=2$
$\mathcal{I}_1+\mathcal{I}_2$
$\mathcal{I}_1$ $\mathcal{I}_2$ $\mathcal{I}_3$ $\mathcal{I}_4$
$s=1$
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_4$
$\mathcal{I}_{1,2,3,4}$
$s=4$
$\mathcal{I}_{1,2}+\mathcal{I}_{3,4}$
$\mathcal{I}_{1,2}$
$\mathcal{I}_{3,4}$
$s=3$
$\mathcal{I}_3+\mathcal{I}_4$
$\mathcal{I}_{1,2}$
$\mathcal{I}_3$
$\mathcal{I}_4$
$s=2$
$\mathcal{I}_1+\mathcal{I}_2$
$\mathcal{I}_1$ $\mathcal{I}_2$ $\mathcal{I}_3$ $\mathcal{I}_4$
$s=1$
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_5$

(a) arbitrary tree

(b) sequential tree

(c) minimum depth tree

Figure 2. Merge trees for Algorithm 1.

Similarly, we can show that

$$\begin{aligned}
&\|\mathbf{P}(l\gamma\mathbf{L}_{\mathcal{G}}+\mathbf{I}_{\mathcal{S}})\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{f}\| \\
\geq&\|\mathbf{P}l\gamma\mathbf{L}_{\mathcal{G}}\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{f}\|-\|\mathbf{P}\mathbf{I}_{\mathcal{S}}\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{f}\| \\
\geq&\|\mathbf{P}l\gamma\mathbf{L}_{\mathcal{G}}\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{f}\|-\|\mathbf{L}_{\mathcal{G}}^{-1/2}\mathbf{f}\| \\
\geq&\left(l\gamma\frac{\lambda_2(\mathcal{G})}{\sqrt{\lambda_2(\mathcal{G})}}-\frac{1}{\sqrt{\lambda_2(\mathcal{G})}}\right)\|\mathbf{f}\| \\
\geq&\frac{1}{\sqrt{\lambda_2(\mathcal{G})}}\left(l\gamma\lambda_2(\mathcal{G})-1\right)\|\mathbf{f}\| \\
\geq&\frac{1}{\sqrt{(1+\varepsilon)\lambda_2(\mathcal{G})}}\left(l\gamma(1-\varepsilon)\lambda_2(\mathcal{G})-1\right)\|\mathbf{f}\|
\end{aligned}$$

Taking this and putting all together shows

$$\widehat{R}(\widetilde{\mathbf{f}})\leq\widehat{R}(\widehat{\mathbf{f}})+\frac{1}{1-\varepsilon}\frac{(1+\varepsilon)^2\varepsilon^2l^2\gamma^2\lambda_2(\mathcal{G})^2M^2}{(l\gamma(1-\varepsilon)\lambda_2(\mathcal{G})-1)^4}$$

Combining the three steps above concludes the proof.

□

C. Proof of Thm. 1

The proof of Thm. 1 draws heavily from the analysis of SQUEAK. In particular, Alg. 1 is an instantiation of SQUEAK to the special case of graph sparsification. While SQUEAK's analysis from Calandriello et al. (2017) holds in general, in DISRE we can exploit the specific structure of graph Laplacians to perform a few optimizations.

We begin by describing more in detail some notation introduced in the main paper and necessary for this proof.

Merge trees We first formalize the random process induced by Alg. 1.

We partition $\mathcal{G}$ into k disjoint sub-graphs $\mathcal{G}_i$ of size n_i, such that $\mathcal{G} = \cup_{e=1}^{k} \mathcal{G}_i$. For each sub-graph $\mathcal{G}_i$, we construct an initial sparsifier $\mathcal{H}_{\{1,i\}} = \{(j, \widetilde{p}_{0,i} = 1, q_{0,i} = \overline{q}) : j \in \mathcal{G}_i\}$ by inserting all edges from $\mathcal{G}_i$ into $\mathcal{H}_{1,i}$ with weight $\widetilde{p}_{0,i} = 1$ and number of copies $q_{0,i} = \overline{q}$. It is easy to see that $\mathcal{H}_{\{1,i\}}$ is an $(0,0)$-accurate sparsifier, and we can split the graph in small enough sub-graphs to make sure that it can be easily stored and manipulated in memory. Afterwards, the initial sparsifiers $\mathcal{H}_{\{1,i\}}$ are included into the sparsifier pool $\mathcal{S}_1$.

At iteration h, the inner loop of Alg. 1 arbitrarily chooses two sparsifiers from $\mathcal{S}_h$ and merges them into a new sparsifier. Any arbitrary sequence of merges can be described by a full binary tree, i.e., a binary tree where each node is either a leaf or has exactly two children. Figure 2 shows several different merge trees corresponding to different choices for the order of the merges. Note that starting from k leaves, a full binary tree will always have exactly $k-1$ internal nodes. Therefore, regardless of the structure of the merge tree, we can always transform it into a tree of depth k, with all the initial sparsifiers $\mathcal{H}_{1,i}$ as leaves on its deepest layer. After this transformation, we index the tree nodes using their height (longest path from the node to a leaf, also defined as depth of the tree minus depth of the node), where leaves have height 1 and the root has height k. We can also see that at each layer, there is a single sparsifier merge, and the size of $\mathcal{S}_h$ (number of sparsifiers present at layer h) is $|\mathcal{S}_h| = k - h + 1$. Therefore, a node corresponding to a sparsifier is uniquely identified with two indices $\{h, l\}$, where h is the height of the layer and $l \leq |\mathcal{S}_h|$ is the index of the node in the layer. For example, in Figure 2(a), the node containing $\mathcal{H}_{1,2,3}$ is indexed as $\{3, 1\}$, and the highest node containing $\mathcal{H}_4$ is indexed as $\{3, 2\}$.

We also define the graph $\mathcal{G}_{\{h,l\}}$ as the union of all sub-graph $\mathcal{G}_{l'}$ that are reachable from node $\{h, l\}$ as leaves. For example, in Fig. 2(a), sparsifier $\mathcal{H}_{1,2,3}$ in node $\{3, 1\}$ is constructed starting from all edges in $\mathcal{G}_{\{3,1\}} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \mathcal{G}_3$, where we highlight in red the descendant tree. We now define $\mathbf{L}^h$ as the block diagonal matrix where each diagonal block $\mathbf{L}_{\mathcal{G}_{\{h,l\}}}$ is the Laplacian constructed on $\mathcal{G}_{\{h,l\}}$. Without loss of generality, we will assume that each of the sub-graphs $\mathcal{G}_i$ is connected and spans all the n nodes in the graph. This simplifies the notation for the $\mathbf{L}_{\mathcal{G}_{\{h,l\}}}$ matrices, making them all $n \times n$ matrices. If this is not the case, the whole proof still follows through with different number of nodes $n_{\{h,l\}}$ for each $\mathcal{G}_{\{h,l\}}$. Again, from Fig. 2, $\mathbf{L}^3$ is a $2n \times 2n$ matrix with two blocks on the diagonal, a first $n \times n$ block $\mathbf{L}_{\mathcal{G}_{\{3,1\}}}$ constructed on $\mathcal{G}_{\{3,1\}} = \mathcal{G}_1 \cup \mathcal{G}_2 \cup \mathcal{G}_3$, and a second $n \times n$ block $\mathbf{L}_{\mathcal{G}_{\{3,2\}}}$ constructed on $\mathcal{G}_{\{3,2\}} = \mathcal{G}_4$. Similarly, we can combine Def. 1 and Prop. 3 to define

$$\mathbf{P}_{\{h,l\}} \triangleq (\mathbf{L}_{\mathcal{G}_{\{h,l\}}} + \gamma\mathbf{I})^{-1/2}\mathbf{L}_{\mathcal{G}_{\{h,l\}}}(\mathbf{L}_{\mathcal{G}_{\{h,l\}}} + \gamma\mathbf{I})^{-1/2}, \quad \widetilde{\mathbf{P}}_{\{h,l\}} \triangleq (\mathbf{L}_{\mathcal{G}_{\{h,l\}}} + \gamma\mathbf{I})^{-1/2}\mathbf{L}_{\mathcal{H}_{\{h,l\}}}(\mathbf{L}_{\mathcal{G}_{\{h,l\}}} + \gamma\mathbf{I})^{-1/2},$$

and have $\mathbf{P}^h$ as a block diagonal projection matrix, where each block $\mathbf{P}_{\{h,l\}}$ is defined using $\mathbf{L}_{\mathcal{G}_{\{h,l\}}}$, and block diagonal $\widetilde{\mathbf{P}}^h$, where each block $\widetilde{\mathbf{P}}_{\{h,l\}}$ is defined using $\mathbf{L}_{\mathcal{H}_{\{h,l\}}}$ and $\mathcal{H}_{\{h,l\}}$.

The statement. Since $\mathbf{P}^h - \widetilde{\mathbf{P}}^h$ is block diagonal, we have that a bound on its largest eigenvalue implies an equal bound on each matrix on the diagonal, i.e.,

$$\|\mathbf{P}^h - \widetilde{\mathbf{P}}^h\| = \max_l \|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\| \leq \varepsilon \Rightarrow \|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\| \leq \varepsilon$$

for all blocks l on the diagonal, and since each block corresponds to a sparsifier $\mathcal{H}_{\{h,l\}}$, this means that if $\|\mathbf{P}^h - \widetilde{\mathbf{P}}^h\| \leq \varepsilon$, all sparsifiers at layer l are (ε, γ)-sparsifier of their respective graphs. Let $d_{\text{eff}}^{\{h,l\}}(\gamma)$ be the effective dimension of $\mathbf{L}_{\mathcal{G}_{\{h,l\}}}$. Our goal is to show

$$\begin{aligned}
&\mathbb{P}\left(\exists h \in \{1, \dots, k\} : \|\mathbf{P}^h - \widetilde{\mathbf{P}}^h\|_2 \geq \varepsilon \,\cup \max_{l=1,\dots,|\mathcal{S}_h|} |\mathcal{H}_{\{h,l\}}| \geq 3\overline{q}d_{\text{eff}}^{\{h,l\}}(\gamma)\right) \\
&= \mathbb{P}\left(\exists h \in \{1, \dots, k\} : \underbrace{\left(\max_{l=1,\dots,|\mathcal{S}_h|} \|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\|_2\right) \geq \varepsilon}_{A_h} \cup \underbrace{\left(\max_{l=1,\dots,|\mathcal{S}_h|} |\mathcal{H}_{\{h,l\}}| \geq 3\overline{q}d_{\text{eff}}^{\{h,l\}}(\gamma)\right)}_{B_h}\right) \leq \delta,
\end{aligned} \tag{10}$$

where event A_h refers to the case when some sparsifier $\mathcal{H}_{\{h,l\}}$ at an intermediate layer h fails to accurately approximate $\mathbf{L}_{\{h,l\}}$ and event B_h considers the case when the memory requirement is not met (i.e., too many edges are kept in one of the sparsifiers $\mathcal{H}_{\{h,l\}}$ at a certain layer h). After reformulating and a union bound we obtain

$$\begin{aligned}
&\mathbb{P}\bigg(\exists h \in \{1,\dots,k\} : \|\mathbf{P}^h - \widetilde{\mathbf{P}}^h\|_2 \geq \varepsilon \;\cup \max_{l=1,\dots,|\mathcal{S}_h|} |\mathcal{H}_{\{h,l\}}| \geq 3\overline{q}d_{\text{eff}}^{\{h,l\}}(\gamma)\bigg) \\
&\leq \sum_{h=1}^{k}\sum_{l=1}^{|\mathcal{S}_h|} \mathbb{P}\left(\|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\|_2 \geq \varepsilon\right) \\
&\quad + \sum_{h=1}^{k}\sum_{l=1}^{|\mathcal{S}_h|} \mathbb{P}\left(|\mathcal{H}_{\{h,l\}}| \geq 3\overline{q}d_{\text{eff}}^{\{h,l\}}(\gamma) \cap \left\{\forall h' \in \{1,\dots,h\} : \|\mathbf{P}^{h'} - \widetilde{\mathbf{P}}^{h'}\|_2 \leq \varepsilon\right\}\right) \leq \delta. \qquad (11)
\end{aligned}$$

The accuracy of the sparsifier (first term in the previous bound) is guaranteed by the fact that given an (ε,γ)-accurate sparsifier we obtain γ-effective resistance estimates (i.e. RLS estimates) which are at least a fraction of the true ones, thus forcing the algorithm to sample each column *enough*. On the other hand, the space complexity bound is achieved by exploiting the fact that estimates are always upper-bounded by the true γ-effective resistance, thus ensuring that Alg. 1 does not oversample columns w.r.t. the sampling process following the exact γ-effective resistance.

In the reminder of the proof, we will show that both events happen with probability smaller than $\delta/(2k^2)$. Since $|\mathcal{S}_h| = k - h + 1$, we have

$$\sum_{h=1}^{k}\sum_{l=1}^{|\mathcal{S}_h|}\frac{\delta}{2k^2} = \sum_{h=1}^{k}(k-h+1)\frac{\delta}{2k^2} = k(k+1)\frac{\delta}{4k^2} \leq k^2\frac{\delta}{2k^2} = \delta/2,$$

and the union bound over all events is smaller than δ. The main advantage of splitting the failure probability as we did in Eq. 11 is that we can now analyze the processes that generated each $\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}$ (and each sparsifier $\mathcal{H}_{\{h,l\}}$) separately. Focusing on a single node $\{h,l\}$ restricts our problem on a well defined graph $\mathcal{G}_{\{h,l\}}$, where we can analyze the evolution of $\mathcal{H}_{\{h,l\}}$ sequentially.

C.1. Bounding the projection error $\|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\|$

The sequential process. Thanks to the union bound in Eq. 11, instead of having to consider the whole merge tree followed by Alg. 1, we can focus on each individual node $\{h,l\}$ and study the sequential process that generated its sparsifier $\mathcal{H}_{\{h,l\}}$. We will now map more clearly the actions taken by Alg. 1 to the process that generated $\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}$. We begin by focusing on $\widetilde{\mathbf{P}}_{\{h,l\}}$, which is a random matrix defined starting from the fixed graph Laplacian $\mathbf{L}_{\mathcal{G}_{\{h,l\}}}$ and the random sparsifier Laplacian $\mathbf{L}_{\mathcal{H}_{\{h,l\}}}$, where the randomness influences both which edges are included in $\mathcal{H}_{\{h,l\}}$, and the weight with which they are added.

Note that since the merge tree is decided in advance, the graph $\mathcal{G}_{\{h,l\}}$ is not a random object, and is fixed for the whole process. Consider now an edge $e \in \mathcal{G}_{\{h,l\}}$. Again for simplicity and without loss of generality[9], we will assume that the starting graphs in the leaves are edge-disjoint. Therefore, there is a single path in the tree, with length h, from the leaves to $\{h,l\}$. This means that for all $s < h$, we can properly define a unique $\widetilde{p}_{s,e}$ and $q_{s,e}$ associated with that point. More in detail, if at layer s point i is present in $\mathcal{G}_{\{s,l'\}}$, it means that either (1) Alg. 1 used $\mathcal{H}_{\{s,l'\}}$ to compute $\widetilde{p}_{s,e}$, and $\widetilde{p}_{s,e}$ to compute $q_{s,e}$, or (2) at layer h Alg. 1 did not have any merge scheduled for point i, and we simply propagate $\widetilde{p}_{s,e} = \widetilde{p}_{s-1,i}$ and $q_{s,e} = q_{s-1,i}$. Consistently with the algorithm, we initialize $\widetilde{p}_{0,i} = 1$ and $q_{0,i} = \overline{q}$.

Denote $m_{\{h,l\}} = |\mathcal{G}_{\{h,l\}}|$ so that we can use index $i \in [m_{\{h,l\}}]$ to index all edges in $\mathcal{G}_{\{h,l\}}$. Given the $n \times m_{\{h,l\}}$ matrix $\mathbf{Q} = (\mathbf{L}_{\mathcal{G}_{\{h,l\}}} + \gamma\mathbf{I})^{-1/2}\mathbf{B}_{\mathcal{G}_{\{h,l\}}}$ with its e-th column $\mathbf{q}_i = (\mathbf{L}_{\mathcal{G}_{\{h,l\}}} + \gamma\mathbf{I})^{-1/2}\mathbf{B}_{\mathcal{G}_{\{h,l\}}}\mathbf{e}_{m_{\{h,l\}},e}$, we can rewrite the projection matrix as that $\mathbf{P}_{\{h,l\}} = \mathbf{Q}\mathbf{Q}^\mathsf{T} = \sum_{e=1}^{m_{\{h,l\}}} \mathbf{q}_e\mathbf{q}_e^\mathsf{T}$. Note that

$$\|\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\| = \mathbf{q}_e^\mathsf{T}\mathbf{q}_e = \mathbf{e}_{m_{\{h,l\}},e}^\mathsf{T}\mathbf{Q}^\mathsf{T}\mathbf{Q}\mathbf{e}_{m_{\{h,l\}},e} = \mathbf{e}_{m_{\{h,l\}},e}^\mathsf{T}\mathbf{Q}\mathbf{Q}^\mathsf{T}\mathbf{e}_{m_{\{h,l\}},e} = \mathbf{e}_{m_{\{h,l\}},e}^\mathsf{T}\mathbf{P}_{\{h,l\}}\mathbf{e}_{m_{\{h,l\}},e} = r_e(\gamma),$$

or, in other words, the norm $\|\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\|$ is equal to the γ-effective resistance of the e-th edge w.r.t. to graph $\mathcal{G}_{\{h,l\}}$. Note that since e is present only in node l on layer h, its γ-effective resistance is uniquely defined w.r.t. $\mathcal{G}_{\{h,l\}}$ and can be shortened as

[9] Alternatively, we can assign an index to each of the edges in the leaf graphs, requiring at most $km \leq kn^2$ indices.

$r_{h,e}$. Using $\mathbf{q}_e$, we can also introduce the random matrix $\widetilde{\mathbf{P}}_s^{\{h,l\}}$ as

$$\widetilde{\mathbf{P}}_s^{\{h,l\}} = \sum_{e=1}^{m_{\{h,l\}}} \frac{q_{s,e}}{\overline{q}\widetilde{p}_{s,e}} \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} = \sum_{e=1}^{m_{\{h,l\}}} \sum_{j=1}^{\overline{q}} \frac{z_{s,e,j}}{\overline{q}\widetilde{p}_{s,e}} \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}}.$$

where $z_{s,e,j}$ are $\{0,1\}$ r.v. such that $q_{s,e} = \sum_{j=1}^{\overline{q}} z_{s,e,j}$, or in other words $z_{s,e,j}$ are the Bernoulli random variables that compose the Binomial $q_{s,e}$ associated with edge e, with j indexing each individual copy of the edge. Note that when $s = h$, we have that $\widetilde{\mathbf{P}}_h^{\{h,l\}} = \widetilde{\mathbf{P}}_{\{h,l\}}$ and we recover the definition of the approximate projection matrix from Alg. 1. But, for a general $s \neq h$ $\widetilde{\mathbf{P}}_s^{\{h,l\}}$ does not have a direct interpretation in the context of Alg. 1. It combines the vectors $\mathbf{q}_e$, which are defined using $\mathbf{L}_{\mathcal{G}_{\{h,l\}}}$ at layer h, with the weights $\widetilde{p}_{s,e}$ computed by Alg. 1 across multiple nodes at layer s, which are potentially stored in different machines that cannot communicate. Nonetheless, $\widetilde{\mathbf{P}}_s^{\{h,l\}}$ is a useful tool to analyze Alg. 1.

Taking into account that we are now considering a specific node $\{h,l\}$, we can drop the index from the graphs $\mathcal{G}_{\{h,l\}} = \mathcal{G}$, γ-effective resistances $\tau_{h,e}$, and size $m_{\{h,l\}} = m$. Using this shorter notation, we can reformulate our objective as bounding $\|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\|_2 = \|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_h^{\{h,l\}}\|_2$, and reformulate the process as a sequence of matrices $\{\mathbf{Y}_s\}_{s=1}^h$ defined as

$$\mathbf{Y}_s = \mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_s^{\{h,l\}} = \frac{1}{\overline{q}} \sum_{e=1}^{m} \sum_{j=1}^{\overline{q}} \left(1 - \frac{z_{s,e,j}}{\widetilde{p}_{s,e}}\right) \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}},$$

where $\mathbf{Y}_h = \mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_h^{\{h,l\}} = \mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}$, and $\mathbf{Y}_1 = \mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_0^{\{h,l\}} = \mathbf{0}$ since $\widetilde{p}_{0,i} = 1$ and $q_{0,i} = \overline{q}$.

C.2. Bounding $\mathbf{Y}_h$

We transformed the problem of bounding $\|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\|$ into the problem of bounding $\mathbf{Y}_h$, which we modeled as a random matrix process, connected to Alg. 1 by the fact that both algorithm and random process $\mathbf{Y}_h$ make use of the same weight $\widetilde{p}_{s,e}$ and multiplicities $q_{s,e}$.

The frozen process. Inspired by Cohen et al. (2016), we will now replace the process $\mathbf{Y}_s$ with an alternative process $\overline{\mathbf{Y}}_s$ defined as

$$\overline{\mathbf{Y}}_s = \mathbf{Y}_{s-1}\mathbb{I}\left\{\|\overline{\mathbf{Y}}_{s-1}\| \le \varepsilon\right\} + \overline{\mathbf{Y}}_{s-1}\mathbb{I}\left\{\|\overline{\mathbf{Y}}_{s-1}\| \ge \varepsilon\right\}.$$

This process starts from $\overline{\mathbf{Y}}_0 = \mathbf{Y}_0 = \mathbf{0}$, and is identical to $\mathbf{Y}_s$ until a step $\overline{s}$ where for the first time $\|\mathbf{Y}_{\overline{s}}\| \le \varepsilon$ and $\|\mathbf{Y}_{\overline{s}+1}\| \ge \varepsilon$. After this failure happen the process $\overline{\mathbf{Y}}_s$ is "frozen" at $\overline{s}$ and $\overline{\mathbf{Y}}_s = \mathbf{Y}_{\overline{s}+1}$ for all $\overline{s}+1 \le s \le h$. Consequently, if any of the intermediate elements of the sequence violates the condition $\|\mathbf{Y}_s\| \le \varepsilon$, the last element will violate it too. For the rest, $\overline{\mathbf{Y}}_s$ behaves exactly like $\mathbf{Y}_s$. Therefore,

$$\mathbb{P}\left(\|\mathbf{Y}_h\| \ge \varepsilon\right) \le \mathbb{P}\Big(\|\overline{\mathbf{Y}}_h\| \ge \varepsilon\Big),$$

and if we can bound $\mathbb{P}\Big(\|\overline{\mathbf{Y}}_h\| \ge \varepsilon\Big)$ we will have a bound for the failure probability of Alg. 1, even though after "freezing" the process $\overline{\mathbf{Y}}_h$ does not make the same choices as the algorithm.

We will see now how to construct the process $\overline{\mathbf{Y}}_s$ starting from $z_{s,e,j}$ and $\widetilde{p}_{s,e,j}$. We recursively define the indicator ($\{0,1\}$) random variable $\overline{z}_{s,e,j}$ as

$$\overline{z}_{s,e,j} = \mathbb{I}\left\{u_{s,e,j} \le \frac{\overline{p}_{s,e,j}}{\overline{p}_{s-1,e,j}}\right\}\overline{z}_{s-1,e,j},$$

where $u_{s,e,j} \sim \mathcal{U}(0,1)$ is a $[0,1]$ uniform random variable and $\overline{p}_{s,e,j}$ is defined as

$$\overline{p}_{s,e,j} = \widetilde{p}_{s,e}\mathbb{I}\left\{\|\overline{\mathbf{Y}}_{s-1}\| \le \varepsilon \cap z_{s-1,e,j} = 1\right\} + \overline{p}_{s-1,e,j}\mathbb{I}\left\{\|\overline{\mathbf{Y}}_{s-1}\| \ge \varepsilon \cup z_{s-1,e,j} = 0\right\}.$$

This definition of the process satisfies the freezing condition, since if $\|\mathbf{Y}_{\overline{s}+1}\| \ge \varepsilon$ (we have a failure at step $\overline{s}$), for all $s' \ge \overline{s}+1$ we have $\overline{z}_{s',i,j} = \overline{z}_{\overline{s}+1,i,j}$ with probability 1 ($\overline{p}_{\overline{s}+1,i,j}/\overline{p}_{\overline{s},i,j} = \overline{p}_{\overline{s},i,j}/\overline{p}_{\overline{s},i,j} = 1$), and the weights $1/(\overline{q}\overline{p}_{\overline{s}+1,i,j}) = 1/(\overline{q}\overline{p}_{\overline{s},i,j})$ never change.

Introducing a per-copy weight $\overline{p}_{s,e,j}$ and enforcing that $\overline{p}_{s+1,i,j} = \overline{p}_{s,e,j}$ when $z_{s,e,j} = 0$ avoids subtle inconsistencies in the formulation. In particular, not doing so would semantically correspond to reweighting dropped copies. Although this does not directly affect $\mathbf{Y}_s$ (since the ratio $z_{s,e,j}/\widetilde{p}_{s,e}$ is zero for dropped copies), and therefore the relationship $\mathbb{P}\left(\|\mathbf{Y}_h\| \geq \varepsilon\right) \leq \mathbb{P}\Big(\|\overline{\mathbf{Y}}_h\| \geq \varepsilon\Big)$ still holds, we will see later how maintaining consistency helps us bound the second moment of our process.

We can now arrange the indices s, e, and j into a linear index $t = s$ in the range $[1, \ldots, m^2\overline{q}]$, obtained as $t = \{s,e,j\} = (s-1)m\overline{q} + (e-1)\overline{q} + j$. We also define the difference matrix as

$$\overline{\mathbf{X}}_{\{s,e,j\}} = \frac{1}{q}\left(\frac{z_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{z_{s,e,j}}{\overline{p}_{s,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}},$$

which allows us to write the cumulative matrix as $\overline{\mathbf{Y}}_{\{s,e,j\}} = \sum_{r=1}^{\{s,e,j\}} \overline{\mathbf{X}}_{\{s,e,j\}}$ where the checkedges $\{s, m, \overline{q}\}$ correspond to $\overline{\mathbf{Y}}_s$,

$$\overline{\mathbf{Y}}_{\{s,m,\overline{q}\}} = \overline{\mathbf{Y}}_s = \frac{1}{q}\sum_{e=1}^{m}\sum_{j=1}^{\overline{q}}\left(1 - \frac{z_{s,e,j}}{\overline{p}_{s,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}.$$

Let $\mathcal{F}_s$ be the filtration containing all the realizations of the uniform random variables $u_{s,e,j}$ up to the step s, that is $\mathcal{F}_s = \{u_{s',e',j'}, \forall\{s',e',j'\} \leq s\}$. Again, we notice that $\mathcal{F}_s$ defines the state of the algorithm after completing iteration s because, unless a "freezing" happened, Alg. 1 and $\overline{\mathbf{Y}}_s$ flip coins with the same probability, and generate the same sparsifiers. Since $\widetilde{r}_{s,e}$ and $\overline{p}_{s,e,j}$ are computed at the beginning of iteration s using the sparsifier $\mathcal{H}_{\{s,l'\}}$ (for some l' unique at layer s), they are fully determined by $\mathcal{F}_{s-1}$. Furthermore, since $\mathcal{F}_{s-1}$ also defines the values of all indicator variables $\overline{z}_{s',e,j}$ up to $\overline{z}_{s-1,e,j}$ for any i and j, we have that all the Bernoulli variables $\overline{z}_{s,e,j}$ at iteration s are conditionally independent given $\mathcal{F}_{s-1}$. In other words, we have that for any e', and j' such that $\{s,1,1\} \leq \{s,e',j'\} < s$ the following random variables are equal in distribution,

$$\overline{z}_{s,e,j}\big|\mathcal{F}_{\{s,e',j'\}} = \overline{z}_{s,e,j}\big|\mathcal{F}_{\{s-1,m,\overline{q}\}} \sim \mathcal{B}\Big(\frac{\overline{p}_{s,e,j}}{\overline{p}_{s-1,e,j}}\Big), \tag{12}$$

and for any e', and j' such that $\{s,1,1\} \leq \{s,e',j'\} \leq \{s,m,\overline{q}\}$ and $s \neq \{s,e',j'\}$ we have the independence

$$\overline{z}_{s,e,j}\big|\mathcal{F}_{\{s-1,m,\overline{q}\}} \perp \overline{z}_{s,e',j'}\big|\mathcal{F}_{\{s-1,m,\overline{q}\}}. \tag{13}$$

While knowing that $\|\mathbf{Y}_s\| \leq \varepsilon$ is not sufficient to provide guarantees for the approximate probabilities $\widetilde{p}_{s,e}$, we can show that it is enough to prove that the frozen probabilities $\overline{p}_{s,e,j}$ are never too small.

Lemma 1. *Let $\alpha = (1+3\varepsilon)/(1-\varepsilon)$ and $\overline{p}_{s,e,j}$ be the sequence of probabilities generated by the freezing process. Then for any s, e, and j, we have $\overline{p}_{s,e,j} \geq p_{h,e}/\alpha = r_{h,e}/\alpha$.*

Proof of Lemma 1. Let $\overline{s}$ be the step where the process freezes ($\overline{s} = h$ if it does not freeze), or, in other words, $\|\mathbf{Y}_{\overline{s}}\| < \varepsilon$ and $\|\mathbf{Y}_{\overline{s}+1}\| \geq \varepsilon$. From the definition of $\overline{p}_{s,e,j}$, we have that

$$\begin{aligned}\overline{p}_{s,e,j} &\geq \overline{p}_{\overline{s},i} = \widetilde{p}_{\overline{s},e} = \max\left\{\min\left\{\widetilde{r}_{\overline{s},e},\ \widetilde{p}_{\overline{s}-1,e}\right\},\ \widetilde{p}_{\overline{s}-1,e}/2\right\}\\ &\geq \min\left\{\widetilde{r}_{\overline{s},e},\ \widetilde{p}_{\overline{s}-1,e}\right\} = \min\left\{\widetilde{r}_{\overline{s},e},\ \widetilde{p}_{\overline{s}-2,e}\right\} = \min\left\{\widetilde{r}_{\overline{s},e},\ \widetilde{p}_{\overline{s}-3,e}\right\}\ldots = \min\left\{\widetilde{r}_{\overline{s},e},\ \widetilde{p}_{0,e}\right\} = \widetilde{r}_{\overline{s},e},\end{aligned}$$

and therefore $\overline{p}_{s,e,j} \geq \widetilde{r}_{\overline{s},e}$. Now let $\{\overline{s}, l'\}$ be the node where $\widetilde{r}_{\overline{s},e}$ was computed. We will again drop the $\{h,l\}$ index from $\mathcal{G}_{\{h,l\}}$, and simply refer to it as $\mathcal{G}$. Similarly, we will refer with $\mathcal{G}_e$ to $\mathcal{G}_{\{\overline{s},l'\}}$ (as in, the dataset used to compute $\widetilde{r}_{\overline{s},e}$ for edge e), and with $\overline{\mathcal{G}}_e$ to the edges in $\mathcal{G}$ not contained in $\mathcal{G}_e$ (complement of $\mathcal{G}_e$). Define $\mathbf{A}$ as the $|\mathcal{G}| \times |\mathcal{G}_e|$ matrix that contains the columns of $\mathbf{S}_{\overline{s}}$ related to edges in $\mathcal{G}_e$, and similarly define $\mathbf{B}$ as the $|\mathcal{G}| \times |\overline{\mathcal{G}}_e|$ matrix that contains the columns of $\mathbf{S}_{\overline{s}}$ related to edges in $\overline{\mathcal{G}}_e$, where $\mathbf{S}_{\overline{s}}$ can be reconstructed by interleaving columns of $\mathbf{A}$ and $\mathbf{B}$. From its definition in Eq.**??**, we know that $\widetilde{r}_{s,e}$ is computed by Alg. 1 as

$$\widetilde{r}_{s,e} = (1-\varepsilon)\phi_e^{\mathsf{T}}\left(\mathbf{B}_{\mathcal{G}}\mathbf{A}\mathbf{A}^{\mathsf{T}}\mathbf{B}_{\mathcal{G}}^{\mathsf{T}} + (1+\varepsilon)\gamma\mathbf{I}_D\right)^{-1}\phi_i,$$

using only the edges in $\mathbf{A}$ that are available at node $\{\overline{s}, l'\}$. From Lem.**??** we know that

$$\begin{aligned}\|\mathbf{Y}_{\overline{s}}\| = \left\|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\overline{s}}^{\{h,l\}}\right\|_2 &= \left\|(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D)^{-1/2}(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} - \mathbf{B}_\mathcal{G}\mathbf{S}_{\overline{s}}\mathbf{S}_{\overline{s}}^\mathsf{T}\mathbf{B}_\mathcal{G}^\mathsf{T})(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D)^{-1/2}\right\|_2 \\ &= \left\|(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D)^{-1/2}(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} - \mathbf{B}_\mathcal{G}\mathbf{A}\mathbf{A}^\mathsf{T}\mathbf{B}_\mathcal{G}^\mathsf{T} - \mathbf{B}_\mathcal{G}\mathbf{B}\mathbf{B}^\mathsf{T}\mathbf{B}_\mathcal{G}^\mathsf{T})(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D)^{-1/2}\right\|_2 \leq \varepsilon\end{aligned}$$

and we know that this implies

$$\mathbf{B}_\mathcal{G}\mathbf{A}\mathbf{A}^\mathsf{T}\mathbf{B}_\mathcal{G}^\mathsf{T} \preceq \mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \varepsilon(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D) - \mathbf{B}_\mathcal{G}\mathbf{B}\mathbf{B}^\mathsf{T}\mathbf{B}_\mathcal{G}^\mathsf{T} \preceq \mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \varepsilon(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D).$$

Plugging it in the initial definition,

$$\begin{aligned}\widetilde{r}_{s,e} &= (1-\varepsilon)\phi_e^\mathsf{T}\left(\mathbf{B}_\mathcal{G}\mathbf{A}\mathbf{A}^\mathsf{T}\mathbf{B}_\mathcal{G}^\mathsf{T} + (1+\varepsilon)\gamma\mathbf{I}_D\right)^{-1}\phi_i \\ &\geq (1-\varepsilon)\phi_e^\mathsf{T}(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \varepsilon(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D) + (1+\varepsilon)\gamma\mathbf{I}_D)^{-1}\phi_e \\ &= (1-\varepsilon)\frac{1}{1+2\varepsilon}\phi_e^\mathsf{T}(\mathbf{B}_\mathcal{G}\mathbf{B}_\mathcal{G}^\mathsf{T} + \gamma\mathbf{I}_D)^{-1}\phi_e \geq \frac{1-\varepsilon}{1+2\varepsilon}\tau_{h,i} \geq \tau_{h,i}/\alpha.\end{aligned}$$

□

We now proceed by studying the process $\{\overline{\mathbf{Y}}_s\}_{s=1}^h$ and showing that it is a bounded martingale. In order to show that $\overline{\mathbf{Y}}_s$ is a martingale, it is sufficient to verify the following (equivalent) conditions

$$\mathbb{E}\left[\overline{\mathbf{Y}}_s \;\middle|\; \mathcal{F}_{s-1}\right] = \overline{\mathbf{Y}}_{s-1} \;\;\Leftrightarrow\;\; \mathbb{E}\left[\overline{\mathbf{X}}_{\{s,e,j\}} \;\middle|\; \mathcal{F}_{s-1}\right] = \mathbf{0}.$$

We begin by inspecting the conditional random variable $\overline{\mathbf{X}}_{\{s,e,j\}}|\mathcal{F}_{s-1}$. Given the definition of $\overline{\mathbf{X}}_{\{s,e,j\}}$, the conditioning on $\mathcal{F}_{s-1}$ determines the values of $\overline{z}_{s-1,e,j}$ and the approximate probabilities $\overline{p}_{s-1,e,j}$ and $\overline{p}_{s,e,j}$. In fact, remember that these quantities are fully determined by the realizations in $\mathcal{F}_{s-1}$ which are contained in $\mathcal{F}_{s-1}$. As a result, the only stochastic quantity in $\overline{\mathbf{X}}_{\{s,e,j\}}$ is the variable $\overline{z}_{s,e,j}$. Specifically, if $\|\overline{\mathbf{Y}}_{s-1}\| \geq \varepsilon$, then we have $\overline{p}_{s,e,j} = \overline{p}_{s-1,e,j}$ and $\overline{z}_{s,e,j} = \overline{z}_{s-1,e,j}$ (the process is stopped), and the martingale requirement $\mathbb{E}\left[\overline{\mathbf{X}}_{\{s,e,j\}} \;\middle|\; \mathcal{F}_{s-1}\right] = \mathbf{0}$ is trivially satisfied. On the other hand, if $\|\overline{\mathbf{Y}}_{s-1}\| \leq \varepsilon$ we have

$$\begin{aligned}&\mathop{\mathbb{E}}_{u_{s,e,j}}\left[\frac{1}{q}\left(\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T} \;\middle|\; \mathcal{F}_{s-1}\right] \\ &\quad= \frac{1}{q}\left(\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s,e,j}}\mathbb{E}\left[\mathbb{I}\left\{u_{s,e,j} \leq \frac{\overline{p}_{s,e,j}}{\overline{p}_{s-1,e,j}}\right\} \;\middle|\; \mathcal{F}_{s-1}\right]\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T} \\ &\quad= \frac{1}{q}\left(\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s,e,j}}\frac{\overline{p}_{s,e,j}}{\overline{p}_{s-1,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T} = \mathbf{0},\end{aligned}$$

where we use the recursive definition of $\overline{z}_{s,e,j}$ and the fact that $u_{s,e,j}$ is a uniform random variable in $[0,1]$. This proves that $\overline{\mathbf{Y}}_s$ is indeed a martingale. We now compute an upper-bound R on the norm of the values of the difference process as

$$\|\overline{\mathbf{X}}_{\{s,e,j\}}\| = \frac{1}{q}\left|\left(\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}\right)\right|\|\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\| \leq \frac{1}{q}\frac{1}{\overline{p}_{s,e,j}}\|\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\| = \frac{1}{q}\frac{1}{\overline{p}_{s,e,j}}\tau_{h,i} \leq \frac{1}{q}\frac{\alpha}{\tau_{h,i}}\tau_{h,i} = \frac{\alpha}{q} \stackrel{\text{def}}{=} R,$$

where we used Lemma 1 to bound $\overline{p}_{s,e,j} \leq r_{h,e}/\alpha$. If instead, $\|\overline{\mathbf{Y}}_{s-1}\| \geq \varepsilon$, the process is stopped and $\|\overline{\mathbf{X}}_s\| = \|\mathbf{0}\| = 0 \leq R$.

We are now ready to use a Freedman matrix inequality from (Tropp, 2011) to bound the norm of $\overline{\mathbf{Y}}$.

Proposition 4 (Tropp (2011), Theorem 1.2)**.** *Consider a matrix martingale $\{\mathbf{Y}_k : k = 0, 1, 2, \dots\}$ whose values are self-adjoint matrices with dimension d, and let $\{\mathbf{X}_k : k = 1, 2, 3, \dots\}$ be the difference sequence. Assume that the difference sequence is uniformly bounded in the sense that*

$$\|\mathbf{X}_k\|_2 \leq R \quad \textit{almost surely} \quad \textit{for } k = 1, 2, 3, \dots.$$

Define the predictable quadratic variation process of the martingale as

$$\mathbf{W}_k \overset{\text{def}}{=} \sum_{j=1}^{k} \mathbb{E}\left[\mathbf{X}_j^2 \;\middle|\; \{\mathbf{X}_s\}_{s=0}^{j-1}\right], \quad \textit{for } k = 1, 2, 3, \ldots.$$

Then, for all $\varepsilon \geq 0$ and $\sigma^2 > 0$,

$$\mathbb{P}\left(\exists k \geq 0 : \|\mathbf{Y}_k\|_2 \geq \varepsilon \;\cap\; \|\mathbf{W}_k\| \leq \sigma^2\right) \leq 2d \cdot \exp\left\{-\frac{\varepsilon^2/2}{\sigma^2 + R\varepsilon/3}\right\}.$$

In order to use the previous inequality, we develop the probability of error for any fixed h as

$$\begin{aligned}\mathbb{P}\left(\|\mathbf{Y}_h\| \geq \varepsilon\right) \leq \mathbb{P}\left(\|\overline{\mathbf{Y}}_h\| \geq \varepsilon\right) &= \mathbb{P}\left(\|\overline{\mathbf{Y}}_h\| \geq \varepsilon \cap \|\mathbf{W}_h\| \leq \sigma^2\right) + \mathbb{P}\left(\|\overline{\mathbf{Y}}_h\| \geq \varepsilon \cap \|\mathbf{W}_h\| \geq \sigma^2\right)\\ &\leq \underbrace{\mathbb{P}\left(\|\overline{\mathbf{Y}}_h\| \geq \varepsilon \cap \|\mathbf{W}_h\| \leq \sigma^2\right)}_{\text{(a)}} + \underbrace{\mathbb{P}\left(\|\mathbf{W}_h\| \geq \sigma^2\right)}_{\text{(b)}}.\end{aligned}$$

Using the bound on $\|\overline{\mathbf{X}}_{\{s,e,j\}}\|_2$, we can directly apply Proposition 4 to bound (a) for any fixed σ^2. To bound the part (b), we use the following lemma, proved later in Sec. C.3.

Lemma 2 (Low probability of the large norm of the predictable quadratic variation process)**.**

$$\mathbb{P}\left(\|\mathbf{W}_h\| \geq \frac{6\alpha}{\overline{q}}\right) \leq n \cdot \exp\left\{-2\frac{\overline{q}}{\alpha}\right\}$$

Since $\mathbf{P}_{\{h,l\}}$ is defined at most on n nodes, combining Prop. 4 with $\sigma^2 = 6\alpha/\overline{q}$, Lem 2, the fact that $2\varepsilon/3 \leq 1$ and the value used by Alg. 1 $\overline{q} = 39\alpha\log(2n/\delta)/\varepsilon^2$ we obtain

$$\begin{aligned}\mathbb{P}\left(\|\mathbf{P}_{\{h,l\}} - \widetilde{\mathbf{P}}_{\{h,l\}}\|_2 \geq \varepsilon\right) &= \mathbb{P}\left(\|\mathbf{Y}_h\| \geq \varepsilon\right) \leq \mathbb{P}\left(\|\overline{\mathbf{Y}}_h\| \geq \varepsilon \cap \|\mathbf{W}_h\| \leq \sigma^2\right) + \mathbb{P}\left(\|\mathbf{W}_h\| \geq \sigma^2\right)\\ &\leq 2n \cdot \exp\left\{-\frac{\varepsilon^2\overline{q}}{\alpha}\left(\frac{1}{12 + 2\varepsilon/3}\right)\right\} + n \cdot \exp\left\{-2\frac{\overline{q}}{\alpha}\right\}\\ &\leq 3n \cdot \exp\left\{-\frac{\varepsilon^2}{13\alpha}\overline{q}\right\} = 3n \cdot \exp\left\{-3\log\left(\frac{2n}{\delta}\right)\right\}\\ &= 3n \cdot \exp\left\{-\log\left(\left(\frac{2n}{\delta}\right)^3\right)\right\} = 3n\frac{\delta^3}{8n^3} \leq \frac{\delta}{2n^2}.\end{aligned}$$

This, combined with the fact that $k \leq n^2/n \leq n$ since at most we can split our graph in n parts each containing n edges, concludes this part of the proof.

C.3. Proof of Lemma 2 (bound on predictable quadratic variation)

Step 1 (a preliminary bound). We start by writing out $\mathbf{W}_t$ for the process $\overline{\mathbf{Y}}_s$,

$$\mathbf{W}_t = \frac{1}{\overline{q}^2}\sum_{\{s,e,j\} \leq t} \mathbb{E}\left[\left(\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}\right)^2 \;\middle|\; \mathcal{F}_{\{s,e,j\}-1}\right]\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T}.$$

We rewrite the expectation terms in the equation above as

$$\begin{aligned}
&\mathbb{E}\left[\left(\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}\right)^2 \;\middle|\; \mathcal{F}_{\{s,e,j\}-1}\right]\\
&= \mathbb{E}\left[\frac{\overline{z}^2_{s-1,e,j}}{\overline{p}^2_{s-1,e,j}} - 2\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}} + \frac{\overline{z}^2_{s,e,j}}{\overline{p}^2_{s,e,j}} \;\middle|\; \mathcal{F}_{\{s,e,j\}-1}\right]\\
&\overset{(a)}{=} \mathbb{E}\left[\frac{\overline{z}^2_{s-1,e,j}}{\overline{p}^2_{s-1,e,j}} - 2\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}} + \frac{\overline{z}^2_{s,e,j}}{\overline{p}^2_{s,e,j}} \;\middle|\; \mathcal{F}_{s-1}\right]\\
&= \frac{\overline{z}^2_{s-1,e,j}}{\overline{p}^2_{s-1,e,j}} - 2\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\frac{1}{\overline{p}_{s,e,j}}\mathbb{E}\left[\overline{z}_{s,e,j} \mid \mathcal{F}_{s-1}\right] + \frac{1}{\overline{p}^2_{s,e,j}}\mathbb{E}\left[\overline{z}^2_{s,e,j} \mid \mathcal{F}_{s-1}\right]\\
&\overset{(b)}{=} \frac{\overline{z}_{s-1,e,j}}{\overline{p}^2_{s-1,e,j}} - 2\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} + \frac{1}{\overline{p}^2_{s,e,j}}\mathbb{E}\left[\overline{z}_{s,e,j} \mid \mathcal{F}_{s-1}\right]\\
&= \frac{1}{\overline{p}^2_{s,e,j}}\mathbb{E}\left[\overline{z}_{s,e,j} \mid \mathcal{F}_{s-1}\right] - \frac{\overline{z}_{s-1,e,j}}{\overline{p}^2_{s-1,e,j}}\\
&\overset{(c)}{=} \frac{1}{\overline{p}_{s,e,j}}\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s-1,e,j}}{\overline{p}^2_{s-1,e,j}} = \frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\left(\frac{1}{\overline{p}_{s,e,j}} - \frac{1}{\overline{p}_{s-1,e,j}}\right),
\end{aligned}$$

where in (a) we use the fact that the approximate probabilities $\overline{p}_{s-1,e,j}$ and $\overline{p}_{s,e,j}$ and $\overline{z}_{s-1,e,j}$ are fixed at the end of the previous iteration, while in (b) and (c) we use the fact that $\overline{z}_{s,e,j}$ is a Bernoulli of parameter $\overline{p}_{s,e,j}/\overline{p}_{s-1,e,j}$ (whenever $\overline{z}_{s-1,e,j}$ is equal to 1). Therefore, we can write $\mathbf{W}_t$ at the end of the process as

$$\mathbf{W}_h = \mathbf{W}_{\{h,m,\overline{q}\}} = \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\sum_{s=1}^{h}\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\left(\frac{1}{\overline{p}_{s,e,j}} - \frac{1}{\overline{p}_{s-1,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T}.$$

We can now upper-bound $\mathbf{W}_h$ as

$$\begin{aligned}
\mathbf{W}_h &\preceq \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\sum_{s=1}^{h}\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\left(\frac{1}{\overline{p}_{s,e,j}} - \frac{1}{\overline{p}_{s-1,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\\
&= \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} - \frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} + \sum_{s=1}^{h}\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s-1,e,j}}\left(\frac{1}{\overline{p}_{s,e,j}} - \frac{1}{\overline{p}_{s-1,e,j}}\right)\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\\
&= \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} + \left(\sum_{s=1}^{h} -\frac{\overline{z}_{s,e,j}}{\overline{p}^2_{s,e,j}} + \frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s,e,j}\overline{p}_{s-1,e,j}}\right) - \frac{\overline{z}_{0,e,j}}{\overline{p}^2_{0,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\\
&\preceq \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} + \left(\sum_{s=1}^{h}\frac{\overline{z}_{s-1,e,j}}{\overline{p}_{s,e,j}\overline{p}_{s-1,e,j}} - \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}\overline{p}_{s-1,e,j}}\right)\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\\
&= \frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} + \sum_{s=1}^{h}\frac{\overline{z}_{s-1,e,j}(1-\overline{z}_{s,e,j})}{\overline{p}_{s,e,j}\overline{p}_{s-1,e,j}}\right)\mathbf{q}_e\mathbf{q}_e^\mathsf{T}\mathbf{q}_e\mathbf{q}_e^\mathsf{T},
\end{aligned}$$

where in the inequality we use the fact $\overline{p}_{s,e,j} \leq \overline{p}_{s-1,e,j}$. From the definition of $\overline{p}_{s,e,j}$, we know that when $\overline{z}_{s,e,j} = 0$, $\overline{p}_{s,e,j} = \overline{p}_{s-1,e,j}$. Therefore $\frac{\overline{z}_{s-1,e,j}(1-\overline{z}_{s,e,j})}{\overline{p}_{s,e,j}\overline{p}_{s-1,e,j}} = \frac{\overline{z}_{s-1,e,j}(1-\overline{z}_{s,e,j})}{\overline{p}^2_{s-1,e,j}}$, since the term is non-zero only when $\overline{z}_{s,e,j} = 0$. Finally, we see that only one of the $\overline{z}_{s-1,e,j}(1-\overline{z}_{s,e,j})$ terms can be active for $s \in [h]$ and thus

$$\begin{aligned}
\mathbf{W}_h &\preceq \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{e=1}^{m} \left(\frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} + \sum_{s=1}^{h} \frac{\overline{z}_{s-1,e,j}(1-\overline{z}_{s,e,j})}{\overline{p}^2_{s-1,e,j}} \right) \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} \\
&= \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{e=1}^{m} \left(\max \left\{ \max_{s=1,\dots,h} \left\{ \frac{\overline{z}_{s-1,e,j}(1-\overline{z}_{s,e,j})}{\overline{p}^2_{s-1,e,j}} \right\} ; \frac{\overline{z}_{h,e,j}}{\overline{p}^2_{h,e,j}} \right\} \right) \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} \\
&= \frac{1}{\overline{q}^2} \sum_{j=1}^{\overline{q}} \sum_{e=1}^{m} \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} \mathbf{q}_e \mathbf{q}_e^{\mathsf{T}} \left(\max_{s=0,\dots,h} \left\{ \frac{\overline{z}_{s,e,j}}{\overline{p}^2_{s,e,j}} \right\} \right). \qquad (14)
\end{aligned}$$

Step 2 (introduction of a stochastically dominant process). We want to study $\max_{s=0,\dots,h} \left\{ \frac{\overline{z}_{s,e,j}}{\overline{p}^2_{s,e,j}} \right\}$. To simplify notation, we will consider $\max_{s=0,\dots,h} \left\{ \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}} \right\}$, where we removed the square, which will be re-added in the end. We know trivially that this quantity is larger or equal than, 1 because $\overline{z}_{0,e,j}/\overline{p}_{0,e,j} = 1$, but upper-bounding this quantity is not trivial as the evolution of the various $\overline{p}_{s,e,j}$ depends in a complex way on the interaction between the random variables $\overline{z}_{s,e,j}$. Nonetheless, whenever $\overline{p}_{s,e,j}$ is significantly smaller than $\overline{p}_{s-1,e,j}$, the probability of keeping a copy of edge e at iteration s (i.e., $\overline{z}_{s,e,j} = 1$) is also very small. As a result, we expect the ratio $\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}$ to be still small with high probability.

Unfortunately, due to the dependence between different copies of the edge at different iterations, it seems difficult to exploit this intuition directly to provide an overall high-probability bound on $\mathbf{W}_h$. For this reason, we simplify the analysis by replacing each of the (potentially dependent) chains $\{\overline{z}_{s,e,j}/\overline{p}_{s,e,j}\}_{s=0}^{h}$ with a set of (independent) random variables $w_{0,e,j}$ that will stochastically dominate them.

We define the random variable $w_{s,e,j}$ using the following conditional distribution,[10]

$$\mathbb{P}\left(\frac{1}{w_{s,e,j}} \leq a \;\middle|\; \mathcal{F}_s \right) = \begin{cases} 0 & \text{for} \quad a < 1/\overline{p}_{s,e,j} \\ 1 - \frac{1}{\overline{p}_{s,e,j} a} & \text{for} \quad 1/\overline{p}_{s,e,j} \leq a < \alpha/p_{h,e} \\ 1 & \text{for} \quad \alpha/p_{h,e} \leq a \end{cases}.$$

To show that this distribution is well defined, we use Lem. 1 to guarantee that $1/\overline{p}_{s,e,j} \leq a < \alpha/p_{h,e}$. Note that the distribution of $\frac{1}{w_{s,e,j}}$ conditioned on $\mathcal{F}_s$ is determined by only $\overline{p}_{s,e,j}$, $p_{h,e}$, and α, where $p_{h,e}$ and α are fixed. Remembering that $\overline{p}_{s,e,j}$ is a function of $\mathcal{F}_{s-1}$ (computed using the previous iteration), we have that

$$\mathbb{P}\left(\frac{1}{w_{s,e,j}} \leq a \;\middle|\; \mathcal{F}_s \right) = \mathbb{P}\left(\frac{1}{w_{s,e,j}} \leq a \;\middle|\; \mathcal{F}_{s-1} \right).$$

Notice that in the definition of $w_{s,e,j}$, none of the other $w_{s',e',j'}$ (for any different s', e', or j') appears and $\overline{p}_{s,e,j}$ is a function of $\mathcal{F}_{s-1}$. It follows that given $\mathcal{F}_{s-1}$, $w_{s,e,j}$ is independent from all other $w_{s',e',j'}$ (for any different s', e', or j'). This is easier to see in the probabilistic graphical model reported in Fig. 3, which illustrates the dependence between the various variables.

Finally for the special case $w_{0,e,j}$ the definition above reduces to

$$\mathbb{P}\left(\frac{1}{w_{0,e,j}} \leq a \right) = \begin{cases} 0 & \text{for} \quad a < 1 \\ 1 - \frac{1}{a} & \text{for} \quad 1 \leq a < \alpha/p_{h,e} \\ 1 & \text{for} \quad \alpha/p_{h,e} \leq a \end{cases}, \qquad (15)$$

since $\overline{p}_{0,e,j} = 1$ by definition. From this definition, $w_{0,e,j}$ and $w_{0,e',j'}$ are all independent, and this will allow us to use stronger concentration inequalities for independent random variables.

[10] Notice that unlike $\overline{z}_{s,e,j}$, $w_{s,e,j}$ is no longer $\mathcal{F}_s$-measurable but it is $\mathcal{F}'_s$-measurable, where

$$\mathcal{F}'_{\{s,e,j\}} = \left\{ u_{s',e',j'},\ \forall \{s',e',j'\} \leq \{s,e,j\} \right\} \cup \{w_{s,e,j}\} = \mathcal{F}_{\{s,e,j\}} \cup \{w_{s,e,j}\}.$$

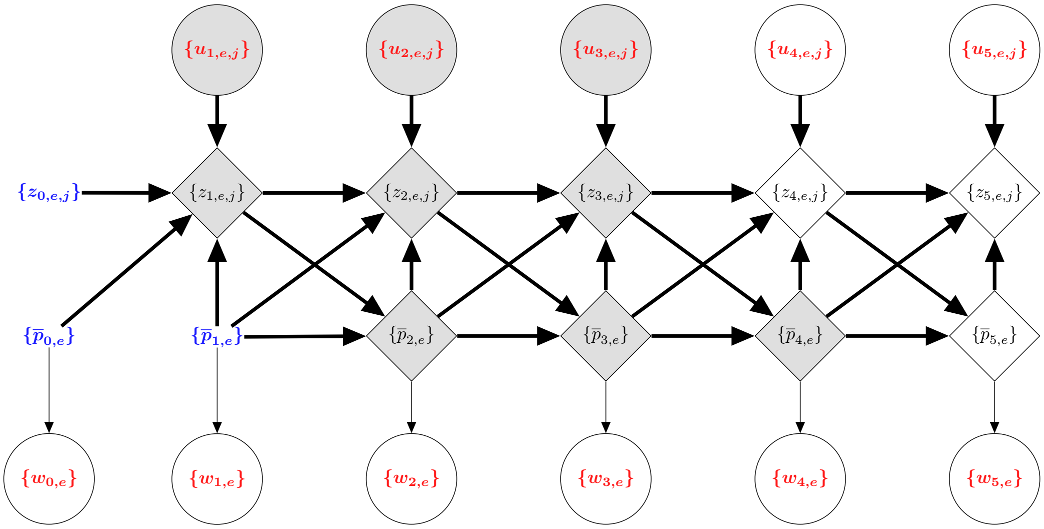


Figure 3. The dependence graph of the considered variables. **Red** variables are **random**. Black variables are deterministically computed using their input (a function of their input), with bold lines indicating the deterministic (functional) relation. **Blue** variables are **constants**. A grey filling indicates that a random variable is observed or a function of observed variables.

Step 3 Proving the dominance. We remind the reader that a random variable A stochastically dominates random variable B, if for all values a the two equivalent conditions are verified,

$$\mathbb{P}(A \geq a) \geq \mathbb{P}(B \geq a) \Leftrightarrow \mathbb{P}(A \leq a) \leq \mathbb{P}(B \leq a).$$

As a consequence, if A dominates B, the following implication holds,

$$\mathbb{P}(A \geq a) \geq \mathbb{P}(B \geq a) \implies \mathbb{E}[A] \geq \mathbb{E}[B],$$

while the reverse (A dominates B, if $\mathbb{E}[A] \geq \mathbb{E}[B]$) is not true in general. Following this definition of stochastic dominance, our goal is to prove

$$\mathbb{P}\left(\max_{s=0}^{h} \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}} \leq a\right) \geq \mathbb{P}\left(\frac{1}{w_{0,e,j}} \leq a\right).$$

We prove this inequality by proceeding backwards with a sequence of conditional probabilities. We first study the distribution of the maximum conditional to the state of the algorithm at the end of iteration h, i.e., $\mathcal{F}_h$. From the definition of $w_{h,e,j}$, we know that, w.p. 1, $1/\overline{p}_{h,e} \leq 1/w_{h,e,j}$. Therefore,

$$\mathbb{P}\left(\max_{s=0,\dots,h} \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}} \leq a\right) \geq \mathbb{P}\left(\max\left\{\max_{s=0,\dots,h-1} \frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}; \frac{\overline{z}_{h,e,j}}{w_{h,e,j}}\right\} \leq a\right).$$

Now focus on an arbitrary intermediate step $1 \leq k \leq h$, where we fix $\mathcal{F}_{k-1}$. Since $u_{k,e,j}$ and $w_{k,e,j}$ are independent given

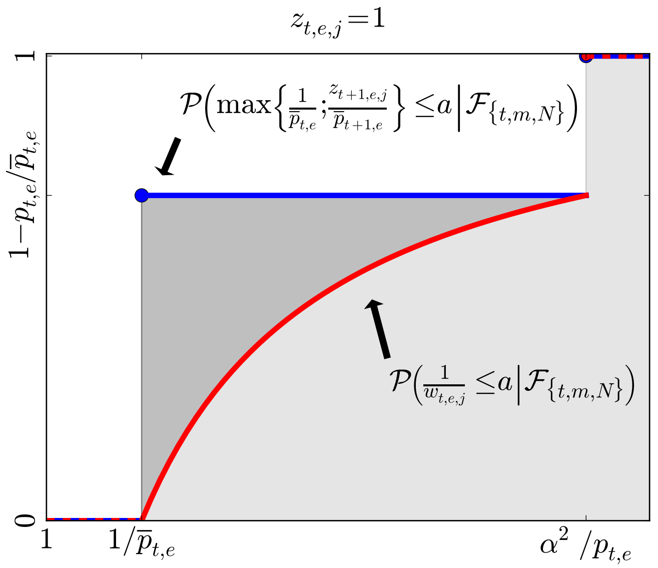


Figure 4. C.d.f. of $\max\left\{\overline{z}_{k-1,e,j}/\overline{p}_{t,e,j};\overline{z}_{k,e,j}/\overline{p}_{k,e,j}\right\}$ and $\overline{z}_{k-1,e,j}/w_{k-1,e,j}$ conditioned on $\mathcal{F}_{\{k-1\}}$. For conciseness, we omit the e, j indices.

$\mathcal{F}_{k-1}$, we have

$$\begin{aligned}
\mathbb{P}\left(\frac{\overline{z}_{k,e,j}}{w_{k,e,j}} \leq a \;\middle|\; \mathcal{F}_{k-1}\right) &= \mathbb{P}\left(\mathbb{I}\left\{u_{k,e,j} \leq \frac{\overline{p}_{k,e,j}}{\overline{p}_{k-1,e,j}}\right\}\frac{1}{w_{k,e,j}} \leq a \;\middle|\; \mathcal{F}_{k-1}\right)\\
&= \begin{cases} 0 & \text{for} \quad a \leq 0\\ 1 - \frac{\overline{p}_{k,e,j}}{\overline{p}_{k-1,e,j}} & \text{for} \quad 0 \leq a < 1/\overline{p}_{k,e,j}\\ 1 - \frac{\overline{p}_{k,e,j}}{\overline{p}_{k-1,e,j}} + \frac{\overline{p}_{k,e,j}}{\overline{p}_{k-1,e,j}}\left(1 - \frac{1}{\overline{p}_{k,e,j}a}\right) = 1 - \frac{1}{\overline{p}_{k-1,e,j}a} & \text{for} \quad 1/\overline{p}_{k,e,j} \leq a < \alpha/p_{h,e}\\ 1 & \text{for} \quad \alpha/p_{h,e} \leq a \end{cases}\\
&\geq \begin{cases} 0 & \text{for} \quad a < 1/\overline{p}_{k-1,e,j}\\ 1 - \frac{1}{\overline{p}_{k-1,e,j}a} & \text{for} \quad 1/\overline{p}_{k-1,e,j} \leq a < 1/\overline{p}_{k,e,j}\\ 1 - \frac{1}{\overline{p}_{k-1,e,j}a} & \text{for} \quad 1/\overline{p}_{k,e,j} \leq a < \alpha/p_{h,e}\\ 1 & \text{for} \quad \alpha/p_{h,e} \leq a \end{cases} \qquad (16)\\
&= \mathbb{P}\left(\frac{1}{w_{k-1,e,j}} \leq a \;\middle|\; \mathcal{F}_{k-2}\right) = \mathbb{P}\left(\frac{1}{w_{k-1,e,j}} \leq a \;\middle|\; \mathcal{F}_{k-1}\right),
\end{aligned}$$

where the inequality is also represented in Fig. 4. We now proceed by peeling off layers from the end of the chain one by one, taking advantage of the dominance we just proved. Fig. 4 visualizes one step of the peeling when $\overline{z}_{k-1,e,j} = 1$ (note that the peeling is trivially true when $\overline{z}_{k-1,e,j} = 0$ since the whole chain terminated at step $\overline{z}_{k-1,e,j}$). We show how to move from an iteration $k \leq h$ to $k-1$.

$$\begin{aligned}
&\mathbb{P}\left(\max\left\{\max_{s=0\dots k-1}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{k,e,j}}{w_{k,e,j}}\right\}\leq a\right) = \mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\max\left\{\max_{s=0\dots k-1}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{k,e,j}}{w_{k,e,j}}\right\}\leq a \;\middle|\; \mathcal{F}_{k-1}\right)\right]\\
&\overset{(a)}{\geq} \mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\max\left\{\max_{s=0\dots k-1}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{k-1,e,j}}{w_{k-1,e,j}}\right\}\leq a \;\middle|\; \mathcal{F}_{k-1}\right)\right]\\
&= \mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\max\left\{\max_{s=0\dots k-2}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{k-1,e,j}}{\overline{p}_{k-1,e,j}};\frac{\overline{z}_{k-1,e,j}}{w_{k-1,e,j}}\right\}\leq a \;\middle|\; \mathcal{F}_{k-1}\right)\right]\\
&= \mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\max\left\{\max_{s=0\dots k-2}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\overline{z}_{k-1,e,j}\max\left\{\frac{1}{\overline{p}_{k-1,e,j}};\frac{1}{w_{k-1,e,j}}\right\}\right\}\leq a \;\middle|\; \mathcal{F}_{k-1}\right)\right]\\
&\overset{(b)}{=} \mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\max\left\{\max_{s=0\dots k-2}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{k-1,e,j}}{w_{k-1,e,j}}\right\}\leq a \;\middle|\; \mathcal{F}_{k-1}\right)\right] = \mathbb{P}\left(\max\left\{\max_{s=0\dots k-2}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{k-1,e,j}}{w_{k-1,e,j}}\right\}\leq a\right),
\end{aligned}$$

where in (a), given $\mathcal{F}_{k-1}$, everything is fixed except $u_{k,e,j}$ and $w_{k,e,j}$ and we can use the stochastic dominance in (16), and in (b) we use the fact that the inner maximum is always attained by $1/w_{k,e,j}$ since by definition $1/w_{k-1,e,j}$ is lower-bounded by $1/\overline{p}_{k-1,e,j}$. Applying the inequality recursively from $k=h$ to $k=1$ removes all $\overline{z}_{s,e,j}$ from the maximum and we are finally left with only $w_{0,e,j}$ as we wanted,

$$\mathbb{P}\left(\max_{s=0,\dots,h}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}\leq a\right)\geq \mathbb{P}\left(\max\left\{\frac{\overline{z}_{0,e,j}}{\overline{p}_{0,e,j}};\frac{\overline{z}_{0,e,j}}{w_{0,e,j}}\right\}\leq a\right)\geq \mathbb{P}\left(\frac{1}{w_{0,e,j}}\leq a\right),$$

where in the last inequality we used that $\overline{z}_{0,e,j}=1$ from the definition of the algorithm and $\overline{p}_{0,e,j}=1$ while $w_{0,e,j}\leq 1$ by (15).

Step 4 (stochastic dominance on $\mathbf{W}_h$). Now that we proved the stochastic dominance of $1/w_{0,e,j}$, we plug this result in the definition of $\mathbf{W}_h$. For the sake of notation, we introduce the term $\overline{p}^{\max}_{h',e,j}$ to indicate the maximum over the first h' step of copy e,j such that

$$\max_{s=0,\dots,h'}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}}=\frac{1}{\overline{p}^{\max}_{h',e,j}}.$$

We first notice that while $\overline{\mathbf{Y}}_h$ is not necessarily PSD, $\mathbf{W}_h$ is a sum of PSD matrices. Introducing the function $\Lambda(\{1/\overline{p}^{\max}_{h,e,j}\}_{e,j})$ we can restate Eq. 14 as

$$\|\mathbf{W}_h\|=\lambda_{\max}(\mathbf{W}_h)\leq\Lambda(\{1/\overline{p}^{\max}_{h,e,j}\}_{e,j})\overset{\text{def}}{=}\lambda_{\max}\left(\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{\overline{p}^{\max}_{h,e,j}}\right)^2\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right).$$

In Step 4, we showed that $1/\overline{p}^{\max}_{h,e,j}$ is stochastically dominated by $1/w_{0,e,j}$ for every e and j. In order to bound $\Lambda(\{1/\overline{p}^{\max}_{h,e,j}\}_{e,j})$, we need to show that this dominance also applies to the summation over all columns inside the matrix norm. We can reformulate $\Lambda(\{1/\overline{p}^{\max}_{h,e,j}\}_{e,j})$ as

$$\begin{aligned}
\lambda_{\max}\left(\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{\overline{p}^{\max}_{h,e,j}}\right)^2\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right) &= \max_{\mathbf{x}:\|\mathbf{x}\|=1}\mathbf{x}^{\mathsf{T}}\left(\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{\overline{p}^{\max}_{h,e,j}}\right)^2\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right)\mathbf{x}\\
= \max_{\mathbf{x}:\|\mathbf{x}\|=1}\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{\overline{p}^{\max}_{h,e,j}}\right)^2\|\mathbf{q}_e\|_2^2\mathbf{x}^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{x} &= \max_{\mathbf{x}:\|\mathbf{x}\|=1}\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{\overline{p}^{\max}_{h,e,j}}\right)^2\left(\|\mathbf{q}_e\|_2\mathbf{q}_e^{\mathsf{T}}\mathbf{x}\right)^2.
\end{aligned}$$

From this reformulation, it is easy to see that, because $1/\overline{p}^{\max}_{h,e,j}$ is strictly positive, the function $\Lambda(\{1/\overline{p}^{\max}_{h,e,j}\}_{e,j})$ is monotonically increasing w.r.t. the individual $1/\overline{p}^{\max}_{h,e,j}$, or in other words that increasing an $1/\overline{p}^{\max}_{h,e,j}$ without decreasing the others can only increase the maximum. Introducing $\Lambda(\{1/w_{0,e,j}\}_{e,j})$ as

$$\Lambda(\{1/w_{0,e,j}\}_{e,j})\overset{\text{def}}{=}\max_{\mathbf{x}:\|\mathbf{x}\|=1}\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{w_{0,e,j}}\right)^2\left(\|\mathbf{q}_e\|_2\mathbf{q}_e^{\mathsf{T}}\mathbf{x}\right)^2,$$

we now need to prove the stochastic dominance of $\Lambda(\{1/w_{0,e,j}\}_{e,j})$ over $\Lambda(\{1/\overline{p}^{\max}_{h,e,j}\}_{e,j})$. Using the definition of $1/\overline{p}^{\max}_{h,e,j}$, $w_{h,e,j}$, and the monotonicity of Λ we have

$$\begin{aligned}\mathbb{P}\left(\Lambda\left(\left\{\frac{1}{\overline{p}^{\max}_{h,e,j}}\right\}_{e,j}\right)\leq a\right) &= \mathbb{P}\left(\Lambda\left(\left\{\max\left\{\max_{s=0,\dots,h-1}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{h,e,j}}{\overline{p}_{h,e,j}}\right\}\right\}_{e,j}\right)\leq a\right)\\ &\geq \mathbb{P}\left(\Lambda\left(\left\{\max\left\{\max_{s=0,\dots,h-1}\frac{\overline{z}_{s,e,j}}{\overline{p}_{s,e,j}};\frac{\overline{z}_{h,e,j}}{w_{h,e,j}}\right\}\right\}_{e,j}\right)\leq a\right).\end{aligned}$$

Now pick $1 \leq k \leq h$, for a fixed $\mathcal{F}_{k-1}$, $\frac{1}{\overline{p}^{\max}_{k-1,e,j}}$ is a constant and $\max\left\{\frac{1}{\overline{p}^{\max}_{k,e,j}};x\right\}$ is a monotonically increasing function in x, making $\Lambda\left(\max\left\{\frac{1}{\overline{p}^{\max}_{k,e,j}};x\right\}\right)$ also an increasing function. Therefore, we have

$$\begin{aligned}&\mathbb{P}\left(\Lambda\left(\left\{\max\left\{\frac{1}{\overline{p}^{\max}_{k-1,e,j}};\frac{\overline{z}_{k,e,j}}{w_{k,e,j}}\right\}\right\}_{e,j}\right)\leq a\right) = \mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\Lambda\left(\left\{\max\left\{\frac{1}{\overline{p}^{\max}_{k-1,e,j}};\frac{\overline{z}_{k,e,j}}{w_{k,e,j}}\right\}\right\}_{e,j}\right)\leq a\;\middle|\;\mathcal{F}_{k-1}\right)\right]\\ &\overset{(a)}{\geq}\mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\Lambda\left(\left\{\max\left\{\frac{1}{\overline{p}^{\max}_{k-1,e,j}};\frac{\overline{z}_{k-1,e,j}}{w_{k-1,e,j}}\right\}\right\}_{e,j}\right)\leq a\;\middle|\;\mathcal{F}_{k-1}\right)\right]\\ &\overset{(b)}{=}\mathop{\mathbb{E}}_{\mathcal{F}_{k-1}}\left[\mathbb{P}\left(\Lambda\left(\left\{\max\left\{\frac{1}{\overline{p}^{\max}_{k-2,e,j}};\frac{\overline{z}_{k-1,e,j}}{w_{k-1,e,j}}\right\}\right\}_{e,j}\right)\leq a\;\middle|\;\mathcal{F}_{k-1}\right)\right],\end{aligned}$$

where inequality (a) follows from the fact that stochastic dominance is preserved by monotonically increasing functions (Levy, 2015), such as Λ, combined with the fact that for a fixed $\mathcal{F}_{k-1}$ the variables $\overline{z}_{k,e,j}$ and $w_{k,e,j}$ are all independent and (b) from the definition of $1/\overline{p}^{\max}_{k-1,e,j}$ and the fact that by definition $1/w_{k-1,e,j}$ is lower-bounded by $1/\overline{p}_{k-1,e,j}$. We can iterate this inequality to obtain the desired result

$$\mathbb{P}(\|\mathbf{W}_h\|\geq\sigma^2)\leq\mathbb{P}\left(\Lambda\left(\left\{\frac{1}{\overline{p}^{\max}_{h,e,j}}\right\}_{e,j}\right)\geq\sigma^2\right)\leq\mathbb{P}\left(\lambda_{\max}\left(\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\left(\frac{1}{w_{0,e,j}}\right)^2\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right)\geq\sigma^2\right).$$

Step 5 (concentration inequality). Since all $w_{0,e,j}$ are (unconditionally) independent from each other, we can apply the following theorem.

Proposition 5 (Tropp (2015), Theorem 5.1.1)**.** *Consider a finite sequence $\{\mathbf{X}_k : k = 1,2,3,\dots\}$ whose values are independent, random, PSD Hermitian matrices with dimension d. Assume that each term in the sequence is uniformly bounded in the sense that*

$$\lambda_{\max}(\mathbf{X}_k)\leq L \quad \textit{almost surely} \quad \textit{for } k=1,2,3,\dots.$$

Introduce the random matrix $\mathbf{V}\overset{\text{def}}{=}\sum_k\mathbf{X}_k$, and the maximum eigenvalue of its expectation

$$\mu_{\max}\overset{\text{def}}{=}\lambda_{\max}(\mathbb{E}[\mathbf{V}])=\lambda_{\max}\left(\sum_k\mathbb{E}[\mathbf{X}_k]\right).$$

Then, for all $h\geq 0$,

$$\begin{aligned}\mathbb{P}\left(\lambda_{\max}(\mathbf{V})\geq(1+h)\mu_{\max}\right)&\leq d\cdot\left[\frac{e^h}{(1+h)^{1+h}}\right]^{\frac{\mu_{\max}}{L}}\\ &\leq d\cdot\exp\left\{-\frac{\mu_{\max}}{L}((h+1)\log(h+1)-h)\right\}.\end{aligned}$$

In our case, we have

$$\mathbf{X}_{\{e,j\}} = \frac{1}{\overline{q}^2}\frac{1}{w_{0,e,j}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}} \preceq \frac{1}{\overline{q}^2}\frac{\alpha^2}{p_{h,e}^2}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}} \preceq \frac{1}{\overline{q}^2}\frac{\alpha^2}{p_{h,e}^2}\|\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\|^2\mathbf{I} \preceq \frac{\alpha^2}{\overline{q}^2}\mathbf{I},$$

where the first inequality follows from the definition of $w_{0,e,j}$ in Eq. 15, the second from the PSD ordering, and the third from the definition of $\|\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\|$.

Therefore, we can use $L \stackrel{\text{def}}{=} \alpha^2/\overline{q}^2$ for the purpose of Prop. 5. We need now to compute $\mathbb{E}\left[\mathbf{X}_k\right]$, that we can use in turn to compute $\mu_{\max}$. We begin by computing the expected value of $1/w_{0,e,j}$. Let us denote the c.d.f. of $1/w_{0,e,j}^2$ as

$$F_{1/w_{0,e,j}^2}(a) = \mathbb{P}\left(\frac{1}{w_{0,e,j}^2} \leq a\right) = \mathbb{P}\left(\frac{1}{w_{0,e,j}} \leq \sqrt{a}\right) = \begin{cases} 0 & \text{for} \quad a < 1 \\ 1 - \frac{1}{\sqrt{a}} & \text{for} \quad 1 \leq a < \alpha^2/p_{h,e}^2 \\ 1 & \text{for} \quad \alpha^2/p_{h,e}^2 \leq a \end{cases}.$$

Since $\mathbb{P}\left(1/w_{0,e,j}^2 \geq 0\right) = 1$, we have that

$$\begin{aligned}
&\mathbb{E}\left[\frac{1}{w_{0,e,j}}\right] = \int_{a=0}^{\infty}\left[1 - F_{1/w_{0,e,j}}(a)\right]\mathrm{d}a \\
&= \int_{a=0}^{1}\left(1 - F_{1/w_{0,e,j}}(a)\right)\mathrm{d}a + \int_{a=1}^{\alpha^2/p_{h,e}^2}\left(1 - F_{1/w_{0,e,j}}(a)\right)\mathrm{d}a + \int_{a=\alpha^2/p_{h,e}^2}^{\infty}\left(1 - F_{1/w_{0,e,j}}(a)\right)\mathrm{d}a \\
&= \int_{a=0}^{1}(1-0)\,\mathrm{d}a + \int_{a=1}^{\alpha^2/p_{h,e}^2}\left(1 - \left(1 - \frac{1}{\sqrt{a}}\right)\right)\mathrm{d}a + \int_{a=\alpha^2/p_{h,e}^2}^{\infty}(1-1)\,\mathrm{d}a \\
&= \int_{a=0}^{1}\mathrm{d}a + \int_{a=1}^{\alpha^2/p_{h,e}^2}\frac{1}{\sqrt{a}}\,\mathrm{d}a = 1 + [2\sqrt{a}]_1^{\alpha^2/p_{h,e}^2} = 2\alpha/p_{h,e} - 1.
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mu_{\max} = \lambda_{\max}(\mathbb{E}\left[\mathbf{V}\right]) &= \lambda_{\max}\Big(\sum_{\{e,j\}}\mathbb{E}\left[\mathbf{X}_{\{e,j\}}\right]\Big) = \lambda_{\max}\left(\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\mathbb{E}\left[\frac{1}{w_{0,e,j}^2}\right]\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right) \\
&= \lambda_{\max}\left(\frac{1}{\overline{q}}\sum_{e=1}^{m}\left(\frac{2\alpha}{p_{h,e}} - 1\right)p_{h,e}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right) \leq \lambda_{\max}\left(\frac{2\alpha}{\overline{q}}\sum_{e=1}^{m}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right) = \frac{2\alpha}{\overline{q}}\lambda_{\max}\left(\mathbf{P}\right) \leq \frac{2\alpha}{\overline{q}} \stackrel{\text{def}}{=} L.
\end{aligned}$$

Therefore, selecting $h = 2$, $\sigma^2 = 6\alpha/\overline{q}$ and applying Prop. 5 we have

$$\begin{aligned}
\mathbb{P}\left(\|\mathbf{W}_h\| \geq \sigma^2\right) &\leq \mathbb{P}\left(\lambda_{\max}\left(\frac{1}{\overline{q}^2}\sum_{j=1}^{\overline{q}}\sum_{e=1}^{m}\frac{1}{w_{0,e,j}^2}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\mathbf{q}_e\mathbf{q}_e^{\mathsf{T}}\right) \geq (1+2)\frac{2\alpha}{\overline{q}}\right) \\
&\leq m \cdot \exp\left\{-\frac{2\alpha}{\overline{q}}\frac{\overline{q}^2}{\alpha^2}(3\log(3) - 2)\right\} \leq n \cdot \exp\left\{-\frac{2\overline{q}}{\alpha}\right\}.
\end{aligned}$$

C.4. Space complexity bound

Denote with A the event $A = \left\{\forall h' \in \{1, \ldots, h\} : \|\mathbf{P}^{h'} - \widetilde{\mathbf{P}}^{h'}\|_2 \leq \varepsilon\right\}$, and again $m = |\mathcal{G}_{\{h,l\}}|$. Letting $q = |\mathcal{H}_{\{h,l\}}| = \sum_{i=1}^{m} q_{h,i} = \sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} z_{h,i,j}$ be the random number of edges in $\mathcal{H}_{\{h,l\}}$, we reformulate

$$\begin{aligned}
&\mathbb{P}\left(|\mathcal{H}_{\{h,l\}}| \geq 3\overline{q}d_{\text{eff}}\gamma_{\{h,l\}} \cap \left\{\forall h' \in \{1, \ldots, h\} : \left(\|\mathbf{P}^{h'} - \widetilde{\mathbf{P}}^{h'}\|_2 \leq \varepsilon\right) \leq \varepsilon\right\}\right) \\
&= \mathbb{P}\left(|\mathcal{H}_{\{h,l\}}| \geq 3\overline{q}d_{\text{eff}}\gamma_{\{h,l\}} \cap A\right) = \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} z_{h,i,j} \geq 3\overline{q}d_{\text{eff}}\gamma_{\{h,l\}} \cap A\right) \\
&= \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} z_{h,i,j} \geq 3\overline{q}d_{\text{eff}}\gamma_{\{h,l\}} \,\middle|\, A\right)\mathbb{P}\left(A\right).
\end{aligned}$$

While we do know that the $z_{h,i,j}$ are Bernoulli random variables (since they are either 0 or 1), it is not easy to compute the success probability of each $z_{h,i,j}$, and in addition there could be dependencies between $z_{h,i,j}$ and $z_{h,i',j'}$. Similarly to Lem. 2, we are going to find a stochastic variable to dominate $z_{h,i,j}$. Denoting with $u'_{s,i,j} \sim \mathcal{U}(0,1)$ a uniform random variable, we will define $w'_{s,i,j}$ as

$$w'_{s,i,j}|\mathcal{F}_{\{s,i',j'\}} = w'_{s,i,j}|\mathcal{F}_{s-2} \stackrel{\text{def}}{=} \mathbb{I}\left\{u'_{s,i,j} \leq \frac{p_{h,i}}{\widetilde{p}_{s-1,i}}\right\} \sim \mathcal{B}\left(\frac{p_{h,i}}{\widetilde{p}_{s-1,i}}\right)$$

for any i' and j' such that $\{s,1,1\} \leq \{s,i',j'\} < \{s,i,j\}$. Note that $w'_{s,i,j}$, unlike $z_{s,i,j}$, does not have a recursive definition, and its only dependence on any other variable comes from $\widetilde{p}_{s-1,i}$. First, we peel off the last step

$$\begin{aligned}\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} z_{h,i,j} \geq g \,\middle|\, A\right) &= \mathop{\mathbb{E}}_{\mathcal{F}_{t-1}|A}\left[\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} \mathbb{I}\left\{u_{h,i,j} \leq \frac{\widetilde{p}_{h,i}}{\widetilde{p}_{t-1,i}}\right\} z_{h-1,i,j} \geq g \,\middle|\, \mathcal{F}_{t-1}\cap A\right)\right]\\
&\leq \mathop{\mathbb{E}}_{\mathcal{F}_{t-1}|A}\left[\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} \mathbb{I}\left\{u'_{h,i,j} \leq \frac{p_{h,i}}{\widetilde{p}_{t-1,i}}\right\} z_{h-1,i,j} \geq g \,\middle|\, \mathcal{F}_{t-1}\cap A\right)\right] = \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{h,i,j} z_{h-1,i,j} \geq g \,\middle|\, A\right),\end{aligned}$$

where we used the fact that conditioned on A, $\mathcal{H}_{\{h,l\}}$ is accurate w.r.t. $\mathbf{L}_{\{h,l\}}$, which guarantees that $\widetilde{p}_{h,i} \leq p_{h,i}$. Plugging this in the previous bound,

$$\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} z_{h,i,j} \geq g \,\middle|\, A\right)\mathbb{P}(A) \leq \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{h,i,j} z_{h-1,i,j} \geq g \cap A\right) \leq \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{h,i,j} z_{h-1,i,j} \geq g\right).$$

We now proceed by peeling off layers from the end of the chain one by one. We show how to move from an iteration $s \leq h$ to $s-1$.

$$\begin{aligned}\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{s,i,j} z_{s-1,i,j} \geq g\right) &= \mathop{\mathbb{E}}_{\mathcal{F}_{s-2}}\left[\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} \mathbb{I}\left\{u'_{s,i,j} \leq \frac{p_{h,i}}{\widetilde{p}_{s-1,i}}\right\} z_{s-1,i,j} \geq g \,\middle|\, \mathcal{F}_{s-2}\right)\right]\\
&= \mathop{\mathbb{E}}_{\mathcal{F}_{s-2}}\left[\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} \mathbb{I}\left\{u'_{s,i,j} \leq \frac{p_{h,i}}{\widetilde{p}_{s-1,i}}\right\}\mathbb{I}\left\{u_{s-1,i,j} \leq \frac{\widetilde{p}_{s-1,i}}{\widetilde{p}_{s-2,i}}\right\} z_{s-2,i,j} \geq g \,\middle|\, \mathcal{F}_{s-2}\right)\right]\\
&= \mathop{\mathbb{E}}_{\mathcal{F}_{s-2}}\left[\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} \mathbb{I}\left\{u'_{s-1,i,j} \leq \frac{p_{h,i}}{\widetilde{p}_{s-2,i}}\right\} z_{s-2,i,j} \geq g \,\middle|\, \mathcal{F}_{s-2}\right)\right] = \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{s-1,i,j} z_{s-2,i,j} \geq g\right)\end{aligned}$$

Applying this repeatedly from $s = h$ to $s = 2$ we have,

$$\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{h,i,j} z_{h-1,i,j} \geq g\right) = \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{1,i,j} z_{0,i,j} \geq g\right) = \mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{1,i,j} \geq g\right).$$

Now, all the $w'_{1,i,j}$ are independent Bernoulli random variables, and we can bound their sum with a Hoeffding-like bound using Markov inequality,

$$\begin{aligned}\mathbb{P}\left(\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m} w'_{1,i,j} \geq g\right) &= \inf_{\theta>0}\mathbb{P}\left(e^{\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m}\theta w'_{1,i,j}} \geq e^{\theta g}\right)\\
&\leq \inf_{\theta>0}\frac{\mathbb{E}\left[e^{\sum_{j=1}^{\overline{q}}\sum_{i=1}^{m}\theta w'_{1,i,j}}\right]}{e^{\theta g}} = \inf_{\theta>0}\frac{\mathbb{E}\left[\prod_{j=1}^{\overline{q}}\prod_{i=1}^{m} e^{\theta w'_{1,i,j}}\right]}{e^{\theta g}} = \inf_{\theta>0}\frac{\prod_{j=1}^{\overline{q}}\prod_{i=1}^{m}\mathbb{E}\left[e^{\theta w'_{1,i,j}}\right]}{e^{\theta g}}\\
&= \inf_{\theta>0}\frac{\prod_{j=1}^{\overline{q}}\prod_{i=1}^{m}(p_{h,i}e^{\theta} + (1-p_{h,i}))}{e^{\theta g}} = \inf_{\theta>0}\frac{\prod_{j=1}^{\overline{q}}\prod_{i=1}^{m}(1 + p_{h,i}(e^{\theta}-1))}{e^{\theta g}}\\
&\leq \inf_{\theta>0}\frac{\prod_{j=1}^{\overline{q}}\prod_{i=1}^{m} e^{p_{h,i}(e^{\theta}-1)}}{e^{\theta g}} \leq \inf_{\theta>0}\frac{e^{\overline{q}(e^{\theta}-1)\sum_{i=1}^{m}p_{h,i}}}{e^{\theta g}} = \inf_{\theta>0} e^{(d_{\text{eff}}\gamma_{\{h,l\}}\overline{q}(e^{\theta}-1)-\theta g)} \leq \inf_{\theta>0} e^{(d_{\text{eff}}\gamma_{\{h,l\}}\overline{q}(e^{\theta}-1)-\theta g)},\end{aligned}$$

where we use the fact that $1 + x \leq e^x$, $w'_{1,i,j} \sim \mathcal{B}(p_{h,i})$ and by Def. **??**, $\sum_{i=1}^m p_{h,i} = \sum_{i=1}^m \tau_{h,i} = d_{\text{eff}}\gamma_{\{h,l\}}$. The choice of θ minimizing the previous expression is obtained as

$$\frac{d}{d\theta} e^{\left(\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}(e^\theta - 1) - \theta g\right)} = e^{\left(\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}(e^\theta - 1) - \theta g\right)} \left(\overline{q} d_{\text{eff}}\gamma_{\{h,l\}} e^\theta - g\right) = 0,$$

and thus $\theta = \log(g/(\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}))$. Plugging this in the previous bound,

$$\begin{aligned} \inf_\theta \exp\left\{\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}(e^\theta - 1) - \theta g)\right\} &= \exp\left\{g - \overline{q} d_{\text{eff}}\gamma_{\{h,l\}} - g \log\left(\frac{g}{\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}}\right)\right\} \\ &= \exp\left\{-g\left(\log\left(\frac{g}{\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}}\right) - 1\right)\right\} e^{-\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}}, \end{aligned}$$

and choosing $g = 3\overline{q} d_{\text{eff}}\gamma_{\{h,l\}}$, we conclude our proof.