Generalized Flow in Nearly-linear Time on Moderately Dense Graphs

Shunhua Jiang* Michael Kapralov† Lawrence Li‡ Aaron Sidford§

Abstract

In this paper we consider generalized flow problems where there is an m-edge n-node directed graph $G = (V, E)$ and each edge $e \in E$ has a loss factor $\gamma_e > 0$ governing whether the flow is increased or decreased as it crosses edge e. We provide a randomized $\widetilde{O}((m+n^{1.5}) \cdot \text{polylog}(\frac{W}{\delta}))$ time algorithm for solving the generalized maximum flow and generalized minimum cost flow problems in this setting where δ is the target accuracy and W is the maximum of all costs, capacities, and loss factors and their inverses. This improves upon the previous state-of-the-art $\widetilde{O}(m\sqrt{n} \cdot \log^2(\frac{W}{\delta}))$ time algorithm, obtained by combining the algorithm of [DS08] with techniques from [LS14]. To obtain this result we provide new dynamic data structures and spectral results regarding the matrices associated to generalized flows and apply them through the interior point method framework of [vdBLL+21].

*sj3005@columbia.edu. Columbia University.

†michael.kapralov@epfl.ch. École Polytechnique Fédérale de Lausanne

‡lawrenceli@cs.toronto.edu. University of Toronto.

§sidford@stanford.edu. Stanford University.

Contents

1 Introduction

In this paper we consider *generalized flow problems* where there is a directed graph $G = (V, E)$ with n-nodes V, m-edges E, together with positive loss factors $\gamma \in \mathbb{R}^E_{>0}$ for edges, and positive edge capacities $u \in \mathbb{R}^E_{>0}$. We call any $f \in \mathbb{R}^E_{\geq 0}$ a *flow* and define the *imbalance* of f at $a \in V$, as

$$\mathrm{im}_G(f)_a \stackrel{\text{def}}{=} \sum_{e=(b,a)\in E} \gamma_e f_e - \sum_{e=(a,b)\in E} f_e.$$

The imbalance of f at a denotes the net amount of flow into a. Whereas in standard flow problems increasing the flow on edge $e = (a, b)$ by α decreases the imbalance at a by α and increases it at b by α, here the increase at b is instead $\gamma_e\alpha$. Consequently, $\gamma_e < 1$ can be viewed as flow leaving the graph, i.e., being *lost*, as flow crosses edge e, and we refer to such problems with $\gamma_e < 1$ as *lossy flow problems*. Conversely, $\gamma_e > 1$ can be viewed as flow entering the graph as the flow crosses e.

In this paper we consider the problems of solving[1] canonical optimization problems for graphs with loss factors. In particular we consider the following:

- *Generalized Maximum Flow*: In this problem there is a specified $s, t \in V$ and the goal is to find flow $f \in \mathbb{R}^E_{\geq 0}$ with $f \leq u$ entrywise and $\mathrm{im}_G(f)_a = 0$ for all $a \notin \{s, t\}$ such that $\mathrm{im}_G(f)_t$ is maximized. This corresponds to maximizing the flow into t given an unlimited supply at s while respecting the capacity constraints.

- *Generalized Minimum Cost Flow*: In this problem there is a specified $d \in \mathbb{R}^V$ and $c \in \mathbb{R}^E$ and the goal is to find $f \in \mathbb{R}^E_{\geq 0}$ with $f \leq u$ entrywise and $\mathrm{im}_G(f) = d$ such that $c^\top f$ is minimized.

In the special case when $\gamma_e = 1$ for all $e \in E$, the generalized maxflow problem is the standard maxflow problem and the generalized min-cost flow problem is the standard min-cost flow problem. A line of work over the past decade has obtained numerous improvements to the running times for solving these standard problems [CKM+11, Mad16, LS20, vdBLSS20, vdBLN+20, vdBLL+21, GLP22, BGS22]. This line of work culminated in [CKL+23, BCK+23, VDBCK+24], which give deterministic almost linear time $O(m^{1+o(1)})$ algorithms for solving maximum flow and minimum cost flow to high-accuracy, and a randomized algorithm [vdBLL+21] that solves these problems in nearly linear time $\widetilde{O}(m + n^{1.5})$ on moderately dense graphs where $m \geq n^{1.5}$.[2] Interestingly, the almost linear time algorithms even apply to more general convex flow problems where the goal is to minimize sums of convex cost functions applied to the edges while routing specified demands d:

$$\min_{\substack{f\in\mathbb{R}^E:\ \mathrm{im}(f)_v = d_v\ \forall v\in V \\ 0\leq f\leq u}} \sum_{e\in E} \phi_e(f_e), \text{ where each } \phi_e : \mathbb{R} \to \mathbb{R} \text{ is a convex.}$$

However, there is no known almost linear time reduction from generalized flow problems to min-cost flow, or even to this more general convex flow problem.

Despite these advances and multiple works studying generalized flows [GPT88, Vai89, GJO97, FW02, DS08], the state-of-the-art running times for solving generalized maxflow and generalized min-cost flow are substantially slower than those for maximum flow and minimum cost flow. These

[1] Unless specified otherwise, we use "solving" to refer to producing a high-accuracy approximately-feasible solution.

[2] $\widetilde{O}(m)$ hides polylog(m) factors. Almost linear time refers to $O(m^{1+o(1)})$ running times, and nearly linear time refers to $\widetilde{O}(m)$ running times. In just the introduction, we assume that the problem magnitude parameter W and the error parameter $1/\delta$ are both bounded by poly(n), and hide polylog(W/δ) factors in $\widetilde{O}(\cdot)$.

state-of-the-art running times include the $\widetilde{O}(m^{1.5})$ time algorithm of [DS08],[3] as well as a $\widetilde{O}(m\sqrt{n})$ time algorithm, obtained by combining the algorithm of [DS08] with techniques from [LS14].

The central goal of this paper is to close this gap. We consider the problem of designing faster algorithms for solving the generalized flow problem and more broadly, developing new algorithmic tools for reasoning about linear programs with at most two variables per inequality. Our main result is a nearly linear time algorithm for solving generalized maximum flow and generalized minimum cost flow to high accuracy on sufficiently dense graphs. To obtain this result, we provide new tools to reason about the linear algebraic structure of matrices associated with generalized flows and new dynamic data structures for maintaining information about them. We then apply these data structures in the interior point method (IPM) optimization framework of [vdBLL+21]. Beyond faster running times, we hope that the spectral graph theory tools we develop for generalized flows will have broader applications and facilitate faster algorithms for a wider range of problems.

1.1 Our Results

Let $\mathbf{B}_G \in \mathbb{R}^{E \times V}$ denote the edge-incidence matrix of the lossy graph $G = (V, E, \gamma)$, where for every edge $e = (a, b)$, the corresponding row e of $\mathbf{B}_G$ has γ_e on entry b and -1 on entry a, and therefore $(\mathbf{B}_G^\top f)_v = \mathrm{im}_G(f)_v$ for all $v \in V$. Let $\mathbf{B}_{G\setminus\{s,t\}} \in \mathbb{R}^{E \times V\setminus\{s,t\}}$ denote the submatrix of $\mathbf{B}_G$ obtained by deleting the columns corresponding to vertices s and t. The generalized maximum flow problem can be formulated as the following linear program:

$$\min_{\substack{\mathbf{B}_{G\setminus\{s,t\}}^\top f = 0 \\ \mathbb{0} \le f \le u}} (\mathbf{B}_G^\top f)_t.$$

The generalized minimum cost flow problem can be formulated as the following linear program:

$$\min_{\substack{\mathbf{B}_G^\top f = d \\ \mathbb{0} \le f \le u}} c^\top f.$$

To obtain our generalized flow algorithms, we consider the more general problem of solving the following linear program where every row of $\mathbf{A} \in \mathbb{R}^{m \times n}$ has at most two non-zero entries:

$$\min_{\substack{\mathbf{A}^\top x = b \\ \ell \le x \le u}} c^\top x. \tag{1}$$

This problem is the dual of a well-studied problem called the two variable per inequality (2VPI) linear program [Meg83, Hoc04]. Additionally, it clearly encompasses the generalized maximum flow and generalized minimum cost flow problems.[4]

We refer to the linear program (1) as a *two-sparse LP*. Additionally, we define matrices with at most two non-zero entries per row as *two-sparse matrices* and measure their magnitude as follows.

Definition 1.1 (Two-sparse matrix and its magnitude parameter)**.** *We say a matrix* $\mathbf{A}$ *is a* two-sparse matrix *if every row of* $\mathbf{A}$ *has at most two non-zero entries. For any matrix* $\mathbf{A}$*, we define*

$$W_{\mathbf{A}} \stackrel{\text{def}}{=} \max_{i,j:\mathbf{A}_{i,j}\neq 0} \left(\max(|\mathbf{A}_{i,j}|, \frac{1}{|\mathbf{A}_{i,j}|}) \right).$$

[3] The algorithm of [DS08] was stated for lossy maximum flow, for which it produces *exact feasible* solutions. Careful inspection shows that their algorithm can also be applied to generalized maximum flow and generalized min-cost flow to produce *approximate feasible* solutions. Our algorithm also outputs approximate feasible solutions for generalized maximum flow and generalized min-cost flow, and the precise guarantees are stated in Theorem 1.4. For lossy maximum flow, however, we can obtain exact feasible solutions using the same techniques as [DS08].

[4] We write $\mathbf{A}^\top x = b$ (instead of $\mathbf{A}x = b$) in (1) so that generalized flow problems are special cases with $\mathbf{A} = \mathbf{B}_G$.

Our main result is that we can solve two-sparse LPs to δ-accuracy in $\widetilde{O}((m+n^{1.5})\cdot\text{poly}\log(W/\delta))$ time, where δ-accuracy is defined in the theorem below. As a corollary, we can also solve the generalized maxflow and generalized min-cost flow problems to δ-accuracy in $\widetilde{O}((m+n^{1.5})\cdot\text{poly}\log(W/\delta))$ time, where W is an upper bound on the magnitude of any integer used to describe the problem instance, and δ is an additive error parameter.

Theorem 1.2 (Two-sparse LP). *Let $\mathbf{A}\in\mathbb{R}^{m\times n}$ be a two-sparse matrix, $c,\ell,u\in\mathbb{R}^m$, and $b\in\mathbb{R}^n$. Let $W\stackrel{\text{def}}{=}\max\left(W_{\mathbf{A}},\|c\|_\infty,\|b\|_\infty,\|u\|_\infty,\|\ell\|_\infty,\frac{\max_i(u_i-\ell_i)}{\min_i(u_i-\ell_i)}\right)$. Assume that there is a point x satisfying $\mathbf{A}^\top x=b$ and $\ell\le x\le u$. There exists an algorithm that given any such $\mathbf{A}$ and any $\delta>0$, runs in time $\widetilde{O}((m+n^{1.5})\cdot\text{poly}\log(\frac{W}{\delta}))$ and w.h.p.[5] outputs a vector $x^{(\text{final})}$ satisfying*

$$\|\mathbf{A}^\top x^{(\text{final})}-b\|_\infty\le\delta\quad\text{and}\quad\ell\le x^{(\text{final})}\le u\quad\text{and}\quad c^\top x^{(\text{final})}\le\min_{\substack{\mathbf{A}^\top x=b\\ \ell\le x\le u}}c^\top x+\delta.$$

Applying Theorem 1.2, we obtain the following results for the generalized maxflow (Theorem 1.3) and generalized min-cost flow (Theorem 1.4) problems. In the special case of lossy maxflow, we further apply the technique of [DS08] to obtain an exactly feasible flow.

Theorem 1.3 (Generalized maxflow). *There exists an algorithm that, given any lossy graph $G=(V,E,\gamma)$, source $s\in V$, sink $t\in V$, capacities $u\in\mathbb{R}^E_{\ge0}$, and $\delta>0$, for which there exists $f\in\mathbb{R}^E_{\ge0}$ such that $f\le u$ and $\text{im}_G(f)_v=0$ for all $v\in V\backslash\{s,t\}$, runs in time*

$$\widetilde{O}\Big((m+n^{1.5})\cdot\text{poly}\log(W/\delta)\Big)\text{ where }W\stackrel{\text{def}}{=}\max\left(\|\gamma\|_\infty,\|\gamma^{-1}\|_\infty,\|u\|_\infty,\frac{\max_e u_e}{\min_e u_e}\right)$$

and w.h.p. outputs $f\in\mathbb{R}^E_{\ge0}$ satisfying $f\le u$,

$$|\text{im}_G(f)_v|\le\delta\ \forall v\in V\backslash\{s,t\},\quad\text{and}\quad\text{im}_G(f)_t\le\min_{\substack{\text{im}_G(f')_v=0\,\forall v\in V\backslash\{s,t\}\\ 0\le f'_e\le u_e\forall e\in E}}\text{im}_G(f')_t+\delta.$$

In the special case of lossy maxflow (i.e., when $\gamma\le\mathbb{1}_V$), the algorithm outputs an exactly feasible flow, i.e., it has the additional property that $\text{im}_G(f)_v=0\ \forall v\in V\backslash\{s,t\}$.

Theorem 1.4 (Generalized min-cost flow). *There exists an algorithm that, given any lossy graph $G=(V,E,\gamma)$, costs $c\in\mathbb{R}^E$, capacities $u\in\mathbb{R}^E_{\ge0}$, demands $d\in\mathbb{R}^V$, and $\delta>0$, for which there exists $f\in\mathbb{R}^E_{\ge0}$ such that $f\le u$ and $\text{im}_G(f)_v=d_v$ for all $v\in V$, runs in time*

$$\widetilde{O}\Big((m+n^{1.5})\cdot\text{poly}\log(W/\delta)\Big)\text{ where }W\stackrel{\text{def}}{=}\max\left(\|\gamma\|_\infty,\|\gamma^{-1}\|_\infty,\|c\|_\infty,\|u\|_\infty,\frac{\max_e u_e}{\min_e u_e}\right)$$

and w.h.p. outputs $f\in\mathbb{R}^E_{\ge0}$ satisfying $f\le u$,

$$|\text{im}_G(f)_v-d_v|\le\delta\ \forall v\in V,\quad\text{and}\quad c^\top f\le\min_{\substack{\text{im}_G(f')_v=d_v\forall v\in V\\ 0\le f'_e\le u_e\forall e\in E}}c^\top f'+\delta.$$

To obtain these results, our key technical contribution is a *dynamic heavy hitter data structure* that can efficiently find all heavy entries $|\mathbf{A}h|_i\ge\epsilon$ for any two-sparse matrix $\mathbf{A}$ and any query vector h. We apply a reduction from [vdBLN+20, vdBLL+21] to show that to prove Theorem 1.2 it suffices to design this data structure (along with additional data structures we provide). Our heavy hitter data structure relies on spectral graph theory tools that we develop to analyze the lossy Laplacian, which is an analog of the standard Laplacian that we introduce for lossy graphs. We believe both the data structures and these spectral results may be of independent interest.

[5] We use "w.h.p." as an abbreviation for "with high probability", which means for any arbitrarily large constant specified in advance, the event happens with probability at least $1-1/m^c$, where m is the input size.

1.2 Related Work

Generalized flow. The generalized maxflow and generalized min-cost flow problems are well-studied and both combinatorial and continuous optimization based algorithms have been developed for them. We provide a brief summary of known weakly polynomial time algorithms for solving these problems in Table 1. For further references, we refer readers to the discussions in these papers and Chapter 15 of [AMO+93]. Prior to our result, the state-of-the-art algorithm for solving generalized flow was given by [DS08], which can be improved to $\widetilde{O}(m\sqrt{n})$ time by combining techniques from [LS14]. A key technical contribution of [DS08] is an M-matrix scaling lemma that reduces solving linear systems in M-matrices to solving Laplacian linear systems, allowing the use of Laplacian solvers of [ST04] to implement each IPM iteration in nearly linear time. We refer readers to [AJSS19] for further discussion and improvements to solving M-matrices.

Year	Authors	Problem	Exact or approx	Time
1988	Goldberg, Plotkin, Tardos [GPT88]	Max	Exact	m^2n^2
1989	Vaidya [Vai89]	Min-Cost	Exact	$m^{1.5}n^2$
1997	Goldfarb, Jin, Orlin [GJO97]	Max	Exact	m^3
2002	Wayne [Way02]	Min-Cost	Exact	m^2n^3
2002	Fleischer, Wayne [FW02]	Min-Cost	$(1+\delta)$-mult. approx	$m^2\delta^{-2}$
2008	Daitch, Spielman [DS08]	Min-Cost	δ-additive error	$m^{1.5}$
2014	Lee, Sidford [LS14]	Min-Cost	δ-additive error	$m\sqrt{n}$
2025	Our Work	Min-Cost	δ-additive error	$m+n^{1.5}$

Table 1: A summary of weakly polynomial algorithms for solving the generalized flow problems. All time complexities hide $n^{o(1)}$ and $\operatorname{poly}\log(W)$ factors, and the time complexities for approximate algorithms hide $\operatorname{polylog}(\delta^{-1})$ factors, where W and δ are defined as in Theorem 1.4.

.

Strongly polynomial algorithms for solving 2VPI. Recently, [DKN+24] provided the first strongly polynomial algorithm for solving generalized min-cost flow, and hence for linear programs with two variables per inequality (2VPI). Prior work had extensively studied and provided strongly polynomial algorithms for solving the primal and dual feasibility variants of the problem [Meg83, CM91, HN94, Vég14, Kar22]. For further references we refer the readers to the discussion in [DKN+24]. Note that this line of work often considers the LP formulation of 2VPI with one-sided constraints $x \geq 0$ instead of two-sided constraints $\ell \leq x \leq u$. Such a restriction is without loss of generality because the 2VPI formulation with two-sided constraints can be reduced to a formulation with one-sided constraints at the cost of increasing the number of constraints from n to $m+n$. Since, in this paper we seek running times which depend on both m and the ratio m/n, we explicitly consider the 2VPI formulation with two-sided constraints, as in (1).

Maximum flow and minimum cost flow. Before the breakthrough almost linear time maxflow and min-cost flow algorithm of [CKL+23], multiple advances had been made to improve the time complexity of IPM algorithms for solving these flow problems including [CKM+11, Mad13, LS14, Mad16, CMSV17, LS20, KLS20, AMV20, AMV22, vdBGJ+22]. There is also a recent, related line of work on improving combinatorial algorithms for solving flow problems [BBST24, CK24].

1.3 Organization

In Section 2, we give an overview of our approach. In Section 3 we cover general notation and known technical tools. In Section 5 and Section 4, we prove some spectral theorems about our lossy Laplacian. In Section 6 and Section 7, we use these spectral theorems to construct heavy hitter and sampler algorithms on lossy graph incidence matrices and 2-sparse matrices. Finally, in Section 8, we show how the heavy hitter and sampler algorithms fit into an IPM that can solve the generalized flow and 2-sparse LP problems.

2 Technical Overview

In this section we give an overview of our approach. We solve two-sparse LPs by applying the general interior point method (IPM) framework of [vdBLN+20, vdBLL+21] (see Theorem 8.11), which applies to linear programs of the form (1) with any constraint matrix $\mathbf{A}$. The framework essentially reduces solving the LP to designing three data structures for the constraint matrix $\mathbf{A}$: (1) a *heavy hitter data structure* that finds the entries of $\mathbf{A}h$ with large absolute value, (2) a *sampler data structure* that samples entries according to a distribution defined by $\mathbf{A}h$, and (3) an *inverse maintenance data structure* that solves linear equations in $\mathbf{A}^\top\mathbf{V}\mathbf{A}$, where $\mathbf{V}$ is a non-negative diagonal matrix. The IPM has $\widetilde{O}(\sqrt{n}\log(\frac{W}{\delta}))$ iterations and runs in time which depends on the time of implementing these data structures.

We obtain our results by providing efficient data structures for each of these three data structure problems for two-sparse $\mathbf{A}$. To motivate our approach and explain our results, in this overview we focus primarily on designing the heavy hitter data structure. This task captures the main difficulty of the problem and illustrates our key ideas for designing our data structures. More specifically, the heavy hitter problem is to find the *heavy hitters* of $\mathbf{A}h$, the entries of $\mathbf{A}h$ that are greater than ϵ in absolute value, where $\mathbf{A}$ is a two-sparse matrix that undergoes a bounded number of updates. Given any $\mathbf{A}$ and ϵ, it is easy to see there are at most $\epsilon^{-2}\|\mathbf{A}h\|_2^2$ heavy hitters, and this bound is our target time complexity for answering heavy hitter queries. Our main theorem regarding a heavy hitter data structures for two-sparse matrices is given informally below.

Theorem 2.1 (Heavy hitter for two-sparse matrices, informal version of Theorem 7.1)**.** *There is a data structure* HeavyHitter *that can be initialized with any two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ in $\widetilde{O}(m)\cdot\text{polylog}(\frac{W}{\delta})$ time, and supports inserting or deleting any row to $\mathbf{A}$ in $\widetilde{O}(1)\cdot\text{polylog}(\frac{W}{\delta})$ time, and querying for all heavy hitters i with $|(\mathbf{A}h)_i| \geq \epsilon$ in $\widetilde{O}\left(\epsilon^{-2}\|\mathbf{A}h\|_2^2 + n\right)\cdot\text{polylog}(\frac{W}{\delta})$ time.*

Theorem 7.1 allows us to support all heavy hitter queries of the IPM in $\widetilde{O}(m+n^{1.5})\cdot\text{polylog}(\frac{W}{\delta})$ time. When we apply the associated algorithm in each iteration of the IPM, the IPM guarantees that the sum of these $\|\mathbf{A}h\|_2^2$ across all iterations is $\widetilde{O}(m)\cdot\text{polylog}(\frac{W}{\delta})$. If we support each heavy hitter query in $\widetilde{O}\left(\epsilon^{-2}\|\mathbf{A}h\|_2^2 + n\right)\cdot\text{polylog}(\frac{W}{\delta})$ time and each row update of $\mathbf{A}$ in $\widetilde{O}(1)\cdot\text{polylog}(\frac{W}{\delta})$ time, then we can obtain our desired runtime of $\widetilde{O}(m+n^{1.5})\cdot\text{polylog}(\frac{W}{\delta})$ across all iterations.

A key ingredient of our proof of the Theorem 7.1 is a heavy hitter data structure for the special case of incidence matrices $\mathbf{B}_G$ of what we call *balanced lossy expanders*. These are unweighted lossy graphs G whose associated non-lossy graphs $\overline{G}$ are expanders, and have flow multipliers $(1-\beta)\mathbb{1}_E \leq \gamma \leq \mathbb{1}_E$ for a sufficiently small β. These graph can be viewed as a natural analogue of unweighted expanders for standard graphs. Perhaps the main technical contribution of this work is developing sufficient spectral graph theory tools, to obtain this heavy hitter data structure.

In Section 2.1 we first review how [vdBLN+20] solves the heavy hitter problem for graphs, and introduce an approach based on *balanced lossy expanders*. In Section 2.2, we discuss our

approach for solving the heavy hitter problem on balanced lossy expanders. We first discuss the error requirements for our approaches to work, and then present the key spectral results. In Section 2.3 we present a proof sketch for Theorem 2.1 by using heavy hitters on balanced lossy expanders. Finally in Section 2.4 we introduce our other two data structures, the sampler and inverse maintenance, which together with the heavy hitter yield our $\widetilde{O}(m+n^{1.5})\cdot\text{polylog}(\frac{W}{\delta})$ time algorithm for solving two-sparse LPs (Theorem 1.2).

2.1 Expander Decomposition Based Approaches

Expander Decompositions for Graphs and Prior Approach. We first discuss the strategy used in [vdBLN+20, vdBLL+21] for solving the heavy hitter problem on the edge-vertex incidence matrix $\mathbf{B}_{\overline{G}} \in \mathbb{R}^{E\times V}$ of a graph $\overline{G}$. They observe that the heavy hitter problem is decomposable in that if there exists an edge decomposition $\overline{G} = \overline{G}_1 \cup \overline{G}_2$, to find the heavy hitters of $\mathbf{B}_{\overline{G}}h = \begin{bmatrix}\mathbf{B}_{\overline{G}_1}\\ \mathbf{B}_{\overline{G}_2}\end{bmatrix} h$, it suffices to find all heavy hitters of $\mathbf{B}_{\overline{G}_1}h$ and $\mathbf{B}_{\overline{G}_2}h$ separately. They use standard tools [SW19] to decompose $\overline{G}$ into expanders with $1/\text{polylog}(n)$ expansion and solve the problem on the expanders.

To solve the heavy hitter problem when $\overline{G}$ is an expander, [vdBLN+20] uses that the Laplacian $\mathbf{L}_{\overline{G}} = \mathbf{B}_{\overline{G}}^\top \mathbf{B}_{\overline{G}}$ can be spectrally approximated by a diagonal matrix minus a rank-1 matrix. For simplicity of presentation, in this section we assume the graph is unweighted and d-regular, in which case this spectral approximation is

$$\mathbf{L}_{\overline{G}} \approx_{\text{polylog}(n)} \mathbf{D} - \frac{d}{n}\mathbb{1}_V\mathbb{1}_V^\top, \tag{2}$$

where $\mathbf{D} = d\cdot\mathbf{I}$ is the degree matrix of $\overline{G}$ and $\mathbb{1}_V$ is the all-ones vector. This spectral approximation implies that for any vector h orthogonal to $\mathbb{1}_V$, $h^\top\mathbf{D}h \approx_{\text{polylog}(n)} h^\top\mathbf{L}_{\overline{G}}h = \|\mathbf{B}_{\overline{G}}h\|_2^2$.

Leveraging this spectral approximation fact, [vdBLN+20] provides the following simple algorithm that finds the heavy hitters in $\widetilde{O}(\|\mathbf{B}_{\overline{G}}h\|_2^2\epsilon^{-2})$ time. Given any query vector h, [vdBLN+20] first subtracts a constant from all entries of h to obtain a shifted vector h' that is orthogonal to the all-ones vector. Such constant shifts do not affect the heavy hitter problem, since $\mathbf{B}_{\overline{G}}\mathbb{1}_V = 0$. (More broadly, $\ker(\mathbf{B}_{\overline{G}}) = \text{span}(\mathbb{1}_V)$.) They then solve the heavy hitter problem with the shifted vector h' by using that a row of $\mathbf{B}_{\overline{G}}$, e.g., $\vec{1}_u - \vec{1}_v$, can only be a heavy hitter if either h'_u or h'_v has absolute value greater than $\epsilon/2$. To find all heavy hitters, the algorithm of [vdBLN+20] checks the adjacent edges of all vertices for which $|h'_v| \geq \epsilon/2$. Checking the adjacent edges of one vertex can be done in $O(d)$ time, and in total this step takes $O(\sum_v d(h'_v)^2/\epsilon^2) = O(\epsilon^{-2}\cdot(h')^\top\mathbf{D}h') = \widetilde{O}(\|\mathbf{B}_{\overline{G}}h\|_2^2\epsilon^{-2})$ time, as desired (where in the last equality we used (2)).

Expander Decompositions for Lossy Graphs. Now let's consider applying an analogous approach to unweighted lossy graphs. With a slight notational overload, we consider the equivalent formulation where instead of working with loss factors γ we work with *flow multipliers* η, where $\eta \stackrel{\text{def}}{=} \gamma^{-1} \geq \mathbb{1}_E$, and denote the lossy graph as $G = (V, E, \eta)$. We define $\mathbf{B}_G \in \mathbb{R}^{E\times V}$ as the incidence matrix of G which has $\vec{1}_{b_e} - \eta_e\vec{1}_{a_e}$ as row $e = (a_e, b_e)$ of $\mathbf{B}_G$ where $\vec{1}_{a_e}$ and $\vec{1}_{b_e}$ are indicator vectors with 1 on a_e and b_e respectively. These two formulations are equivalent, and we choose to use flow multipliers $\eta_e \geq \mathbb{1}_E$ since it makes it cleaner to describe balanced lossy graphs where $\eta_e \leq 1+\beta_\eta$. For a lossy graph $G = (V, E, \eta)$, we call $\mathbf{L}_G = \mathbf{B}_G^\top\mathbf{B}_G$ the *lossy Laplacian* of G. We denote by $\overline{G} = (V, E)$ the *smoothed graph* of G, which is a graph with no flow multipliers and the same vertices and edges as G and let $\mathbf{L}_{\overline{G}} \stackrel{\text{def}}{=} \mathbf{B}_{\overline{G}}^\top\mathbf{B}_{\overline{G}}$ denote the Laplacian of $\overline{G}$.

We first observe that the heavy hitter problem for a lossy graph $\mathbf{B}_G$ is still decomposable: Given any edge decomposition $G = G_1 \cup G_2$, it suffices to find all heavy hitters of $\mathbf{B}_{G_1}h$ and $\mathbf{B}_{G_2}h$

separately. Our first attempt is to compute an expander decomposition on the smoothed graph $\overline{G}$, and use the same decomposition on our lossy graph G. However, even if the underlying smoothed graph $\overline{G}$ is an expander, the corresponding lossy Laplacian $\mathbf{L}_G$ does not necessarily have a large enough gap between the least and second least eigenvalues $\lambda_1(\mathbf{L}_G)$ and $\lambda_2(\mathbf{L}_G)$. For example, if the incidence matrix of G is $\mathbf{B}_G = \begin{pmatrix} 1 & -\frac{1}{x} & 0 \\ 0 & 1 & -\frac{1}{x} \\ -\frac{1}{x} & 0 & 1 \end{pmatrix}$, then its underlying smoothed graph $\overline{G}$ has $\lambda_1(\mathbf{L}_{\overline{G}}) = 0$ and $\lambda_2(\mathbf{L}_{\overline{G}}) = 3$. However, the lossy Laplacian $\mathbf{L}_G$ has $\lambda_1(\mathbf{L}_G)$ and $\lambda_2(\mathbf{L}_G)$ that both tend to 1 as x goes to infinity, and consequently a gap between $\lambda_1(\mathbf{L}_G)$ and $\lambda_2(\mathbf{L}_G)$ that goes to 0.

To overcome this obstacle, our first spectral result for lossy graphs is the following lemma that shows that if we have the extra condition where the flow multipliers η_e are all relatively close to one, then the lossy graph indeed has a large enough spectral gap.

Lemma 2.2 (Informal version of Theorems 4.3 and 4.4)**.** *If lossy graph $G = (V, E, \eta)$ satisfies $\mathbb{1}_E \leq \eta \leq (1+\beta_\eta)\mathbb{1}_E$ where $\beta_\eta \leq 1/\text{polylog}(n)$, and the underlying smoothed graph $\overline{G}$ is a d-regular expander with conductance at least $20\beta_\eta$, then the least eigenvalue $\lambda_{\min}$ of $\mathbf{L}_G$ and its corresponding unit eigenvector $v_{\min}$ satisfy*

$$\mathbf{L}_G \approx_{\text{polylog}(n)} \mathbf{D} - d(1-\lambda_{\min})v_{\min}v_{\min}^\top. \tag{3}$$

We use the term *balanced lossy expanders* to refer to such unweighted lossy graphs that have both a small flow multiplier, as well as an underlying smoothed graph that is an expander. We show that any lossy graph can be decomposed into a collection of such balanced lossy expanders by partitioning its edges, applying a suitable vertex scaling, and performing expander decomposition to the underlying smoothed graph. The total number of vertices across all subgraphs in this decomposition is $O(n \cdot \text{polylog}(nW))$ (more formal statements can be found in Section 7, where edge partition and vertex scaling are shown in Section 7.2, and expander decomposition is shown in Section 7.1). As such, from now on we focus on balanced lossy expanders.

Leveraging (3), if we could somehow explicitly compute the exact eigenvector $v_{\min}$ and $\mathbf{B}_G v_{\min}$ in each iteration, with the latter given in sorted order, then the following natural generalization of the algorithm in [vdBLL+21, vdBLN+20] would solve the heavy hitter problem for $\mathbf{B}_G$. Given a query vector h, the algorithm projects h onto the direction of $v_{\min}$ rather than the all-ones vector as before, i.e., $h^{v_{\min}} = v_{\min}v_{\min}^\top h$, and the algorithm also computes the component orthogonal to $v_{\min}$, i.e., $h^{\perp v_{\min}} = (\mathbf{I} - v_{\min}v_{\min}^\top)h$. The algorithm then finds the heavy hitters of these two vectors separately. In the direction perpendicular to $v_{\min}$, similar as before, the algorithm checks the adjacent edges of all vertices i for which $|(h^{\perp v_{\min}})_i| \geq \epsilon/2$. This step takes time $O((h^{\perp v_{\min}})^\top \mathbf{D} h^{\perp v_{\min}})$, which can be bounded by the spectral approximation in Theorem 2.2: $(h^{\perp v_{\min}})^\top \mathbf{D} h^{\perp v_{\min}} \approx_{\text{polylog}(n)} (h^{\perp v_{\min}})^\top \mathbf{L}_G h^{\perp v_{\min}} = \|\mathbf{B}_G h^{\perp v_{\min}}\|_2^2 \leq \|\mathbf{B}_G h\|_2^2$. In the direction of $v_{\min}$, since an appropriately sorted $\mathbf{B}_G v_{\min}$ has already been computed, finding the heavy hitters of $\mathbf{B}_G h^{v_{\min}}$ is straightforward, as $\mathbf{B}_G h^{v_{\min}}$ is simply a scalar multiple of $\mathbf{B}_G v_{\min}$.

Difficulties Generalizing. Unfortunately, this approach encounters two immediate obstacles.

First, the smallest eigenvector $v_{\min}$ is not known. For lossy graphs, the smallest eigenvector is not necessarily the all-ones vector as it is for graphs. The eigenvector $v_{\min}$ can be computed to ϵ accuracy using the power method and lossy Laplacian solvers in $\widetilde{O}(m \log(1/\epsilon))$ time (see Theorem 4.9). However, this runtime is prohibitively expensive for each heavy hitter query. An alternative would be to efficiently maintain a coarser approximation of $v_{\min}$, however this would require care regarding the level of approximation. For example, it is no longer necessarily true

that $\|\mathbf{B}_G h^{\perp v}\|_2^2 \leq O(\|\mathbf{B}_G h\|_2^2)$, when v is only a coordinate-wise multiplicative $(1 \pm 1/\text{polylog})$ approximation to the smallest eigenvector $v_{\min}$.

Second, the vector $\mathbf{B}_G v_{\min}$ is also not known. Even if $v_{\min}$ were given explicitly, computing $\mathbf{B}_G v_{\min}$ would still take $O(m)$ time per iteration, as the eigenvector changes from iteration to iteration. One way of getting around this problem is to maintain an approximation to $\mathbf{B}_G v_{\min}$ explicitly, but this would require some type of stability on $v_{\min}$ that allows us to either sparsely update our approximation v in each iteration, or only update our entire approximation infrequently.

2.2 Heavy Hitters on Balanced Lossy Expanders

In this section we show how to overcome the obstacles mentioned above to solve the heavy hitter problem on balanced lossy expanders.

Approximation Requirement of v. We first address the accuracy required in our approximation v of the smallest eigenvector $v_{\min}$ for our generalization of the algorithm in [vdBLN+20] to work. Recall that this algorithm finds the heavy hitters of $\mathbf{B}_G h^{\perp v}$ and $\mathbf{B}_G h^v$ for $h^v = vv^\top h$ and $h^{\perp v} = (\mathbf{I} - vv^\top)h$. This algorithm is able to correctly output all heavy hitters of $\mathbf{B}_G h$, because if $|(\mathbf{B}_G h)_i| \geq \epsilon$, we must have that either $|(\mathbf{B}_G h^{\perp v})_i| \geq \epsilon/2$ or $|(\mathbf{B}_G h^v)_i| \geq \epsilon/2$. So we focus on bounding the time complexity of this algorithm.

When finding the heavy hitters of $\mathbf{B}_G h^{\perp v}$, by a similar argument as before, checking all neighbours of each vertex i where $h_i^{\perp v} \geq \epsilon/2$ requires $O((h^{\perp v})^\top \mathbf{D} h^{\perp v} \epsilon^{-2})$ time. We therefore seek a v for which $O((h^{\perp v})^\top \mathbf{D} h^{\perp v} \epsilon^{-2})$ is bounded by $\widetilde{O}(\|\mathbf{B}_G h\|_2^2 \epsilon^{-2})$, the budget that we have for one heavy hitter query. In particular, we would like to find a unit vector v satisfying

$$(h^{\perp v})^\top \mathbf{D} h^{\perp v} \leq \text{polylog}(n) \cdot h^\top \mathbf{L}_G h \text{ for all } h,$$

or equivalently that

$$\mathbf{D} - dvv^\top \preceq \text{polylog}(n) \cdot \mathbf{L}_G. \tag{4}$$

When $\lambda_2(\mathbf{L}_G) \geq \frac{d}{\text{polylog}(n)}$, we have that $v = v_{\min}(\mathbf{L}_G)$ satisfies the requirement in Equation (4). Our next spectral result shows that perhaps surprisingly, Equation (4) holds whenever v has a reasonably small Rayleigh quotient $v^\top \mathbf{L}_G v \leq \text{polylog}(n) \cdot \lambda_{\min}(\mathbf{L}_G)$.

Theorem 2.3 (Spectral approximation from low Rayleigh quotient, informal version of Theorem 4.1)**.** *If G is a balanced lossy expander and unit vector v satisfies $v^\top \mathbf{L}_G v \leq \text{polylog}(n) \cdot \lambda_{\min}(\mathbf{L}_G)$, then*

$$\mathbf{L}_G \approx_{\text{polylog}(n)} \mathbf{D} - (1-\lambda) dvv^\top \text{ for } \lambda = v^\top \mathbf{L}_G v\,.$$

Theorem 2.3 shows that in the heavy hitter algorithm it suffices to use a unit vector v whose Rayleigh quotient $v^\top \mathbf{L}_G v$ is within a polylog(n) factor from the least eigenvalue $\lambda_{\min}$. Another equivalent statement of this requirement is that v needs to be very aligned with $v_{\min}$, more precisely, $1 - (v^\top v_{\min})^2 \leq \text{polylog}(n) \cdot \lambda_{\min}$. The relationship of these two requirements are shown in Theorem 4.8 and Theorem A.1.

Theorem 2.3 suggests a natural approach to maintain the approximate eigenvector v: simply use the same v until the Rayleigh quotient $v^\top \mathbf{L}_G v$ is no longer smaller than $\text{polylog}(n) \cdot \lambda_{\min}$. Note that in this approach it is also easy to find the heavy hitters in $\mathbf{B}_G h^v$ because we can explicitly maintain $\mathbf{B}_G v$ together with v. The number of heavy hitters in $\mathbf{B}_G h^v$, which also determines the running time of this step, is $\|\mathbf{B}_G h^v\|_2^2 \epsilon^{-2}$, and it can be bounded by our budget $\widetilde{O}(\|\mathbf{B}_G h\|_2^2 \epsilon^{-2})$ using the triangle inequality $\|\mathbf{B}_G h^v\|_2^2 \leq O(\|\mathbf{B}_G h\|_2^2 + \|\mathbf{B}_G h^{\perp v}\|_2^2)$, and Theorem 2.3 that $\|\mathbf{B}_G h^{\perp v}\|_2^2 \leq \widetilde{O}\big((h^{\perp v})^\top (\mathbf{D} - (1-\lambda) dvv^\top) h^{\perp v}\big) = \widetilde{O}\big(h^\top (\mathbf{D} - dvv^\top) h\big) \leq \widetilde{O}(\|\mathbf{B}_G h\|_2^2)$.

Structure of Eigenvector $v_{\min}$. To effectively follow this approach, we need to handle that whenever a sufficient number of vertex deletions occur our algorithms need to perform a type of re-normalization. This re-normalization can potentially increase the Rayleigh quotient $v^\top \mathbf{L}_G v$ for the re-normalized approximate smallest eigenvector v. We show however that this increase is fairly limited and to do this we prove another another structural result about $v_{\min}$ of potential independent interest. In particular, we show that $v_{\min}$ of a balanced lossy expanders is coordinate-wise close to the all-ones vector.

Theorem 2.4 (Uniformity of $v_{\min}$, informal version of Theorem 5.1)**.** *If G is a balanced lossy expander, then*

$$\frac{1}{\sqrt{n}} \cdot (1 - 1/\text{polylog}(n)) \leq (v_{\min}(\mathbf{L}_G))_i \leq \frac{1}{\sqrt{n}} \cdot (1 + 1/\text{polylog}(n)) \text{ for all } i \in [n]\,.$$

Theorem 2.4 shows that the least eigenvector $v_{\min}$ of a balanced lossy expander behaves similarly to that of a graph, which is always the all-ones vector. One might wonder if this coordinate-wise approximation to $\frac{\mathbb{1}_V}{\sqrt{n}}$ would allow us to apply the same heavy hitter algorithm used for graphs to balanced lossy expanders. Unfortunately, this is not the case; Theorem 2.4 only guarantees that $1 - (v_{\min}^\top \frac{\mathbb{1}_V}{\sqrt{n}})^2 \leq 1/\text{polylog}(n)$, whereas our algorithm requires an approximate vector v satisfying $1 - (v_{\min}^\top v)^2 \leq \text{polylog}(n) \cdot \lambda_{\min}$ (as in Theorem 2.3).

Heavy Hitters on Balanced Lossy Expanders. We now have all the spectral results needed to design and analyze our heavy hitter algorithm on balanced lossy expanders. Our algorithm uses a reweighting technique together with dynamic expander decomposition to maintain balanced lossy expanders as the graph evolves. This decomposition algorithm routinely deletes edges and vertices, and our goal is to support heavy hitter queries of $\mathbf{B}_G$ under such deletions. Our heavy hitter algorithm maintains an approximate eigenvector $v \in \mathbb{R}^n$ of the Laplacian $\mathbf{L}_G$ throughout all updates, and we begin by explaining how to maintain this vector v.

Instead of finding all heavy entries of $\mathbf{B}_G h$, we introduce a modified problem $\mathbf{B}'$ to ensure that the smallest eigenvalue of the associated Laplacian $\mathbf{L}' \stackrel{\text{def}}{=} \mathbf{B}'^\top \mathbf{B}'$ is bounded away from zero. Concretely, we define

$$\mathbf{B}' \stackrel{\text{def}}{=} \begin{bmatrix} \mathbf{B}_G \\ \epsilon_{\text{add}} \mathbf{I} \end{bmatrix}, \text{ where } \epsilon_{\text{add}} \stackrel{\text{def}}{=} \text{poly}(\delta/(mW)),$$

and consider the heavy hitter problem of $\mathbf{B}' h$. The key advantage of this modified problem is that the modified Laplacian $\mathbf{L}'$ now satisfies $\lambda_{\min}(\mathbf{L}') \geq \epsilon_{\text{add}}$. This lower bound guarantees that $\lambda_{\min}(\mathbf{L}')$ can decrease by a constant factor for at most $O(\log(\epsilon_{\text{add}}^{-1}))$ times, which is crucial for bounding the number of times to recompute the approximate eigenvector v. Since ϵ_{add} is chosen sufficiently small, replacing $\mathbf{B}_G$ with $\mathbf{B}'$ only adds an additional $\delta/10$ error to the final LP solution.

Our algorithm only recomputes v when the condition, $v^\top \mathbf{L}' v \leq \text{polylog}(n) \cdot \lambda_{\min}(\mathbf{L}')$ from Theorem 2.3, no longer holds. Let $\mathbf{L}'^{\text{old}}$ denote the Laplacian at the last time v was updated, where $v = v_{\min}(\mathbf{L}'^{\text{old}})$, and note that $v^\top \mathbf{L}'^{\text{old}} v = \lambda_{\min}(\mathbf{L}'^{\text{old}})$. After each edge deletion, the Rayleigh quotient can only decrease. After each vertex deletion, our algorithm restricts v onto the new vertex support, and re-normalizes it so that it remains a unit vector. This re-normalization does not increase the Rayleigh quotient by more than a constant factor as a result of Theorem 5.1. Our algorithm restarts whenever more than half of the edges are deleted, and Theorem 5.1 ensures that v is close to the all-ones vector, so no set of deleted vertices can be too large in v, i.e., $\sum_{i \in S} v_i^2$ of the deleted vertices S is bounded by a constant. More formally, we show that $v^\top \mathbf{L}' v \leq 3 \cdot v^\top \mathbf{L}'^{\text{old}} v$.

Therefore, whenever the algorithm recomputes v, we must have

$$\lambda_{\min}(\mathbf{L}') < \frac{1}{\text{polylog}(n)} \cdot v^\top \mathbf{L}' v \leq \frac{3}{\text{polylog}(n)} \cdot v^\top \mathbf{L}'^{\text{old}} v = \frac{3}{\text{polylog}(n)} \cdot \lambda_{\min}(\mathbf{L}'^{\text{old}}).$$

This inequality can occur at most $O(\log(\epsilon_{\text{add}}^{-1}))$ times since $\lambda_{\min}(\mathbf{L}') \geq \epsilon_{\text{add}}$.

One last remaining issue is that it is computationally expensive to directly compute the least eigenvalue in order to check if $v^\top \mathbf{L}' v \leq \text{polylog}(n) \cdot \lambda_{\min}(\mathbf{L}')$ still holds. We instead test whether the norm $\|\mathbf{D}' h^{\perp v}\|_2$ exceeds our budget $\|\mathbf{B}' h^{\perp v}\|_2$ by using a Johnson-Lindenstrauss sketch $\mathbf{JB}'$. By Theorem 2.3, if $\|\mathbf{D}' h^{\perp v}\|_2 > \text{polylog}(n) \cdot \|\mathbf{B}' h^{\perp v}\|_2$, then the spectral approximation $\mathbf{L}_G \approx_{\text{polylog}(n)} \mathbf{D} - (1-\lambda) dvv^\top$ no longer holds, and so the Rayleigh quotient must have decreased significantly.

Having established how to maintain the approximate eigenvector v, next we explain how to use it in the heavy hitter queries. Given a query vector h, we decompose it as $h^v = vv^\top h$ and $h^{\perp v} = (\mathbf{I} - vv^\top)h$, and compute the heavy hitters of both $\mathbf{B}_G h^{\perp v}$ and $\mathbf{B}_G h^v$.

- For $\mathbf{B}_G h^{\perp v}$, we check all neighbours of each vertex i where $h_i^{\perp v} \geq \epsilon/2$, and this step takes $O(\|\mathbf{D} h^{\perp v}\|_2^2 \epsilon^{-2}) \leq \widetilde{O}(\|\mathbf{B}_G h\|_2^2 \epsilon^{-2})$ time.
- For $\mathbf{B}_G h^v$, we compute $\mathbf{B}_G v$ explicitly whenever v is updated and use this precomputed vector to find the heavy entries of $\mathbf{B}_G h^v$. This step takes $O(\|\mathbf{B}_G h^v\|_2^2 \epsilon^{-2}) \leq \widetilde{O}(\|\mathbf{B}_G h\|_2^2 \epsilon^{-2})$ time.

This covers the central ideas of our heavy hitter data structure for balanced lossy expanders. Our spectral results and full algorithm provided in Section 5, 4, and 6 are slightly more complex in how they handle arbitrary degrees, and work with the normalized Laplacian instead of the Laplacian.

2.3 Heavy Hitters on Two-Sparse Matrices

In this section, we discuss in greater detail how to build upon our heavy hitter data structure for balanced lossy Laplacians to obtain a heavy hitter data structure for general two-sparse matrices, using a sequence of reductions shown in Figure 1.

We first reduce the problem of heavy hitters on general two-sparse matrices to the problem of heavy hitters on two-sparse matrices with a positive and negative entry in each row, i.e., incidence matrix of general weighted lossy graphs. This reduction is standard and is similar to the one in [Hoc04]; the formal result is given in Theorem 7.8.

Next we reduce the heavy hitter problem on general weighted lossy graphs to that on balanced lossy graphs, which are unweighted lossy graphs with bounded flow multipliers (Section 7.2). To do this, we first partition the edges of the weighted lossy graph so that, in each subgraph, all edge weights are within a constant factor of each other. Each subgraph can then be treated as an unweighted lossy graph, with a single scalar scaling factor applied later. We further decompose each unweighted subgraph into bipartite components with all edges oriented from left to right, bucket the edges based on their flow multipliers, and rescale the vertices so that all flow multipliers lie in the range $\mathbb{1}_E \leq \eta \leq (1+\beta)\mathbb{1}_E$ for a sufficiently small β.

Finally, through the use of the dynamic expander maintenance data structures in [SW19], we reduce the heavy hitter problem on balanced lossy graphs, to the heavy hitter problem on balanced lossy expanders (Section 7.1). The heavy hitter algorithm on balanced lossy expanders requires that the vertex degrees remain approximately regular. To maintain this property, we delete vertices once their degrees fall below a polylog(n) fraction of their initial degrees. We show that this degree preservation process, combined with the expander pruning, results in at most polylog(n) overhead.

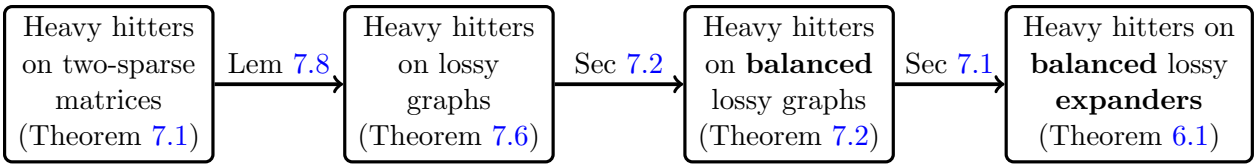

Figure 1: An illustration of the chain of reductions for the heavy hitter data structures.

2.4 Other Data Structures

We conclude the overview by describing how we implement the other two data structures, i.e., the sampler and inverse maintenance, required by the IPM framework of [vdBLL+21]. (The formal definition of inverse maintenance and our result can be found in Theorem 8.6 and Theorem 8.7.)

Sampler. This data structure for a two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ is required to support, for any query vector h, sampling each $e \in [m]$ independently with probability

$$p_e \geq \frac{m}{\sqrt{n}} \frac{(\mathbf{A}h)_e^2}{\|\mathbf{A}h\|_2^2}$$

in $\widetilde{O}(\frac{m}{\sqrt{n}}) \cdot \text{polylog}(\frac{W}{\delta})$ time. Similar to prior results [vdBLN+20, vdBLL+21], we implement the sampler using the same approach as the heavy hitter data structure. We support the sampling operation using a similar chain of reductions as the heavy hitter operation, so that it reduces to sampling an edge of a balanced lossy expander G with probability $p'_e \geq C \cdot (\mathbf{B}_G h)_e^2$, and estimating the ℓ_2 norm $\|\mathbf{B}_G h\|_2$. Using the approximate eigenvector v that is maintained as in Section 2.2 that satisfies $(\mathbf{D} - (1-\lambda)vv^\top) \approx_{\text{polylog}(n)} \mathbf{L}_G$, we again decompose the query vector $h = h^v + h^{\perp v}$ where h^v is in the direction of v, and $h^{\perp v}$ is orthogonal to v.

To sample an edge $e = (i, j)$, our spectral approximation results ensures that

$$(\mathbf{B}_G h)_e^2 \approx (\mathbf{B}_G h^v)_e^2 + \frac{(h_i^{\perp v})^2}{d_i} + \frac{(h_j^{\perp v})^2}{d_j}.$$

Since the algorithm already maintains $\mathbf{B}_G v$, we can sample from the first term $(\mathbf{B}_G h^v)_e^2$ efficiently. For the second term $\frac{(h_i^{\perp v})^2}{d_i} + \frac{(h_j^{\perp v})^2}{d_j}$, we sample uniformly each incident edge of every vertex i with probability $\frac{(h_i^{\perp v})^2}{d_i}$. The total time is proportional to the expected number of edges sampled.

To estimate the ℓ_2 norm, the spectral approximation also implies that

$$\|\mathbf{B}_G h\|_2^2 \approx \|\mathbf{B}_G h^v\|_2^2 + (h^{\perp v})^\top \mathbf{D} h^{\perp v}.$$

We can compute the first term using the maintained value of $\|\mathbf{B}_G v\|_2^2$, and compute the second term in $O(n)$ time.

The formal definition of the sampler and our result regarding an efficient sampler can be found in Theorem 8.4 and Theorem 8.5.

Inverse Maintenance. This data structure supports approximately solving a linear system with $\mathbf{A}^\top \mathbf{V} \mathbf{A}$ in $\widetilde{O}(n) \cdot \text{polylog}(\frac{W}{\delta})$ time. We follow the same idea as [vdBLL+21] that uses leverage score sampling to maintain a $\widetilde{O}(n)$-sparse diagonal matrix $\mathbf{S}$ such that $\mathbf{A}^\top \mathbf{S} \mathbf{A} \approx \mathbf{A}^\top \mathbf{V} \mathbf{A}$. We then use the M-matrix solver of [DS08] (see Theorem 8.9) to solve this lossy Laplacian system in $\widetilde{O}(n) \cdot \text{polylog}(\frac{W}{\delta})$ time.

3 Preliminaries

Vectors. We use $\vec{1}_i$ to denote the i-th standard unit vector, and we use $\mathbb{1}_n$ and $\mathbb{0}_n$ to denote all-ones and all-zeros vectors in $\mathbb{R}^n$. When the dimension of the vector is clear from context, omit the subscript.

Given any $a, b \in \mathbb{R}^n$, we use $a \cdot b$ to denote the vector obtained from coordinate wise multiplication. Similarly, we also use other scalar operations to denote the corresponding entrywise vector operations when clear from context.

Matrices. We use $\mathbf{0}_{m,n}$ to denote the all-zeros matrix of dimension $m \times n$, and we use $\mathbf{I}_n$ to denote the identity matrix of dimension $n \times n$. When the dimension of the matrix is clear from context, we also use $\mathbf{0}$ and $\mathbf{I}$ without subscripts to denote all-zeros and identity matrices.

For any $\mathbf{A} \in \mathbb{R}^{m \times n}$, $I \subseteq [m]$, and $J \subseteq [n]$, we use $\mathbf{A}_{I,J}$ to denote the submatrix of $\mathbf{A}$ with rows in I and columns in J, and we use $\mathbf{A}_{I,:}$ to denote the submatrix of $\mathbf{A}$ with rows in I, and $\mathbf{A}_{:,J}$ to denote the submatrix of $\mathbf{A}$ with columns in J.

We use $\mathbf{M} \succeq \mathbf{0}$ to denote that the matrix $\mathbf{M}$ is positive semidefinite (PSD). We use $\mathbf{A} \succeq \mathbf{B}$ to denote $\mathbf{A} - \mathbf{B} \succeq \mathbf{0}$. We use $\mathbf{A} \approx_c \mathbf{B}$ to denote that $c^{-1}\mathbf{B} \preceq \mathbf{A} \preceq c\mathbf{B}$. For any PSD matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ we let $\lambda_1(\mathbf{M}) \leq \ldots \leq \lambda_n(\mathbf{M})$ denote the eigenvalues of $\mathbf{M}$. Let $v_1(\mathbf{M}), \ldots, v_n(\mathbf{M})$ denote a corresponding orthonormal basis of eigenvectors, and let $\lambda_{\min}(\mathbf{M}) \stackrel{\text{def}}{=} \lambda_1(\mathbf{M})$ and $\lambda_{\max}(\mathbf{M}) \stackrel{\text{def}}{=} \lambda_n(\mathbf{M})$. When $\lambda_{\min}(\mathbf{M})$ is a simple eigenvalue, we also define $v_{\min}(\mathbf{M})$ to be the unit eigenvector corresponding to $\lambda_{\min}(\mathbf{M})$ such that its first non-zero coordinate is positive. When the matrix is clear from context, we also use the shorthands $\lambda_{\min}$, $\lambda_{\max}$, and $v_{\min}$.

For any $\mathbf{M} \in \mathbb{R}^{n \times n}$, we use $\mathbf{diag}(\mathbf{M})$ to denote the diagonal matrix consisting of the diagonal entries of $\mathbf{M}$, and for any vector $v \in \mathbb{R}^n$, we also use $\mathbf{diag}(v)$ to denote the diagonal matrix with v on the diagonal. Additionally, for any vector denoted in lowercase, we also refer to the diagonal matrix with the vector being on the diagonal with the same letter in uppercase, e.g., $\mathbf{D} = \mathbf{diag}(d)$.

For any non-degenerate (i.e., full column rank) matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, we denote the leverage scores of $\mathbf{A}$ as $\sigma(\mathbf{A}) \in \mathbb{R}^m$ where $\sigma(\mathbf{A})_i = (\mathbf{A}(\mathbf{A}^\top \mathbf{A})^{-1}\mathbf{A}^\top)_{i,i}$.

Norms. For any vector $w \in \mathbb{R}^n_{>0}$, we define $\|v\|_w \stackrel{\text{def}}{=} (\sum_{i=1}^n w_i v_i^2)^{1/2}$. For any norm $\|\cdot\|$, we define its induced matrix norm as $\|\mathbf{M}\| \stackrel{\text{def}}{=} \sup_{\|x\|=1} \|\mathbf{M}x\|$.

Sets. We let $[n] \stackrel{\text{def}}{=} \{1, 2, \cdots, n\}$. For any $v \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$, we define $S_{\geq \alpha}(v) \stackrel{\text{def}}{=} \{i \in [n] \mid v_i \geq \alpha\}$ and $S_{\leq \alpha}(v) \stackrel{\text{def}}{=} \{i \in [n] \mid v_i \leq \alpha\}$.

Graphs. In this paper we consider undirected graphs that allow multi-edges but no self-loops, and we assign an orientation to each edge. For any graph G, we use $V(G)$ and $E(G)$ to denote its vertices and edges. For any $V_1, V_2 \subseteq V(G)$, we use $E_G(V_1, V_2)$ to denote the set of edges between vertices in V_1 and vertices in V_2 (in either direction). With an abuse of notation, for any vertex $v \in V(G)$, we also denote $E_G(v, V_1) = E_G(\{v\}, V_1)$. For any vertex $v \in V(G)$, we use $\mathcal{N}_G(v)$ to denote the neighboring vertices of v in G. We omit the subscript of $E_G(\cdot)$ and $\mathcal{N}_G(\cdot)$ when the graph is clear from context.

For any weighted graph $G = (V, E, w)$, for any $E' \subseteq E$, we define $w(E') \stackrel{\text{def}}{=} \sum_{e \in E'} w_e$, and for any vertex $v \in V$, we define its weighted degree $\deg_w(v)$ as the sum of the weights of all edges incident to v, and for any $S \subseteq V$, we define $\text{vol}_G(S) \stackrel{\text{def}}{=} \sum_{v \in S} \deg_w(v)$. We omit the subscript of $\text{vol}_G(\cdot)$ when the graph and its weights are clear from context. We define $\mathbf{B}_G \in \mathbb{R}^{E \times V}$ as the

incidence matrix of G where row e of $\mathbf{B}_G$ is $\vec{1}_{b_e} - \vec{1}_{a_e}$. We define the (weighted) Laplacian of G as $\mathbf{L}_G \stackrel{\text{def}}{=} \mathbf{B}_G^\top \mathbf{W} \mathbf{B}_G$, and let $\mathbf{D}_G \stackrel{\text{def}}{=} \mathbf{diag}(\mathbf{L}_G)$ denote its diagonal. We define the normalized Laplacian of G as $\mathbf{N}_G \stackrel{\text{def}}{=} \mathbf{D}_G^{-1/2} \mathbf{L}_G \mathbf{D}_G^{-1/2}$, and we let $d_G \in \mathbb{R}^V$ denote the vector of diagonal entries of $\mathbf{D}_G$. We define the conductance $\phi(G)$ as follows:

$$\phi(G) \stackrel{\text{def}}{=} \min_{S \subseteq V} \frac{w(E_G(S, V \backslash S))}{\min\{\text{vol}_G(S), \text{vol}_G(V \backslash S)\}}.$$

We say a graph G has expansion ϕ if $\phi(G) \geq \phi$, and we also call such G a *ϕ-expander*.

Lossy Flow and Lossy Graph Matrices. We call $G = (V, E, \eta, w)$ a *weighted lossy flow graph* if E is a multiset of pairs of distinct vertices in V, i.e., for each $e \in E$, $e = (a_e, b_e)$ with $a_e, b_e \in V$ and $a_e \neq b_e$, $\eta \in \mathbb{R}^E_{\geq 1}$ are the flow multipliers, and $w \in \mathbb{R}^E_{\geq 0}$ are the edge weights.

We define $V(G)$, $E(G)$, and $E_G(V_1, V_2)$ similarly as for graphs. We define $\mathbf{B}_G \in \mathbb{R}^{E \times V}$ as the incidence matrix of G where row e of $\mathbf{B}_G$ is $\vec{1}_{b_e} - \eta_e \vec{1}_{a_e}$.

We define the (weighted) Laplacian of G as $\mathbf{L}_G \stackrel{\text{def}}{=} \mathbf{B}_G^\top \mathbf{W} \mathbf{B}_G$, and let $\mathbf{D}_G \stackrel{\text{def}}{=} \mathbf{diag}(\mathbf{L}_G)$ denote its diagonal. We also refer to the Laplacian of a lossy graph as a lossy Laplacian, to distinguish it from the standard graph Laplacian. We further define the normalized Laplacian of G as $\mathbf{N}_G \stackrel{\text{def}}{=} \mathbf{D}_G^{-1/2} \mathbf{L}_G \mathbf{D}_G^{-1/2}$. We also let $d_G \in \mathbb{R}^V$ denote the vector of diagonal entries of $\mathbf{D}_G$. We say that the graph is unweighted when w is all one, and with an abuse of notation we use the shorthand $G = (V, E, \eta)$ to denote $G = (V, E, \eta, \mathbb{1}_m)$.

We define $\overline{G} = (V, E, w)$ as the *smoothed graph* associated with $G = (V, E, \eta, w)$, where $\overline{G}$ has the same set of edges as G and its edges are non-lossy. We say a lossy graph G is connected if $\overline{G}$ is connected. We call a lossy graph $G = (V, E, \eta, w)$ *β_η-balanced* if $\eta_e \leq 1 + \beta_\eta$ for all $e \in E$. We say a lossy graph G has expansion ϕ, or equivalently we call G a *ϕ-expander*, if the underlying smoothed graph $\overline{G}$ has conductance $\phi(\overline{G}) \geq \phi$.

Data Structure Guarantees. In all data structures provided in this paper, amortized bounds are taken over the entire sequence of all operations, unless an operation is explicitly stated to have a worst-case time bound. Only limited effort is made to optimize the polylogarithmic terms throughout this paper.

4 Spectral Approximation of Lossy Graphs

In this section we first prove Theorem 4.1, the formal version of Theorem 2.3. Towards the end of this section, we also use the spectral results developed to prove Theorem 4.1 to show that the standard power iteration can be used to approximately compute $v_{\min}$ of the lossy Laplacian.

Theorem 4.1 shows that in order to obtain a constant spectral approximation of the form $\mathbf{I} - (1 - \lambda) v v^\top$ to $\mathbf{N}_G$, we only require $v^\top \mathbf{N}_G v \leq c \lambda_1(\mathbf{N}_G)$, for some constant c. To facilitate our later algorithmic development, we prove a more general version of this fact that allows more flexibility in the normalizing diagonal $\mathbf{D}$, as well as some additive error, $\epsilon_{\text{add}}\mathbf{I}$,.

Theorem 4.1 (Spectral approximation from low Rayleigh quotient)**.** *Let $c_1, c_2 \geq 1$, let $0 \leq \epsilon_{\text{add}} \leq 1$, let $G = (V, E, \eta, w)$ be a β_η-balanced lossy flow graph with $\lambda_2(\mathbf{N}_{\overline{G}}) \geq 20\beta_\eta$, let $d \in \mathbb{R}^V_{\geq 0}$ be a vector that satisfies $d_G \leq d \leq c_1 d_G$. Furthermore, let $v \in \mathbb{R}^V$ be a unit vector such that $v^\top (\mathbf{D}^{-1/2} \mathbf{L}_G \mathbf{D}^{-1/2} + \epsilon_{\text{add}} \mathbf{I}) v \leq c_2 \lambda_1(\mathbf{D}^{-1/2} \mathbf{L}_G \mathbf{D}^{-1/2} + \epsilon_{\text{add}} \mathbf{I})$ where $\mathbf{D} = \mathbf{diag}(d)$. Then, for*

$\lambda \stackrel{\text{def}}{=} v^\top(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})v$,

$$\frac{(\lambda_2(\mathbf{N}_{\overline{G}}))^2}{12c_1^2c_2^2}\left(\mathbf{I} - (1-\lambda)vv^\top\right) \preceq \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I} \preceq \frac{24c_1c_2}{\lambda_2(\mathbf{N}_{\overline{G}})}\left(\mathbf{I} - (1-\lambda)vv^\top\right).$$

Applying Theorem 4.1 with $\mathbf{D} = \mathbf{D}_G$, and $\epsilon_{\text{add}} = 0$ immediately yields the following corollary:

Corollary 4.2. *Let $G = (V, E, \eta, w)$ be a β_η-balanced lossy flow graph with $\lambda_2(\mathbf{N}_{\overline{G}}) \geq 20\beta_\eta$. Let v be a unit vector such that $v^\top\mathbf{N}_G v \leq c\lambda_1(\mathbf{N}_G)$ for some $c \geq 1$. Then for $\lambda \stackrel{\text{def}}{=} v^\top\mathbf{N}_G v$,*

$$\frac{(\lambda_2(\mathbf{N}_{\overline{G}}))^2}{12c^2}\left(\mathbf{I} - (1-\lambda)vv^\top\right) \preceq \mathbf{N}_G \preceq \frac{24c}{\lambda_2(\mathbf{N}_{\overline{G}})}\left(\mathbf{I} - (1-\lambda)vv^\top\right).$$

Before proving Theorem 4.1, we prove some structural lemmas. First, we show that the normalized graph Laplacian and a sufficiently balanced normalized lossy Laplacian are close spectrally.

Lemma 4.3. *If $G = (V, E, \eta, w)$* is a β_η-balanced lossy flow graph *for $\beta_\eta \leq 1$, then,*

$$\mathbf{D}_{\overline{G}} \preceq \mathbf{D}_G \preceq (1 + 3\beta_\eta)\mathbf{D}_{\overline{G}} \text{ and } \|\mathbf{N}_G - \mathbf{N}_{\overline{G}}\|_2 \leq 10\beta_\eta\,.$$

Proof. Recall that each edge $e \in E$ is denoted as (a_e, b_e), and row e of $\mathbf{B}_G$ and $\mathbf{B}_{\overline{G}}$ are $\vec{1}_{b_e} - \eta_e\vec{1}_{a_e}$ and $\vec{1}_{b_e} - \vec{1}_{a_e}$ respectively. We have that

$$\begin{aligned}(\mathbf{B}_G)_{e,:}^\top(\mathbf{B}_G)_{e,:} - (\mathbf{B}_{\overline{G}})_{e,:}^\top(\mathbf{B}_{\overline{G}})_{e,:} &= (\vec{1}_{b_e} - \eta_e\vec{1}_{a_e})(\vec{1}_{b_e} - \eta_e\vec{1}_{a_e})^\top - (\vec{1}_{b_e} - \vec{1}_{a_e})(\vec{1}_{b_e} - \vec{1}_{a_e})^\top \\ &= (\eta_e^2 - 1)\vec{1}_{a_e}\vec{1}_{a_e}^\top - (\eta_e - 1)\left(\vec{1}_{a_e}\vec{1}_{b_e}^\top + \vec{1}_{b_e}\vec{1}_{a_e}^\top\right).\end{aligned}$$

Using this equation, and by the definition of the Laplacian,

$$\mathbf{L}_G - \mathbf{L}_{\overline{G}} = \mathbf{B}_G^\top\mathbf{W}\mathbf{B}_G - \mathbf{B}_{\overline{G}}^\top\mathbf{W}\mathbf{B}_{\overline{G}} = \sum_{e\in E} w_e\left[(\eta_e^2 - 1)\vec{1}_{a_e}\vec{1}_{a_e}^\top - (\eta_e - 1)\left(\vec{1}_{a_e}\vec{1}_{b_e}^\top + \vec{1}_{b_e}\vec{1}_{a_e}^\top\right)\right].$$

Let $\mathbf{D}^\Delta \stackrel{\text{def}}{=} \mathbf{diag}(\mathbf{L}_G - \mathbf{L}_{\overline{G}})$ and let $\mathbf{A}^\Delta \stackrel{\text{def}}{=} \mathbf{L}_{\overline{G}} - \mathbf{L}_G + \mathbf{D}^\Delta$. Recall that $\eta_e \geq 1$, $\beta_\eta \leq 1$, and $\eta_e \leq 1 + \beta_\eta$. Since $\eta_e^2 - 1 \leq 3(\eta_e - 1) \leq 3\beta_\eta$, we have that entrywise

$$\begin{aligned}\mathbf{0} \leq \mathbf{D}^\Delta &= \sum_{e\in E} w_e(\eta_e^2 - 1)\vec{1}_{a_e}\vec{1}_{a_e}^\top \leq \sum_{e\in E} 3\beta_\eta w_e\vec{1}_{a_e}\vec{1}_{a_e}^\top \leq 3\beta_\eta\mathbf{D}_G \text{ and} \\ \mathbf{0} \leq \mathbf{A}^\Delta &= \sum_{e\in E} w_e(\eta_e - 1)\left(\vec{1}_{a_e}\vec{1}_{b_e}^\top + \vec{1}_{b_e}\vec{1}_{a_e}^\top\right) \leq \sum_{e\in E}\beta_\eta w_e\left(\vec{1}_{a_e}\vec{1}_{b_e}^\top + \vec{1}_{b_e}\vec{1}_{a_e}^\top\right).\end{aligned}$$

Since the entries of $\mathbf{D}_G^{-1/2}\mathbf{D}^\Delta\mathbf{D}_G^{-1/2}$ are non-negative and the matrix is diagonal,

$$\|\mathbf{D}_G^{-1/2}\mathbf{D}^\Delta\mathbf{D}_G^{-1/2}\|_2 \leq \|\mathbf{D}_G^{-1/2}\left(3\beta_\eta\mathbf{D}_G\right)\mathbf{D}_G^{-1/2}\|_2 \leq 3\beta_\eta\,.$$

This proves the first claim that $\mathbf{D}_{\overline{G}} \preceq \mathbf{D}_G \preceq (1 + 3\beta_\eta)\mathbf{D}_{\overline{G}}$.

Similarly, since the entries of $\mathbf{D}_G^{-1/2}\mathbf{A}^\Delta\mathbf{D}_G^{-1/2}$ are non-negative, by the Perron–Frobenius theorem the largest eigenvalue of this matrix is equals to its spectral radius, and the corresponding eigenvector is non-negative, so

$$\|\mathbf{D}_G^{-1/2}\mathbf{A}^\Delta\mathbf{D}_G^{-1/2}\|_2 \leq \left\|\mathbf{D}_G^{-1/2}\sum_{e\in E}\beta_\eta w_e\left(\vec{1}_{a_e}\vec{1}_{b_e}^\top + \vec{1}_{b_e}\vec{1}_{a_e}^\top\right)\mathbf{D}_G^{-1/2}\right\|_2 \leq \beta_\eta\,,$$

where in the last step we used that the matrix $\sum_{e\in E} w_e(\vec{1}_{a_e}\vec{1}_{b_e}^\top + \vec{1}_{b_e}\vec{1}_{a_e}^\top)$ is the adjacency matrix of a weighted graph where every vertex $a \in V$ has weighted degree at most $[\mathbf{D}_G]_{aa}$.

Consequently, we have that

$$\|\mathbf{D}_G^{-1/2}(\mathbf{L}_G - \mathbf{L}_{\overline{G}})\mathbf{D}_G^{-1/2}\|_2 = \|\mathbf{D}_G^{-1/2}(\mathbf{D}^\Delta - \mathbf{A}^\Delta)\mathbf{D}_G^{-1/2}\|_2 \le 4\beta_\eta\,.$$

Additionally, note that $1 \le \eta_e^2 \le (1+\beta_\eta)^2 \le 1+3\beta_\eta$, and hence $\mathbf{D}_{\overline{G}} \preceq \mathbf{D}_G \preceq (1+3\beta_\eta)\mathbf{D}_{\overline{G}}$. This then implies that

$$\|\mathbf{I} - \mathbf{D}_G^{-1/2}\mathbf{D}_{\overline{G}}^{1/2}\|_2 \le 1 - \frac{1}{\sqrt{1+3\beta_\eta}} \le 3\beta_\eta.$$

Since

$$\|\mathbf{D}_G^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_{\overline{G}}^{-1/2}\|_2 \le \|\mathbf{D}_{\overline{G}}^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_{\overline{G}}^{-1/2}\|_2 \le 1,$$

we have that:

$$\begin{aligned}&\|\mathbf{D}_{\overline{G}}^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_{\overline{G}}^{-1/2} - \mathbf{D}_G^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_G^{-1/2}\|_2\\ \le\ &\|(\mathbf{I} - \mathbf{D}_G^{-1/2}\mathbf{D}_{\overline{G}}^{1/2})\mathbf{D}_{\overline{G}}^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_{\overline{G}}^{-1/2}\|_2 + \|\mathbf{D}_G^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_{\overline{G}}^{-1/2}(\mathbf{I} - \mathbf{D}_G^{-1/2}\mathbf{D}_{\overline{G}}^{1/2})\|_2 \le 6\beta_\eta,\end{aligned}$$

and hence:

$$\|\mathbf{N}_G - \mathbf{N}_{\overline{G}}\|_2 \le \|\mathbf{D}_G^{-1/2}(\mathbf{L}_G - \mathbf{L}_{\overline{G}})\mathbf{D}_G^{-1/2}\|_2 + \|\mathbf{D}_{\overline{G}}^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_{\overline{G}}^{-1/2} - \mathbf{D}_G^{-1/2}\mathbf{L}_{\overline{G}}\mathbf{D}_G^{-1/2}\|_2 \le 10\beta_\eta. \quad \square$$

A consequence of Theorem 4.3 is that so long as $\lambda_2(\mathbf{N}_{\overline{G}})$ is greater than $20\beta_\eta$, we can obtain a nontrivial lower bound on $\lambda_2(\mathbf{N}_G)$.

Next we prove that given the exact least eigenvalue λ_1 and the corresponding unit eigenvector $v_{\min}$ of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$, the matrix $\mathbf{I} - (1-\lambda_1)v_{\min}v_{\min}^\top$ is a good spectral approximation of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$.

Lemma 4.4. *Let $G = (V, E, \eta)$ be a β_η-balanced lossy flow graph with $20\beta_\eta \le \lambda_2(\mathbf{N}_{\overline{G}}) \le 1$. Let $d \in \mathbb{R}^V_{\ge 0}$ satisfy $d_G \le d \le c \cdot d_G$ for some $c \ge 1$. Let $\lambda_1 = \lambda_1(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})$ for some $0 \le \epsilon_{\text{add}} \le 1$, and let $v_{\min}$ be a corresponding unit eigenvector. Then,*

$$\frac{\lambda_2(\mathbf{N}_{\overline{G}})}{2c}\Big(\mathbf{I} - (1-\lambda_1)v_{\min}v_{\min}^\top\Big) \preceq \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I} \preceq 4\Big(\mathbf{I} - (1-\lambda_1)v_{\min}v_{\min}^\top\Big).$$

In particular, $\lambda_2(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}) \ge \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{2c}$ and $\lambda_n(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}) \le 4$.

To prove this lemma, we use the following fact.

Fact 4.5. *Let $\mathbf{M}, \mathbf{D} \in \mathbb{R}^{n\times n}$ be positive semi-definite matrices. If $\frac{1}{c_1}\mathbf{I} \preceq \mathbf{D} \preceq c_2\mathbf{I}$ for $c_1, c_2 \ge 1$, then $\mathbf{M}' \stackrel{\text{def}}{=} \mathbf{DMD}$ satisfies that*

$$\frac{\lambda_i(\mathbf{M})}{c_1^2} \le \lambda_i(\mathbf{M}') \le c_2^2\lambda_i(\mathbf{M}) \text{ for all } i \in [n]\,.$$

Proof. The Courant-Fischer min-max theorem implies that

$$\begin{aligned}\lambda_i(\mathbf{M}') = \min_{\dim(U)=i}\max_{x\in U}\frac{x^\top\mathbf{M}'x}{x^\top x} &= \min_{\dim(U)=i}\max_{x\in U}\frac{x^\top\mathbf{DMD}x}{x^\top\mathbf{D}^2x}\cdot\frac{x^\top\mathbf{D}^2x}{x^\top x}\\ &\le \min_{\dim(U)=i}\max_{x\in U}\frac{x^\top\mathbf{DMD}x}{x^\top\mathbf{D}^2x}\cdot c_2^2 = \lambda_i(\mathbf{M})\cdot c_2^2.\end{aligned}$$

An analogous argument implies that $\lambda_i(\mathbf{M}') \ge \lambda_i(\mathbf{M})/c_1^2$. $\square$

Proof of Theorem 4.4. Theorem 4.3 implies that $\lambda_2(\mathbf{N}_G) \geq \lambda_2(\mathbf{N}_{\overline{G}}) - 10\beta_\eta \geq \lambda_2(\mathbf{N}_{\overline{G}})/2$. Since $\lambda_n(\mathbf{N}_{\overline{G}}) \leq 2$, we also have that $\lambda_n(\mathbf{N}_G) \leq 2 + 10\beta_\eta \leq 3$. Now, since $\frac{1}{c}\mathbf{I} \preceq \mathbf{D}_G\mathbf{D}^{-1} \preceq \mathbf{I}$, Theorem 4.5 implies that $\lambda_2(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}) \geq \frac{1}{c}\lambda_2(\mathbf{N}_G) \geq \lambda_2(\mathbf{N}_{\overline{G}})/(2c)$, and $\lambda_n(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}) \leq \lambda_n(\mathbf{N}_G) \leq 3$. Finally using that $0 \leq \epsilon_{\text{add}} \leq 1$ we get the inequalities as desired. □

In order to prove Theorem 4.1, we also need to guarantee that there is a gap between the smallest and second smallest eigenvalues of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$. The following lemma shows that when β_η is small this gap is at least $\Omega(\lambda_2(\mathbf{N}_{\overline{G}}))$. Since the rescaling matrix $\mathbf{D}$ is not exactly $\mathbf{D}_G$, the next lemma requires an even smaller β_η than Theorem 4.4 to ensure such a spectral gap.

Lemma 4.6. *Let $G = (V, E, \eta)$ be a β_η-balanced lossy flow graph with $\lambda_2(\mathbf{N}_{\overline{G}}) \leq 1$. Let $d \in \mathbb{R}^V_{\geq 0}$ satisfy $d_G \leq d \leq c \cdot d_G$ for some $c \geq 1$, and let $\mathbf{D} = \mathbf{diag}(d)$. If $\beta_\eta \leq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{40c}$ then,*

$$\lambda_2\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right) - \lambda_1\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right) \geq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{4c}.$$

Proof. $\lambda_2(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}) \geq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{2c}$ by Theorem 4.4 with $\epsilon_{\text{add}} = 0$. Theorem 4.3 also implies that $\lambda_1(\mathbf{N}_G) \leq \lambda_1(\mathbf{N}_{\overline{G}}) + 10\beta_\eta = 10\beta_\eta$. Since Theorem 4.5, $\lambda_1(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}) \leq 10\beta_\eta$. When $\beta_\eta \leq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{40c}$, we have that $\lambda_1(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}) \leq \frac{1}{2}\lambda_2(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$, yielding the desired result. □

Next we show that if two unit vectors v_1 and v_2 are close to each other, i.e., $v_1^\top v_2$ is close to 1, then the two matrices $\mathbf{I} - (1-\lambda)v_1v_1^\top$ and $\mathbf{I} - (1-\lambda)v_2v_2^\top$ are good spectral approximations of each other. This gives a sufficient condition for a vector v to approximate the true eigenvector $v_{\min}$ so that $\mathbf{I} - (1-\lambda)vv^\top \approx \mathbf{I} - (1-\lambda)v_{\min}v_{\min}^\top$. The proof of the following lemma is inspired by the proof of Lemma 27 of [CLM+16].

Lemma 4.7. *Let $v_1 \neq v_2 \in \mathbb{R}^n$ be unit vectors where $1 - (v_1^\top v_2)^2 \leq c\lambda$, for $\lambda \in (0, 1]$ and $c \geq 1$. Additionally, let $\mathbf{M}_i \stackrel{\text{def}}{=} \mathbf{I} - (1-\lambda)v_iv_i^\top$ for $i \in \{1, 2\}$. Then, $\frac{1}{3c}\mathbf{M}_1 \preceq \mathbf{M}_2 \preceq 3c\mathbf{M}_1$.*

Proof. By symmetry, it suffices to show that $\mathbf{M}_2 \preceq (1+2c)\mathbf{M}_1$, which is equivalent to

$$\mathbf{M}_1^{-1/2}(\mathbf{M}_2 - \mathbf{M}_1)\mathbf{M}_1^{-1/2} \preceq 2c\mathbf{I}.$$

We will explicitly compute the eigenvalues of $\mathbf{M}_1^{-1/2}(\mathbf{M}_2 - \mathbf{M}_1)\mathbf{M}_1^{-1/2}$, and show that they are at most $2c$. Note that $\mathbf{M}_1^{-1/2}(\mathbf{M}_2 - \mathbf{M}_1)\mathbf{M}_1^{-1/2}v = \gamma v$ if and only if $x = \mathbf{M}_1^{-1/2}v$ satisfies

$$(\mathbf{M}_2 - \mathbf{M}_1)x = \gamma\mathbf{M}_1x\,.$$

Additionally, since $\mathbf{M}_2 - \mathbf{M}_1 = (1-\lambda)(v_1v_1^\top - v_2v_2^\top)$, we see that γ is a non-zero eigenvalue of $\mathbf{M}_1^{-1/2}(\mathbf{M}_2 - \mathbf{M}_1)\mathbf{M}_1^{-1/2}$ only when x is in the span of $v_{\min}$ and v_2. Let $x = \alpha v_1 + \beta v_2$ such that $(\mathbf{M}_2 - \mathbf{M}_1)x = \gamma\mathbf{M}_1x$. Since

$$(\mathbf{M}_2 - \mathbf{M}_1)x = (1-\lambda)\left[\big(\alpha + \beta(v_1^\top v_2)\big)v_1 - \big(\alpha(v_1^\top v_2) + \beta\big)v_2\right] \text{ and}$$
$$\gamma\mathbf{M}_1x = \gamma\left[\big(\lambda\alpha - \beta(1-\lambda)(v_1^\top v_2)\big)v_1 + \beta v_2\right],$$

we have $(\mathbf{M}_2 - \mathbf{M}_1)x = \gamma\mathbf{M}_1x$ is equivalent to

$$(1-\lambda)\begin{pmatrix} 1 & v_1^\top v_2 \\ -v_1^\top v_2 & -1 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \gamma\begin{pmatrix} \lambda & -(1-\lambda)v_1^\top v_2 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix},$$

or equivalently:

$$\begin{pmatrix} 1-\lambda-\gamma\lambda & (1-\lambda)(1+\gamma)v_1^\top v_2 \\ -(1-\lambda)v_1^\top v_2 & \lambda-1-\gamma \end{pmatrix}\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = 0.$$

Denote the matrix in the last equation above as $\mathbf{N}$. A non-trivial solution $(\alpha,\beta) \neq (0,0)$ exists if and only if $\det(\mathbf{N}) = 0$, and

$$\begin{aligned} 0 = \det(\mathbf{N}) &= (1-\lambda-\gamma\lambda)\cdot(\lambda-1-\gamma) - \big((1-\lambda)(1+\gamma)v_1^\top v_2\big)\cdot\big(-(1-\lambda)v_1^\top v_2\big) \\ &= \lambda\gamma^2 - (1-\lambda)^2\gamma - (1-\lambda)^2 + (1-\lambda)^2(v_1^\top v_2)^2\gamma + (1-\lambda)^2(v_1^\top v_2)^2 \\ &= \lambda\gamma^2 - (1-\lambda)^2\big(1-(v_1^\top v_2)^2\big)\gamma - (1-\lambda)^2\big(1-(v_1^\top v_2)^2\big). \end{aligned}$$

Solving this quadratic equation, the two possible values of γ are:

$$\gamma = \frac{(1-\lambda)^2\big(1-(v_1^\top v_2)^2\big) \pm \sqrt{(1-\lambda)^4\big(1-(v_1^\top v_2)^2\big)^2 + 4\lambda(1-\lambda)^2\big(1-(v_1^\top v_2)^2\big)}}{2\lambda}.$$

Since $\sqrt{a^2+b^2} \leq a+b$ for any $a \geq 0$ and $b \geq 0$, we have that

$$|\gamma| \leq \frac{2(1-\lambda)^2\big(1-(v_1^\top v_2)^2\big) + 2\sqrt{\lambda}(1-\lambda)\sqrt{\big(1-(v_1^\top v_2)^2\big)}}{2\lambda}.$$

Since v_1 and v_2 satisfy $1-(v_1^\top v_2)^2 \leq c\lambda$, we have that

$$|\gamma| \leq c(1-\lambda)^2 + \sqrt{c}(1-\lambda) \leq 2c.$$

Consequently, each eigenvalue of $\mathbf{M}_1^{-1/2}(\mathbf{M}_2-\mathbf{M}_1)\mathbf{M}_1^{-1/2}$ has absolute value at most $2c$, and therefore $\mathbf{M}_2 \preceq (1+2c)\mathbf{M}_1 \preceq 3c\mathbf{M}_1$. By symmetry, $\mathbf{M}_1 \preceq 3c\mathbf{M}_2$ as well, completing the proof. □

Next, in Theorem 4.8 we prove that $1 - v^\top v_{\min}(\mathbf{M})$ is small if the Rayleigh quotient $v^\top\mathbf{M}v$ is small. Combined with Theorem 4.7, Theorem 4.8 shows that a vector v with a small Rayleigh quotient is sufficient to obtain the spectral approximation $\mathbf{M} \approx \mathbf{I} - (1-\lambda)vv^\top$.

Lemma 4.8. $1 - \big(v^\top v_1(\mathbf{M})\big)^2 \leq \frac{v^\top\mathbf{M}v}{\lambda_2(\mathbf{M})}$ *for any PSD* $\mathbf{M} \in \mathbb{R}^{n\times n}$ *and unit* $v \in \mathbb{R}^n$.

Proof. Note that $v = \sum_{i\in[n]}(v_i(\mathbf{M})^\top v)v_i(\mathbf{M})$, and $1 = \|v\|_2^2 = \sum_{i\in[n]}(v_i(\mathbf{M})^\top v)^2$. Consequently,

$$v^\top\mathbf{M}v = \sum_{i=1}^n (v_i(\mathbf{M})^\top v)^2\lambda_i(\mathbf{M}) \geq \sum_{i=2}^n (v_i(\mathbf{M})^\top v)^2\lambda_2(\mathbf{M}) = \Big(1-(v_1(\mathbf{M})^\top v)^2\Big)\lambda_2(\mathbf{M}).$$

Rearranging terms proves this lemma. □

Putting these lemmas together, we now have the tools needed to prove Theorem 4.1.

Proof of Theorem 4.1. In this proof we define $\mathbf{M} \stackrel{\text{def}}{=} \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$, and we denote $v_{\min} \stackrel{\text{def}}{=} v_{\min}(\mathbf{M})$. By Theorem 4.4,

$$\frac{\lambda_2(\mathbf{N}_{\overline{G}})}{2c_1}\Big(\mathbf{I} - (1-\lambda_1(\mathbf{M}))v_{\min}v_{\min}^\top\Big) \preceq \mathbf{M} \preceq 4\Big(\mathbf{I} - (1-\lambda_1(\mathbf{M}))v_{\min}v_{\min}^\top\Big). \tag{5}$$

In particular, the above equation implies that $\lambda_2(\mathbf{M}) \geq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{2c_1}$. Now, applying Theorem 4.8 to $\mathbf{M} = \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$:

$$1 - (v^\top v_{\min})^2 \leq \frac{v^\top \mathbf{M} v}{\lambda_2(\mathbf{M})} \leq \frac{2c_1c_2}{\lambda_2(\mathbf{N}_{\overline{G}})}\lambda_1(\mathbf{M}).$$

Using this upper bound on $1 - (v^\top v_{\min})^2$, we can apply Theorem 4.7 to v,$v_{\min}$ and $\lambda_1(\mathbf{M})$:

$$\frac{\lambda_2(\mathbf{N}_{\overline{G}})}{6c_1c_2}\left(\mathbf{I} - (1-\lambda_1(\mathbf{M}))vv^\top\right) \preceq \mathbf{I} - (1-\lambda_1(\mathbf{M}))v_{\min}v_{\min}^\top \preceq \frac{6c_1c_2}{\lambda_2(\mathbf{N}_{\overline{G}})}\left(\mathbf{I} - (1-\lambda_1(\mathbf{M}))vv^\top\right). \tag{6}$$

Since $\lambda_1(\mathbf{M}) \leq \lambda \leq c_2\lambda_1(\mathbf{M})$,

$$\frac{1}{c_2}\left(\mathbf{I} - (1-\lambda)vv^\top\right) \preceq \mathbf{I} - (1-\lambda_1(\mathbf{M}))vv^\top \preceq \mathbf{I} - (1-\lambda)vv^\top. \tag{7}$$

Combining Eq. (5), (6), and (7) implies the desired bounds of

$$\mathbf{M} \preceq 4\left(\mathbf{I} - (1-\lambda_1(\mathbf{M}))v_{\min}v_{\min}^\top\right) \preceq \frac{24c_1c_2}{\lambda_2(\mathbf{N}_{\overline{G}})}\left(\mathbf{I} - (1-\lambda_1(\mathbf{M}))vv^\top\right) \preceq \frac{24c_1c_2}{\lambda_2(\mathbf{N}_{\overline{G}})}\left(\mathbf{I} - (1-\lambda)vv^\top\right)$$

and

$$\begin{aligned}\mathbf{M} &\succeq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{2c_1}\left(\mathbf{I} - (1-\lambda_1(\mathbf{M}))v_{\min}v_{\min}^\top\right) \\ &\succeq \frac{(\lambda_2(\mathbf{N}_{\overline{G}}))^2}{12c_1^2c_2}\left(\mathbf{I} - (1-\lambda_1(\mathbf{M}))vv^\top\right) \succeq \frac{(\lambda_2(\mathbf{N}_{\overline{G}}))^2}{12c_1^2c_2^2}\left(\mathbf{I} - (1-\lambda)vv^\top\right).\end{aligned}$$ □

Power iteration for lossy Laplacian. Using the spectral results proven in this section, we can also show that the standard power iteration can compute a constant approximation to the least eigenvector of the lossy Laplacian. We will use the following standard power iteration with a random start (see e.g. [KW92]).

Theorem 4.9 (Power Iteration)**.** *There is an algorithm* PowerIteration$(\mathbf{M}, \epsilon)$ *that for any input PSD* $\mathbf{M} \in \mathbb{R}^{n\times n}$*, and a parameter* $\epsilon \in (0,1)$*, returns w.h.p. a unit vector* $v \in \mathbb{R}^n$ *such that* $v^\top \mathbf{M} v \geq (1-\epsilon)\lambda_n(\mathbf{M})$ *in* $O\left(\text{nnz}(\mathbf{M})\log(\frac{n}{\epsilon})\max\{1, (\frac{\lambda_n(\mathbf{M})}{\lambda_{n-1}(\mathbf{M})} - 1)^{-1}\}\right)$ *time.*

Next we apply the spectral results proven in this section to prove the guarantees of PowerIteration when used to compute the least eigenvalue of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$.

Theorem 4.10. *Let* $G = (V, E, \eta)$ *be a* β_η*-balanced lossy flow graph with* $\lambda_2(\mathbf{N}_{\overline{G}}) \leq 1$*, let* $d \in \mathbb{R}^V_{\geq 0}$ *be a vector that satisfies* $d_G \leq d \leq c \cdot d_G$ *for some* $c \geq 1$*, and let* $\mathbf{D} = \mathbf{diag}(d)$*. Furthermore, assume that* $\beta_\eta \leq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{40c}$*. Let the unit vector*

$$v = \text{PowerIteration}\left(4\mathbf{I} - (\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}), \epsilon_{\text{add}}/4\right),$$

where PowerIteration *is defined in Theorem 4.9. Then w.h.p.* v *satisfies*

$$v^\top\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v \leq 2\lambda_1\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right),$$

and can be computed in $O\left(m(\lambda_2(\mathbf{N}_{\overline{G}}))^{-1}c\log(\frac{n}{\epsilon_{\text{add}}})\right)$ *time.*

Proof. In this proof we denote $\mathbf{M} \stackrel{\text{def}}{=} \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$.

Correctness: We first show that $v^\top \mathbf{M} v \leq 2\lambda_1(\mathbf{M})$. Theorem 4.4 implies that $\lambda_n(\mathbf{M}) \leq 4$, and as such $4\mathbf{I} - \mathbf{M}$ is PSD. Let $v_{\min} \stackrel{\text{def}}{=} v_{\min}(\mathbf{M}) = v_n(4\mathbf{I} - \mathbf{M})$. The guarantees of PowerIteration in Theorem 4.9 imply that

$$v^\top(4\mathbf{I} - \mathbf{M})v \geq (1 - \epsilon_{\text{add}}/4)v_{\min}^\top(4\mathbf{I} - \mathbf{M})v_{\min},$$

and re-arranging this inequality yields

$$v^\top \mathbf{M} v \leq (1 - \epsilon_{\text{add}}/4)v_{\min}^\top \mathbf{M} v_{\min} + \epsilon_{\text{add}} \leq 2\lambda_1(\mathbf{M}).$$

Time complexity: By Theorem 4.9, the time of each call to PowerIteration is

$$O\Big(m \log(\frac{n}{\epsilon_{\text{add}}})(\frac{\lambda_n(4\mathbf{I} - \mathbf{M})}{\lambda_{n-1}(4\mathbf{I} - \mathbf{M})} - 1)^{-1}\Big).$$

We now bound $\frac{\lambda_n(4\mathbf{I}-\mathbf{M})}{\lambda_{n-1}(4\mathbf{I}-\mathbf{M})} - 1$. By Theorem 4.6 we have that $\lambda_2(\mathbf{M}) - \lambda_1(\mathbf{M}) \geq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{4c}$, which implies that

$$\lambda_n(4\mathbf{I} - \mathbf{M}) - \lambda_{n-1}(4\mathbf{I} - \mathbf{M}) \geq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{4c}.$$

Since $\mathbf{M}$ is PSD, we also have that $\lambda_{n-1}(4\mathbf{I} - \mathbf{M}) \leq 4$, which gives us

$$\frac{\lambda_n(4\mathbf{I} - \mathbf{M})}{\lambda_{n-1}(4\mathbf{I} - \mathbf{M})} - 1 = \frac{\lambda_n(4\mathbf{I} - \mathbf{M}) - \lambda_{n-1}(4\mathbf{I} - \mathbf{M})}{\lambda_{n-1}(4\mathbf{I} - \mathbf{M})} \geq \frac{\lambda_2(\mathbf{N}_{\overline{G}})}{16c}.$$

This lower bound implies that $\left(\frac{\lambda_n(4\mathbf{I}-\mathbf{M})}{\lambda_{n-1}(4\mathbf{I}-\mathbf{M})} - 1\right)^{-1} \leq \frac{16c}{\lambda_2(\mathbf{N}_{\overline{G}})}$, so the total time complexity is $O\Big(m \log(\frac{n}{\epsilon_{\text{add}}})(\frac{\lambda_n(4\mathbf{I}-\mathbf{M})}{\lambda_{n-1}(4\mathbf{I}-\mathbf{M})} - 1)^{-1}\Big) \leq O\left(m(\lambda_2(\mathbf{N}_{\overline{G}}))^{-1} c \log(\frac{n}{\epsilon_{\text{add}}})\right)$. □

5 Uniformity of $v_{\min}(\mathbf{L}_G)$

In this section we show that $v_{\min}$ of a lossy Laplacian is approximately the all-ones vector when the underlying lossy graph is an expander and its flow multipliers are close to 1. We use this structural property in later sections when performing vertex deletions in our heavy hitter data structure for lossy incidence matrices.

Our proof of the uniformity of $v_{\min}$ can be viewed as a strengthening of the standard argument that the all-ones vector is in the kernel of a non-lossy Laplacian (and is therefore an eigenvector corresponding to the smallest eigenvalue). For a standard graph Laplacian $\mathbf{L}_{\overline{G}}$, any eigenvector v with eigenvalue λ satisfies

$$\lambda v_i = (\mathbf{L}_{\overline{G}} v)_i = d_i v_i - \sum_{j \in \mathcal{N}(i)} v_j.$$

Consequently, when $\lambda = 0$, $(v_{\min})_i = \frac{1}{d_i}\sum_{j \in \mathcal{N}(i)}(v_{\min})_j$, or in other words, for each vertex i, $(v_{\min})_i$ has the average value of its neighboring vertices. This is consistent with the fact that $v_{\min}$ is a scaling of $\mathbb{1}$.

Now consider the lossy Laplacian $\mathbf{L}_G$. Let V_{large} denote the set of vertices for which $(v_{\min})_i$ is at least some threshold ζ, i.e., $V_{\text{large}} = S_{\geq\zeta}(v)$. We show that many vertices adjacent to V_{large} must have $v_{\min}$ values that are larger than $\zeta - \epsilon$ for some small ϵ. If the underlying graph is an expander, after repeating this argument for roughly $\log n$ steps, we can bound each entry of $v_{\min}$. (Similar proof ideas were used for proving spectral clustering results, see e.g. [AALOG18, GKL+21].)

Applying this approach, we prove the following theorem.

Theorem 5.1 (Uniformity of $\frac{v_{\min}}{\sqrt{d}}$)**.** *Let $G = (V, E, \eta)$ be a β_η-balanced lossy flow graph that is connected and has expansion ϕ, with $|E| = m$. Let $d \in \mathbb{R}^V_{\geq 0}$ satisfy $d_G \leq d \leq c \cdot d_G$ for $c \geq 1$, and let $\mathbf{D} = \mathbf{diag}(d)$. Let $\lambda \stackrel{\text{def}}{=} \lambda_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$, and let $v \stackrel{\text{def}}{=} v_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$. If $\beta_\eta < 0.1$, $\phi < 0.1$, $c\lambda < 1$, and $\beta_\eta + c\lambda \leq \frac{\phi^2}{100 \log m}$, then*

$$\frac{\max_{i \in V} z_i}{\min_{i \in V} z_i} \leq \exp\left(O\left(\frac{(\beta_\eta + c\lambda)\log m}{\phi^2}\right)\right), \text{ where } z_i \stackrel{\text{def}}{=} \frac{v_i}{\sqrt{d_i}} \text{ for all } i \in V.$$

Before proving this theorem, consider a lossy graph where $\beta_\eta \leq \frac{\phi^2}{2000c\log^2(m)}$, then Theorem 4.3 and Theorem 4.5 imply that $\lambda = \lambda_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}) \leq 10\beta_\eta$, so the parameters satisfy $\beta_\eta + c\lambda \leq \frac{\phi^2}{100\log^2(m)}$, which means $\exp\left(O\left(\frac{(\beta_\eta+c\lambda)\log m}{\phi^2}\right)\right) \leq 1 + O(\frac{(\beta_\eta+c\lambda)\log m}{\phi^2}) \leq 1 + O(\frac{1}{\log m})$.

Corollary 5.2. *In the setting of Theorem 5.1, if the parameters satisfy $\beta_\eta \leq \frac{\phi^2}{2000c\log^2(m)}$, then*

$$\frac{\max_{i \in V} \frac{v_i}{\sqrt{d_i}}}{\min_{i \in V} \frac{v_i}{\sqrt{d_i}}} \leq 1 + O\left(\frac{1}{\log m}\right).$$

To prove Theorem 5.1, we first prove the following basic fact that $\lambda_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$ is a simple eigenvalue and all entries of $v_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$ are positive.

Fact 5.3. *Let $G = (V, E, \eta)$ be a lossy flow graph that is connected. Let $d \in \mathbb{R}^V_{\geq 0}$ satisfy $d_G \leq d \leq c \cdot d_G$ for some $c \geq 1$, and let $\mathbf{D} = \mathbf{diag}(d)$. Then $\lambda_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$ is a simple eigenvalue, and $v = v_{\min}(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2})$ is strictly positive, i.e., $v_i > 0$ for all $i \in V$.*

Proof. First, note that $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}$ is a matrix with non-positive off diagonal entries, and a positive diagonal. As such, we can write it as $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} = \mu\mathbf{I} - \mathbf{A}$ for some positive integer μ, and some non-negative matrix $\mathbf{A}$. Now, $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}$ and $\mathbf{A}$ have the same eigenspaces. $\mathbf{A}$ is irreducible because G is connected, so the Perron-Frobenius theorem implies that $\lambda_{\max}(\mathbf{A})$ is a simple eigenvalue, and it has a corresponding eigenvector $\lambda_{\max}(\mathbf{A})$ can be scaled to entrywise positive. This vector is also an eigenvector of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}$. □

Next we prove the following two key lemmas. The first lemma provides a lower bound on how much the volume of the set $S_{\geq\zeta}(z)$ increases when ζ is decreased.

Lemma 5.4 (Sweep cut bound of $S_{\geq\zeta}(z)$ with decreasing threshold)**.** *In the setting of Theorem 5.1, for any $\zeta > 0$ such that $\max_{i \in V} z \leq 2\zeta$ and $\sum_{i \in S_{\geq\zeta}(z)} (d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i \in V}(d_{\overline{G}})_i$,*

$$\sum_{i \in S_{\geq\zeta'}(z)} (d_{\overline{G}})_i \geq \left(1 + \frac{\phi}{2}\right) \cdot \sum_{i \in S_{\geq\zeta}(z)} (d_{\overline{G}})_i \quad \text{for } \zeta' \stackrel{\text{def}}{=} \left(1 - \frac{10(\beta_\eta + c\lambda)}{\phi}\right)\zeta.$$

Proof. Let $S \stackrel{\text{def}}{=} S_{\geq\zeta}(z)$ and $S' \stackrel{\text{def}}{=} S_{\geq\zeta'}(z)$. Also recall that we use d to denote our approximation to d_G, the diagonal of the Laplacian $\mathbf{L}_G$, and $d_{\overline{G}}$ to denote the degrees of the smoothed graph $\overline{G}$.

Since v is the eigenvector corresponding to the smallest eigenvalue λ of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}$, it satisfies that $\lambda v = \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}v$. Therefore $z = \mathbf{D}^{-1/2}v$ satisfies $\lambda\mathbf{D}z = \mathbf{L}_G z$. By considering

S and z we see that,

$$\begin{aligned}\lambda \cdot \sum_{i\in S} d_i \cdot z_i = & \sum_{i\in S} (\mathbf{L}_G \cdot z)_i \\ = & \sum_{i\in S} \left(\sum_{e=(a,i)\in E} (z_i - \eta_e z_a) + \sum_{e=(i,a)\in E} (-\eta_e z_a + \eta_e^2 z_i) \right) \\ \geq & \sum_{i\in S} \left((d_G)_i \cdot z_i - (1+\beta_\eta) \sum_{e=(i,a) \text{ or } (a,i)\in E} z_a \right),\end{aligned}$$

where the second step follows from $\mathbf{L}_G = \mathbf{B}_G^\top \mathbf{B}_G$, and the row $e = (a, b)$ of $\mathbf{B}_G$ is $\vec{1}_b - \eta_e \vec{1}_a$, and the third step follows from $\eta_e \leq (1 + \beta_\eta)$. Rearranging yields that:

$$\sum_{i\in S} \Big((d_G)_i - \lambda d_i \Big) \cdot z_i \leq (1+\beta_\eta) \sum_{i\in S} \sum_{e=(a,i) \text{ or } (i,a)\in E} z_a.$$

Since $d \leq c \cdot d_G$, the above equation implies that:

$$\begin{aligned}(1-c\lambda) \sum_{i\in S} (d_G)_i \cdot z_i \leq & (1+\beta_\eta) \sum_{i\in S} \sum_{e=(a,i) \text{ or } (i,a)\in E} z_a \\ = & (1+\beta_\eta) \left(\sum_{i\in S} |E(i,S)| z_i + \sum_{i\in S} \sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a \right),\end{aligned}$$

where in the second step we define $\overline{S} \stackrel{\text{def}}{=} E \backslash S$ and split the last summation over edges from S to $\overline{S}$ and edges from S to S. Rearranging terms yields that

$$\begin{aligned}& (1-c\lambda) \left(\sum_{i\in S} (d_G)_i \cdot z_i - \sum_{i\in S} |E(i,S)| z_i \right) \\ \leq & (\beta_\eta + c\lambda) \sum_{i\in S} |E(i,S)| z_i + (1+\beta_\eta) \sum_{i\in S} \sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a. \end{aligned} \tag{8}$$

Since for every $i \in S$, $(d_G)_i \geq |E(i,S)|$ and $z_i \geq \zeta$,

$$(1-c\lambda) \left(\sum_{i\in S} (d_G)_i \cdot z_i - \sum_{i\in S} |E(i,S)| z_i \right) \geq (1-c\lambda) \left(\sum_{i\in S} (d_G)_i - \sum_{i\in S} |E(i,S)| \right) \zeta. \tag{9}$$

Since $z_i \leq 2\zeta$, we also have

$$\begin{aligned}& (\beta_\eta + c\lambda) \sum_{i\in S} |E(i,S)| z_i + (1+\beta_\eta) \sum_{i\in S} \sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a \\ \leq & 2(\beta_\eta + c\lambda) \sum_{i\in S} |E(i,S)| \zeta + (1+\beta_\eta) \sum_{i\in S} \sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a. \end{aligned} \tag{10}$$

Combining the above three equations Equation (8), (9), and (10) yields that

$$\left((1-c\lambda) \sum_{i\in S} (d_G)_i - \Big(1 - c\lambda + 2(\beta_\eta + c\lambda)\Big) \sum_{i\in S} |E(i,S)| \right) \zeta \leq (1+\beta_\eta) \sum_{i\in S} \sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a. \tag{11}$$

Since $\sum_{i\in S}(d_G)_i \geq \sum_{i\in S}(d_{\overline{G}})_i = (\sum_{i\in S}|E(i,S)|) + |E(S,\overline{S})|$,

$$\begin{aligned}
&(1-c\lambda)\cdot\sum_{i\in S}(d_G)_i - \Big(1-c\lambda+2(\beta_\eta+c\lambda)\Big)\cdot\sum_{i\in S}|E(i,S)| \\
&= (1-c\lambda)\cdot\sum_{i\in S}(d_G)_i - \Big(1+c\lambda+2\beta_\eta\Big)\cdot\left(\sum_{i\in S}(d_{\overline{G}})_i - |E(S,\overline{S})|\right) \\
&\geq\ (1+c\lambda+2\beta_\eta)\cdot|E(S,\overline{S})| - (2\beta_\eta+2c\lambda)\sum_{i\in S}(d_G)_i.
\end{aligned}$$

Since the graph $\overline{G}$ has expansion ϕ, and since $\sum_{i\in S}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$, we have that $|E(S,\overline{S})| \geq \phi\sum_{i\in S}(d_{\overline{G}})_i \geq \phi(1+\beta_\eta)^{-2}\sum_{i\in S}(d_G)_i$, hence the above equation gives

$$\begin{aligned}
&(1-c\lambda)\cdot\sum_{i\in S}(d_G)_i - \Big(1-c\lambda+2(\beta_\eta+c\lambda)\Big)\cdot\sum_{i\in S}|E(i,S)| \\
&\geq\ \left(1+c\lambda+2\beta_\eta - \frac{(2\beta_\eta+2c\lambda)(1+\beta_\eta)^2}{\phi}\right)\cdot|E(S,\overline{S})| \\
&\geq\ \left(1-\frac{3(\beta_\eta+c\lambda)}{\phi}\right)\cdot|E(S,\overline{S})|.
\end{aligned}$$

Combining this equation with Equation (11) yields that

$$\begin{aligned}
\sum_{i\in S}\sum_{e=(i,a)\text{ or }(a,i)\in E(S,\overline{S})} z_a &\geq \left(1-\frac{3(\beta_\eta+c\lambda)}{\phi}-\beta_\eta\right)\cdot|E(S,\overline{S})|\cdot\zeta \\
&\geq \left(1-\frac{4\beta_\eta+3c\lambda}{\phi}\right)\cdot|E(S,\overline{S})|\cdot\zeta. \qquad (12)
\end{aligned}$$

Let $T\subseteq\overline{S}$ denote the set of vertices such that $z_a \geq \zeta' = (1-\frac{10(\beta_\eta+c\lambda)}{\phi})\cdot\zeta$. Equation (12) implies that we must have $|E(S,T)| \geq \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$, and next we prove this claim by contradiction. Assume that we instead have $|E(S,T)| < \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$. First note that since any $a\notin S$ satisfies $z_a\leq\zeta$ and any $a\notin S\cup T$ satisfies $z_a \leq (1-\frac{10(\beta_\eta+c\lambda)}{\phi})\cdot\zeta$,

$$\begin{aligned}
\sum_{i\in S}\sum_{e=(i,a)\text{ or }(a,i)\in E(S,\overline{S})} z_a &= \sum_{i\in S}\sum_{e=(i,a)\text{ or }(a,i)\in E(S,T)} z_a + \sum_{i\in S}\sum_{e=(i,a)\text{ or }(a,i)\in E(S,\overline{S}\setminus T)} z_a \\
&\leq \zeta\cdot|E(S,T)| + \left(1-\frac{10(\beta_\eta+c\lambda)}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S}\setminus T)| \\
&= \frac{10(\beta_\eta+c\lambda)}{\phi}\cdot\zeta\cdot|E(S,T)| + \left(1-\frac{10(\beta_\eta+c\lambda)}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})|,
\end{aligned}$$

where in the third step we used that $|E(S,\overline{S}\setminus T)| = |E(S,\overline{S})| - |E(S,T)|$. Using the assumption that $|E(S,T)| < \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$, and since $|E(S,\overline{S})| \geq \phi\sum_{i\in S}(d_{\overline{G}})_i$, the above equation gives

$$\begin{aligned}
\sum_{i\in S}\sum_{e=(i,a)\text{ or }(a,i)\in E(S,\overline{S})} z_a &< \frac{10(\beta_\eta+c\lambda)}{2}\cdot\zeta\cdot\sum_{i\in S}(d_{\overline{G}})_i + \left(1-\frac{10(\beta_\eta+c\lambda)}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})| \\
&\leq \left(1-\frac{10(\beta_\eta+c\lambda)}{\phi}+\frac{10(\beta_\eta+c\lambda)}{2\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})| \\
&< \left(1-\frac{4\beta_\eta+3c\lambda}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})|.
\end{aligned}$$

This inequality contradicts with Equation (12), so we have proved $|E(S,T)| \geq \frac{\phi}{2} \cdot \sum_{i\in S}(d_{\overline{G}})_i$.

Finally note that by definition $S' = S \cup T$, and $|E(S,T)| \leq \sum_{i\in T}(d_{\overline{G}})_i$, so we have

$$\sum_{i\in S'}(d_{\overline{G}})_i = \sum_{i\in S}(d_{\overline{G}})_i + \sum_{i\in T}(d_{\overline{G}})_i \geq \sum_{i\in S}(d_{\overline{G}})_i + |E(S,T)| \geq \left(1+\frac{\phi}{2}\right)\cdot \sum_{i\in S}(d_{\overline{G}})_i. \qquad \square$$

Note that Theorem 5.4 only holds for sets $S \subseteq V$ where $\sum_{i\in S}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$. For the rest of the vertices, the following lemma provides a lower bound on how much the volume of the set $S_{\leq\zeta}(z)$ increases when ζ is increased.

Lemma 5.5 (Sweep cut bound of $S_{\leq\zeta}(z)$ with increasing threshold)**.** *In the setting of Theorem 5.1, for any $\zeta > 0$ such that $\sum_{i\in S_{\leq\zeta}(z)}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$,*

$$\sum_{i\in S_{\leq\zeta'}(z)}(d_{\overline{G}})_i \geq \left(1+\frac{\phi}{2}\right)\cdot \sum_{i\in S_{\leq\zeta}(z)}(d_{\overline{G}})_i \quad \textit{for } \zeta' \stackrel{\text{def}}{=} \left(1+\frac{10\beta_\eta}{\phi}\right)\zeta.$$

Proof. In this proof we define $S \stackrel{\text{def}}{=} S_{\leq\zeta}(z)$ and $S' \stackrel{\text{def}}{=} S_{\leq\zeta'}(z)$. Since v is the eigenvector corresponding to the smallest eigenvalue λ of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}$, we have that $\lambda v = \mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2}v$. So we have that $\lambda\mathbf{D}z = \mathbf{L}_G z$. By considering the vertices in S for the previous equation, we get:

$$\begin{aligned}
\lambda\cdot\sum_{i\in S} d_i\cdot z_i &= \sum_{i\in S}(\mathbf{L}_G\cdot z)_i \\
&= \sum_{i\in S}\left(\sum_{e=(a,i)\in E}(z_i-\eta_e z_a) + \sum_{e=(i,a)\in E}(-\eta_e z_a+\eta_e^2 z_i)\right) \\
&\leq \sum_{i\in S}\left((d_G)_i\cdot z_i - \sum_{e=(i,a) \text{ or } (a,i)\in E} z_a\right),
\end{aligned}$$

where the second step follows from $\mathbf{L}_G = \mathbf{B}_G^\top\mathbf{B}_G$, and the row $e=(a,b)$ of $\mathbf{B}_G$ is $\vec{1}_b - \eta_e\vec{1}_a$, and the third step follows since $\eta_e \geq 1$ and all $z_a \geq 0$ by Theorem 5.3. Rearranging, we get

$$\sum_{i\in S}\Big((d_G)_i - \lambda d_i\Big)\cdot z_i \geq \sum_{i\in S}\sum_{e=(a,i) \text{ or } (i,a)\in E} z_a.$$

Since $d_G \leq d \leq c\cdot d_G$, we have that $(d_G)_i - \lambda d_i \leq (1-\lambda)(d_G)_i$, and combining with the above equation we have

$$\begin{aligned}
(1-\lambda)\sum_{i\in S}(d_G)_i\cdot z_i &\geq \sum_{i\in S}\sum_{e=(a,i) \text{ or } (i,a)\in E} z_a \\
&= \sum_{i\in S}|E(i,S)|z_i + \sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a,
\end{aligned}$$

where in the second step we define $\overline{S} \stackrel{\text{def}}{=} E\backslash S$ and split the last summation over edges from S to $\overline{S}$ and edges from S to S. Rearranging terms yields that

$$(1-\lambda)\left(\sum_{i\in S}(d_G)_i\cdot z_i - \sum_{i\in S}|E(i,S)|z_i\right) \geq \lambda\sum_{i\in S}|E(i,S)|z_i + \sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a.$$

Since for every $i \in S$, $(d_G)_i \geq |E(i,S)|$ and $0 \leq z_i \leq \zeta$, the above inequality implies that

$$(1-\lambda)\left(\sum_{i\in S}(d_G)_i - \sum_{i\in S}|E(i,S)|\right)\zeta \geq \sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a.$$

Since $\sum_{i\in S}(d_G)_i \leq (1+\beta_\eta)^2 \sum_{i\in S}(d_{\overline{G}})_i = (1+\beta_\eta)^2 \cdot \big(|E(S,\overline{S})| + \sum_{i\in S}|E(i,S)|\big)$, the above inequality implies that

$$\begin{aligned}
\sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a &\leq (1-\lambda)\left((1+\beta_\eta)^2|E(S,\overline{S})| + (2\beta_\eta+\beta_\eta^2)\sum_{i\in S}|E(i,S)|\right)\zeta \\
&\leq \left((1+\beta_\eta)^2|E(S,\overline{S})| + \frac{(2\beta_\eta+\beta_\eta^2)}{\phi}|E(S,\overline{S})|\right)\zeta \\
&\leq \left(1+\frac{4\beta_\eta}{\phi}\right)|E(S,\overline{S})|\zeta, \qquad (13)
\end{aligned}$$

where the second step follows from $\overline{G}$ has expansion ϕ, which means $|E(S,\overline{S})| \geq \phi \sum_{i\in S}(d_{\overline{G}})_i$, and so $\sum_{i\in S}|E(i,S)| \leq \sum_{i\in S}(d_{\overline{G}})_i \leq \frac{1}{\phi}|E(S,\overline{S})|$.

Let $T \subseteq \overline{S}$ denote the set of vertices such that $z_a \leq \zeta' = (1+\frac{10\beta_\eta}{\phi})\cdot\zeta$. Equation (13) implies that we must have $|E(S,T)| \geq \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$, and next we prove this claim by contradiction. Assume that we instead have $|E(S,T)| < \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$. First note that since any $a \notin S$ satisfies $z_a \geq \zeta$ and any $a \notin S\cup T$ satisfies $z_a \geq (1+\frac{10\beta_\eta}{\phi})\cdot\zeta$,

$$\begin{aligned}
\sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a &= \sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,T)} z_a + \sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S}\setminus T)} z_a \\
&\geq \zeta\cdot|E(S,T)| + \left(1+\frac{10\beta_\eta}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S}\setminus T)| \\
&= -\frac{10\beta_\eta}{\phi}\cdot\zeta\cdot|E(S,T)| + \left(1+\frac{10\beta_\eta}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})|,
\end{aligned}$$

where in the third step we used that $|E(S,\overline{S}\setminus T)| = |E(S,\overline{S})| - |E(S,T)|$. Using the assumption that $|E(S,T)| < \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$, and since $|E(S,\overline{S})| \geq \phi\sum_{i\in S}(d_{\overline{G}})_i$, the above equation gives

$$\begin{aligned}
\sum_{i\in S}\sum_{e=(i,a) \text{ or } (a,i)\in E(S,\overline{S})} z_a &> -\frac{10\beta_\eta}{2}\cdot\zeta\cdot\sum_{i\in S}(d_{\overline{G}})_i + \left(1+\frac{10\beta_\eta}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})| \\
&\geq \left(1+\frac{10\beta_\eta}{\phi}-\frac{10\beta_\eta}{2\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})| \\
&\geq \left(1+\frac{5\beta_\eta}{\phi}\right)\cdot\zeta\cdot|E(S,\overline{S})|.
\end{aligned}$$

This inequality contradicts with Equation (13), so we have proved $|E(S,T)| \geq \frac{\phi}{2}\cdot\sum_{i\in S}(d_{\overline{G}})_i$.

Finally note that by definition $S' = S\cup T$, and $|E(S,T)| \leq \sum_{i\in T}(d_{\overline{G}})_i$, so we have

$$\sum_{i\in S'}(d_{\overline{G}})_i = \sum_{i\in S}(d_{\overline{G}})_i + \sum_{i\in T}(d_{\overline{G}})_i \geq \sum_{i\in S}(d_{\overline{G}})_i + |E(S,T)| \geq \left(1+\frac{\phi}{2}\right)\cdot\sum_{i\in S}(d_{\overline{G}})_i. \qquad \square$$

When S is the set of vertices with z value greater than ζ, we have shown in Theorem 5.4 that a large fraction of the neighboring vertices of S has z value that is only slightly smaller than ζ. Similarly we have also shown in Theorem 5.5 that when S is the set of vertices with z value less than ζ, a large fraction of the neighboring vertices of S has z value that is only slightly larger than ζ. Using these two lemmas, together with the fact that the underlying graph is an expander, we can prove the bound on z over the whole graph.

Proof of Theorem 5.1. Initially let $\zeta_0 \stackrel{\text{def}}{=} \max_i z_i$ and let $S_0 \stackrel{\text{def}}{=} S_{\geq \zeta_0}(z)$, and note that $|S_0| \geq 1$. For any integer $k \geq 1$, let $\zeta_k \stackrel{\text{def}}{=} (1 - \frac{10(\beta_\eta + c\lambda)}{\phi})\zeta_{k-1}$, and let $S_k \stackrel{\text{def}}{=} S_{\geq \zeta_k}(z)$. For any integer $k \geq 0$ such that $(1 - \frac{10(\beta_\eta+c\lambda)}{\phi})^k \geq \frac{1}{2}$ and $\sum_{i\in S_k}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$, any $k' < k$ also satisfies these two conditions, so applying Theorem 5.4 for $k+1$ times yields that

$$\sum_{i\in S_{k+1}}(d_{\overline{G}})_i \geq \left(1+\frac{\phi}{2}\right)\cdot\sum_{i\in S_k}(d_{\overline{G}})_i \geq \cdots \geq \left(1+\frac{\phi}{2}\right)^{k+1}\sum_{i\in S_0}(d_{\overline{G}})_i \geq \left(1+\frac{\phi}{2}\right)^{k+1}.$$

Let k^* denote the largest integer such that $(1 - \frac{10(\beta_\eta+c\lambda)}{\phi})^{k^*} \geq \frac{1}{2}$ and $\sum_{i\in S_{k^*}}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$. Then the above inequality implies that $\left(1+\frac{\phi}{2}\right)^{k^*} \leq \sum_{i\in S_{k^*}}(d_{\overline{G}})_i \leq m$, which gives that $k^* \leq \frac{2\log m}{\phi}$. Also note that $(1-\frac{10(\beta_\eta+c\lambda)}{\phi})^{k^*} \geq \frac{1}{2}$ must be true because of the assumption $(\beta_\eta+c\lambda)\log m \leq \frac{1}{100}\phi^2$. So by definition the set S_{k^*+1} satisfies $\sum_{i\in S_{k^*+1}}(d_{\overline{G}})_i > \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$, and all vertices $i \in S_{k^*+1}$ satisfy

$$\frac{v_i}{\sqrt{d_i}} \geq \zeta_{k^*+1} \geq \left(1-\frac{10(\beta_\eta+c\lambda)}{\phi}\right)^{2\log m/\phi+1}\cdot\zeta_0 \geq \exp\left(-O\left(\frac{(\beta_\eta+c\lambda)\log m}{\phi^2}\right)\right)\cdot\max_{i'}\frac{v_{i'}}{\sqrt{d_{i'}}}. \tag{14}$$

Similarly, we let $\zeta'_0 \stackrel{\text{def}}{=} \min_i z_i$, and $S'_0 \stackrel{\text{def}}{=} S_{\leq\zeta'_0}(z)$. Note that $\zeta'_0 > 0$ by Theorem 5.3, and $|S'_0| \geq 1$. For any integer $k \geq 1$, let $\zeta'_k \stackrel{\text{def}}{=} (1 + \frac{10\beta_\eta}{\phi})\zeta'_{k-1}$, and $S'_k \stackrel{\text{def}}{=} S_{\leq\zeta'_k}(z)$. For any integer $k \geq 0$ such that $\sum_{i\in S'_k}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$, applying Theorem 5.5 for $k+1$ times yields that

$$\sum_{i\in S'_{k+1}}(d_{\overline{G}})_i \geq (1+\frac{\phi}{2})^{k+1}.$$

Let k'^* denote the largest interger such that $\sum_{i\in S'_{k'^*}}(d_{\overline{G}})_i \leq \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$. Then the above inequality implies that $\left(1+\frac{\phi}{2}\right)^{k'^*} \leq \sum_{i\in S'_{k'^*}}(d_{\overline{G}})_i \leq m$, which gives that $k'^* \leq \frac{2\log m}{\phi}$. So by definition the set $S'_{k'^*+1}$ satisfies $\sum_{i\in S'_{k'^*+1}}(d_{\overline{G}})_i > \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$, and all vertices $i \in S'_{k'^*+1}$ satisfy

$$\frac{v_i}{\sqrt{d_i}} \leq \zeta'_{k'^*+1} \leq \left(1+\frac{10\beta_\eta}{\phi}\right)^{2\log m/\phi+1}\cdot\zeta'_0 \leq \exp\left(O\left(\frac{\beta_\eta\log m}{\phi^2}\right)\right)\cdot\min_{i'}\frac{v_{i'}}{\sqrt{d_{i'}}}, \tag{15}$$

Finally since $\sum_{i\in S_{k^*+1}}(d_{\overline{G}})_i > \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$ and $\sum_{i\in S'_{k'^*+1}}(d_{\overline{G}})_i > \frac{1}{2}\sum_{i\in V}(d_{\overline{G}})_i$, we have that $S_{k^*+1}\cap S'_{k'^*+1} \neq \emptyset$. Combining Equation (14) and Equation (15) for $i \in S_{k^*+1}\cap S'_{k'^*+1}$ gives us the claimed bound of this theorem. □

6 Heavy Hitters for Balanced Lossy Expanders

In this section we present our heavy hitter data structure for lossy graphs that are β_η-balanced ϕ-expanders undergoing edge and vertex deletions, under the assumption that the degree of each vertex remains within a 1/9 fraction of its original degree. The pseodocode for the data structure is given in Algorithm 1 and 2. This section proves Theorem 6.1 below which analyzes the pseudocode and thereby essentially provides the formal version of Section 2.2 from the technical overview and serves as a key building block towards our heavy hitter data structure for general two-sparse matrices, which we will present in the next section.

Before analyzing Theorem 6.1, we first provide an overview of QueryHeavy, a key operation of this data structure (Line 5 to 14), which find entries e of $\mathbf{B}h$ such that $(\mathbf{B}h)_e \geq \epsilon$ for a query h. All other operations supported by the data structure are either straightforward or easily implementable using the variables maintained to efficiently answer these heavy hitter queries.

Finding the heavy hitters of $\mathbf{B}h$ is equivalent to finding the heavy hitters of $\mathbf{B}\mathbf{D}^{-1/2}\mathbf{D}^{1/2}h$, so QueryHeavy finds the heavy hitters of $\mathbf{B}\mathbf{D}^{-1/2}g$ for $g = \mathbf{D}^{1/2}h$. As discussed in the technical overview, the key technical part of the algorithm is to maintain an approximate vector v to the smallest eigenvector $v_{\min}$ of the normalized lossy Laplacian $\widetilde{\mathbf{N}}_G$. Using v, the algorithm decomposes $g = g^v + g^{\perp v}$ where $g^v = vv^\top g$ and $g^{\perp v} = (\mathbf{I} - vv^\top)g$, and then finds the heavy hitters of $\mathbf{B}\mathbf{D}^{-1/2}g^v$ and $\mathbf{B}\mathbf{D}^{-1/2}g^{\perp v}$ separately (see Line 9 and Line 11). By Theorem 4.1, a unit vector v is a sufficiently good approximation to $v_{\min}$ if $v^\top\widetilde{\mathbf{N}}_G v \leq 10\lambda_1(\widetilde{\mathbf{N}}_G)$. In each heavy hitter query, we check whether this condition still holds by testing the necessary condition $\|g^{\perp v}\|_2^2 \leq \widetilde{O}(g^\top\widetilde{\mathbf{N}}_G g)$ (see Line 8), where the norms can be computed efficiently using a Johnson-Lindenstrauss sketch. If the condition no longer holds, then we recompute v. One final detail is that the Rayleigh quotient $v^\top\widetilde{\mathbf{N}}_G v$ may increase when v is renormalized after vertex deletions (see Line 20). Crucially, by Theorem 5.1, $\frac{v}{\sqrt{d}}$ is approximately uniform, so as long as we do not delete too many edges, this renormalization cannot increase the Rayleigh quotient by too much.

Theorem 6.1 (Heavy hitters on balanced lossy expanders)**.** *There is a data structure (Algorithm 1) that supports the following operations w.h.p. against adaptive inputs:*

- Initialize$(G = (V, E, \eta), \epsilon_{\text{add}} \in (0, 1), \overline{\tau} \in \mathbb{R}^E_{\geq 0})$*: Initializes with a lossy β_η-balanced ϕ-expander $G = (V, E, \eta)$ where $|V| = n$, $|E| = m$, and $\beta_\eta \leq \frac{\phi^2}{10^5\log^2(m)}$, a parameter $0 \leq \epsilon_{\text{add}} \leq \frac{\phi^2}{10^5\log^2(m)}$, and a vector $\overline{\tau} \geq \mathbb{0}$ in amortized $O(m\phi^{-2}\log^2(n/\epsilon_{\text{add}})\log(m))$ time.*
- Delete$(F \subseteq E)$*: Delete the edges in F from G in amortized $O(|F|\log m)$ time, also remove any vertex whose degree drops to 0, provided that the graph after deletion is still a ϕ-expander and the degree of any vertex v remains greater than 1/9 of the original initialized degree.*
- ScaleTau$(e \in E, b \in \mathbb{R}_{\geq 0})$*: Sets $\overline{\tau}_e \leftarrow b$ in worst-case $O(1)$ time.*
- QueryHeavy$(h \in \mathbb{R}^V, \epsilon \in (0, 1))$*: Returns a set $I \subseteq E$ containing exactly all $e \in E$ that satisfies $|\mathbf{B}_G h|_e \geq \epsilon$ in amortized $O\left(\phi^{-4}\epsilon^{-2}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon^{-2}\epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2 + n\right)$ time.*
- Norm$(h \in \mathbb{R}^V)$*: Returns $L \in \mathbb{R}_{\geq 0}$ in amortized $O(n)$ time such that*

$$\|\mathbf{B}_G h\|_2^2 \leq L \leq O\left(\phi^{-4}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2\right).$$

- Sample$(h \in \mathbb{R}^V, C_0, \overline{C}_1, \overline{C}_2, C_3)$*: Let a vector $p \in \mathbb{R}^E$ satisfy*

$$p_e \geq \min\left\{1,\ \overline{C_1}\cdot(\mathbf{B}_G h)_e^2 + \overline{C}_2 + C_3\overline{\tau}_e\right\}.$$

Let $S = \sum_{e\in E} p_e$. Let X be a random variable which equals to $p_e^{-1}\vec{1}_e$ with probability p_e/S for all $e \in E$. This operation returns a random diagonal matrix $\mathbf{R} = C_0^{-1}\sum_{j=1}^{C_0 S} \mathbf{diag}(X_j)$, where X_j are i.i.d. copies of X. The amortized time of this operation and also the output size of $\mathbf{R}$ are bounded by

$$O\left(C_0\overline{C}_1\phi^{-4}\log m(\|\mathbf{B}_G h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2) + C_0\overline{C}_2 m\log m + C_0 C_3\|\overline{\tau}\|_1\log m + n\log n\right).$$

Algorithm 1: Heavy hitter on β_η-balanced lossy ϕ-expander

Global: $v \in \mathbb{R}^V$, $u \in \mathbb{R}^E$, $\overline{\tau} \in \mathbb{R}^E_{\geq 0}$, an array π_u of size $|E|$, a diagonal matrix $\mathbf{D} \in \mathbb{R}^{V\times V}$, $\mathbf{J} \in \mathbb{R}^{k\times E}$, $\mathbf{M} \in \mathbb{R}^{k\times V}$ // $k = O(\log|E|)$ `chosen at initialization`

procedure `Initialize`$(G = (V, E, \eta), \epsilon_{\text{add}} \in (0,1), \overline{\tau} \in \mathbb{R}^E_{\geq 0})$`:`

  $\mathbf{D} \leftarrow \mathbf{D}_{\overline{G}}$, $\overline{\tau} \leftarrow \overline{\tau}$

  RESET()

procedure `QueryHeavy`$(h \in \mathbb{R}^V, \epsilon \in (0,1))$`:`

  $g \leftarrow \mathbf{D}^{1/2}h$, $g^v \leftarrow vv^\top g$, and $g^{\perp v} \leftarrow (\mathbf{I} - vv^\top)g$

  Compute $t = \phi^{-4}\cdot\left(\|\mathbf{M}h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2\right)$

  if $\|g^{\perp v}\|_2^2 \geq 5\cdot 10^6 t$ **then** RESET()

  Compute $I \leftarrow S_{\geq \frac{\epsilon}{2}}(\|g^v\|_2 \cdot |u|)$ by binary search on π_u

  // `Compute heavy entries of` $\mathbf{B}_G\mathbf{D}^{-1/2}g^v = \|g^v\|_2 \cdot u$

  for $j \in V$ **do**

    if $|d_j^{-1/2}g_j^{\perp v}| \geq \frac{\epsilon}{6}$ **then**

      Add all edges $e \in E$ that are adjacent to vertex j to I

      // `Compute heavy entries of` $\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v}$

  Remove from I all edges e that doesn't satisfy $|\mathbf{B}_G h|_e \geq \epsilon$.

  return I

procedure `Delete`$(F \subseteq E)$`:`

  $\mathbf{M} \leftarrow \mathbf{M} - \mathbf{J}_{:,F}(\mathbf{B}_G)_{F,:}$

  Delete the edges in F from the graph G, the vector u, and the list π_u.

  Let $S \subset V$ be the set of vertices adjacent to edges in F whose degree has dropped to 0. Delete the vertices S from the graph G and the matrix $\mathbf{D}$.

  $\mathbf{M} \leftarrow \mathbf{M}_{:V\setminus S}$

  $R \leftarrow \|v_{V\setminus S}\|_2$, $v \leftarrow \frac{v_{V\setminus S}}{R}$, $u \leftarrow \frac{u}{R}$

  if *number of edges in G decreased by a factor of 2 since the last reset* **then** RESET()

procedure `Reset()`**:**

  // `Reset` v `to a close approximation of` $v_{\min}$`, generate a new JL matrix` $\mathbf{J}$`, and update` u, π_u, $\mathbf{M}$ `accordingly`

  $v \leftarrow \text{POWERITERATION}\left(4\mathbf{I} - (\mathbf{D}^{-1/2}\mathbf{B}_G^\top\mathbf{B}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}), \epsilon_{\text{add}}/4\right)$ // `Theorem 4.9`

  $u \leftarrow \mathbf{B}_G\mathbf{D}^{-1/2}\cdot v$

  Compute an array π_u of indices such that $|u_{\pi_u(1)}| \geq |u_{\pi_u(2)}| \geq \cdots \geq |u_{\pi_u(|E|)}|$

  $\mathbf{J} \leftarrow \text{JL}(|V|, \text{poly}(|E|), 0.01, |V|^{-10}) \in \mathbb{R}^{k\times E}$ where $k = O(\log|E|)$ // `Theorem 6.3`

  $\mathbf{M} \leftarrow \mathbf{J}\mathbf{B}_G$

To prove Theorem 6.1, we use the following tools:

Algorithm 2: Heavy hitter on β_η-balanced lossy ϕ-expander (Algorithm 1 continued)

procedure `ScaleTau`$(e \in E, b \in \mathbb{R}_{\geq 0})$**:** $\overline{\tau}_e \leftarrow b$

procedure `Norm`$(h \in \mathbb{R}^V)$**:**

  Execute Line 6 to Line 8 of QueryHeavy to update v if necessary, and to compute vectors $g, g^v, g^{\perp v}$.

  return $10(\|g^v\|_2^2 \cdot \|u\|_2^2 + \|g^{\perp v}\|_2^2)$

procedure `Sample`$(h \in \mathbb{R}^V, C_0, \overline{C}_1, \overline{C}_2, C_3)$**:**

  Execute Line 6 to Line 8 of QueryHeavy to update v if necessary, and to compute vectors $g, g^v, g^{\perp v}$.

  Implicitly define $p_e = \min\left\{1,\ 5\overline{C}_1\big(\|g^v\|_2^2 u_e^2 + \frac{(g^{\perp v}_{a_e})^2}{d_{a_e}} + \frac{(g^{\perp v}_{b_e})^2}{d_{b_e}}\big) + \overline{C}_2 + C_3\overline{\tau}_e\right\}$, $\forall e \in E$

  $S_1 \leftarrow 5\overline{C}_1\|g^v\|_2^2\|u\|_2^2$, $S_2 \leftarrow \sum_{i\in V} \frac{5\overline{C}_1(g_i^{\perp v})^2\cdot(d_{\overline{G}})_i}{d_i}$, $S_3 \leftarrow \overline{C}_2 m + C_3\|\overline{\tau}\|_1$, $S \leftarrow S_1 + S_2 + S_3$

  for $j \in [C_0 S]$ **do**

    Sample a uniformly random number $r \in [0, 1]$.

    if $r \leq \frac{S_1}{S}$ **then**

      Let $x_j \leftarrow p_e^{-1}\vec{1}_e$ with probability $\frac{5\overline{C}_1\|g^v\|_2^2 u_e^2}{S_1}$ for each $e \in E$.

    if $\frac{S_1}{S} < r \leq \frac{S_1+S_2}{S}$ **then**

      Sample a vertex $i \in V$ with probability $\frac{5\overline{C}_1(g_i^{\perp v})^2\cdot(d_{\overline{G}})_i}{d_i \cdot S_2}$, then uniformly sample an edge e adjacent to i, and let $x_j \leftarrow p_e^{-1}\vec{1}_e$.

    if $\frac{S_1+S_2}{S} < r \leq 1$ **then**

      Let $x_j \leftarrow p_e^{-1}\vec{1}_e$ with probability $\frac{\overline{C}_2 + C_3\overline{\tau}_e}{S_3}$ for each $e \in E$.

  return $\mathbf{R} = C_0^{-1}\sum_{j=1}^{C_0 S} \mathbf{diag}(x_j)$

Cheeger's inequality. We use Cheeger's inequality which relates the conductance of a graph and the second smallest eigenvalue of its normalized Laplacian.

Theorem 6.2 (Cheeger's Inequality)**.** *Let $G = (V, E, w)$ be a weighted graph, let $\mathbf{N}_G$ be its normalized Laplacian, and let $\phi(G)$ be the conductance of G. Then,*

$$\frac{\phi(G)^2}{2} \leq \lambda_2(\mathbf{N}_G) \leq 2 \cdot \phi(G).$$

Using Cheeger's inequality, the lossy β_η-balanced ϕ-expander G of Theorem 6.1 satisfies $\lambda_2(\mathbf{N}_{\overline{G}}) \geq \frac{\phi^2}{2} > 20\beta_\eta$, allowing us to use the spectral result Theorem 4.1.

JL estimate. We use the Johnson-Lindenstrauss Lemma that allows us to approximate the ℓ_2 norm of a vector via a lower-dimensional embedding:

Lemma 6.3 (Johnson-Lindenstrauss Lemma [JL84])**.** *There exists a function $\mathrm{JL}(n, m, \epsilon, \delta)$ that returns a random matrix $\mathbf{J} \in \mathbb{R}^{k\times n}$ where $k = O(\epsilon^{-2}\log(m/\delta))$, and $\mathbf{J}$ satisfies that for any m-element subset $V \subset \mathbb{R}^n$,*

$$\Pr\left[\forall v \in V, (1-\epsilon)\|v\|_2 \leq \|\mathbf{J}v\|_2 \leq (1+\epsilon)\|v\|_2\right] \geq 1 - \delta.$$

Furthermore, the function JL runs in $O(kn)$ time.

Power Method. We also use the standard power iteration to compute the maximum eigenvector of a matrix, as defined in Theorem 4.9. The guarantees of applying PowerIteration to our desired matrix $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$ are given in Theorem 4.10.

Proof of Theorem 6.1. **Correctness of JL estimates against adaptive inputs:** We first prove that w.h.p. in any execution of QueryHeavy$(\cdot)$, $\|\mathbf{M}h\|_2 \approx_{1.01} \|\mathbf{B}_G h\|_2$. Since the IPM makes at most poly(m) calls to QueryHeavy, the Johnson-Lindenstrauss lemma (Theorem 6.3) implies that $\|\mathbf{M}h\|_2 \approx_{1.01} \|\mathbf{B}_G h\|_2$ for all queries with high probability, provided the query vectors h are oblivious to the JL matrix $\mathbf{J}$.

Next we prove that the query vectors h are indeed oblivious to $\mathbf{J}$. We consider an auxiliary algorithm AlgSlow that computes $t' = 2\phi^{-4} \cdot \left(\|\mathbf{B}_G h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2\right)$ on Line 7, and uses t' instead of t on Line 8 to determine if $\|g^{\perp v}\|_2^2 \geq 5 \cdot 10^6 t'$. The outputs of Algorithm 1 and AlgSlow are exactly the same until one of them enters the if-clause on Line 8. Conditioned on the high probability event that $\|\mathbf{M}h\|_2 \approx_{1.01} \|\mathbf{B}_G h\|_2$ for all previous queries, whenever AlgSlow enters the if-clause on Line 8, Algorithm 1 also does so, and Algorithm 1 generates a new JL matrix $\mathbf{J}$ whenever this happens. Consequently, with high probability the output of Algorithm 1 is independent of the current JL matrix $\mathbf{J}$.

For the remainder of the proof, we condition on the high probability event that $\|\mathbf{M}h\|_2 \approx_{1.01} \|\mathbf{B}_G h\|_2$ in any execution of QueryHeavy$(\cdot)$.

Time complexity of Initialize and Reset: We first bound the running time of one call to Reset$(\cdot)$. The most time-consuming step in Reset$(\cdot)$ is the call to PowerIteration$(\cdot)$, which by Theorem 4.10 takes $O(m\phi^{-2}\log(\frac{n}{\epsilon_{\text{add}}}))$ time since $c = 9$, and because Delete$(\cdot)$ guarantees the degree of any vertex v remains greater than $1/9$ of the original initialized degree or drops to 0. All other steps, including computing $\mathbf{M} = \mathbf{J}\mathbf{B}_G$, and sorting the list π_u, can all be computed in $O(m \log m)$ time.

Note that apart from Initialize$(\cdot)$, Reset$(\cdot)$ is only called inside QueryHeavy$(\cdot)$ and Delete$(\cdot)$. Next we prove that during all QueryHeavy$(\cdot)$ and Delete$(\cdot)$, Reset$(\cdot)$ is invoked for at most $O(\log(\frac{n}{\epsilon_{\text{add}}}) + \log(m))$ times, which then proves that the total runtime of all Reset$(\cdot)$ is bounded by $O(m\phi^{-2}\log^2(\frac{n}{\epsilon_{\text{add}}})\log(m))$ as claimed. Note that since Reset$(\cdot)$ is invoked inside a Delete$(\cdot)$ only when the number of edges in G decreased by a factor of 2 since the last reset (see Line 21 of Algorithm 1), we have that during all Delete$(\cdot)$, Reset$(\cdot)$ is invoked for at most $O(\log(m))$ times. It remains to prove that during all QueryHeavy$(\cdot)$, Reset$(\cdot)$ (see Line 8 of Algorithm 1) is invoked for at most $\log(\epsilon_{\text{add}}^{-1})$ number of times.

Line 8 is executed only if $\|g^{\perp v}\|_2^2 \geq 5 \cdot 10^6 t$, and we will show that in this case we must have

$$v^\top \left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right) v \geq 10\lambda_1 \left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right). \tag{16}$$

Suppose on the contrary that $v^\top(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})v < 10\lambda_1(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})$. This assumption satisfies the requirement on v for Theorem 4.1 with $c_2 = 10$. We next show that $\mathbf{D} \approx_{30} \mathbf{D}_G$, which would satisfy the requirement on d with $c_1 = 30$ for Theorem 4.1. The guarantees of Delete$(\cdot)$ ensure that $\mathbf{D} \approx_9 \mathbf{D}_{\overline{G}}$. Since $\beta_\eta \leq 0.01$, Theorem 4.3 implies that $\mathbf{D}_{\overline{G}} \approx_{1.03} \mathbf{D}_G$, and so $\mathbf{D} \approx_{30} \mathbf{D}_G$. Finally, by Cheeger's inequality (Theorem 6.2) and the fact that G is a lossy β_η-balanced ϕ-expander with $\beta_\eta \leq \frac{\phi^2}{10^5\log^2(n)}$, the second smallest eigenvalue of $\mathbf{N}_{\overline{G}}$ satisfies $\lambda_2(\mathbf{N}_{\overline{G}}) \geq \frac{\phi^2}{2} > 20\beta_\eta$. Hence, all requirements of Theorem 4.1 are satisfied. Now, applying Theorem 4.1 with $c_1 = 30, c_2 = 10$,

$$\mathbf{I} - (1-\lambda)vv^\top \preceq \frac{48 \cdot 30^2 \cdot 10^2}{\phi^4}\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right) \preceq \frac{4.4 \cdot 10^6}{\phi^4}\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right).$$

Note that $\|g^{\perp v}\|_2^2 = g^\top \left(\mathbf{I} - vv^\top\right) g \le g^\top \left(\mathbf{I} - (1-\lambda)vv^\top\right) g$, so the above equation implies that

$$\begin{aligned}\|g^{\perp v}\|_2^2 &\le \frac{4.4\cdot 10^6}{\phi^4} g^\top \left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right) g \\ &= \frac{4.4\cdot 10^6}{\phi^4}\left(\|\mathbf{B}_G h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2\right) \\ &\le 1.01\frac{4.4\cdot 10^6}{\phi^4}\left(\|\mathbf{M}h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2\right) < 5\cdot 10^6 t,\end{aligned}$$

which contradicts with the condition that $\|g^{\perp v}\|_2^2 \ge 5\cdot 10^6 t$. Consequently, we have proven by contradiction that Equation (16) holds.

Next we prove that the Rayleigh quotient $v^\top(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})v$ does not increase too much due to the vertex deletions that occur between two RESETs. Suppose the algorithm executes Line 8 when running QUERYHEAVY with vector v and matrix $\mathbf{L}_G$. Let $G^{\text{old}}, V^{\text{old}}, \mathbf{L}_G^{\text{old}}, \mathbf{D}^{\text{old}}, d^{\text{old}}$ and v^{old} be the $G, V, \mathbf{L}_G, \mathbf{D}, d$ and v at the last time when RESET was invoked, at which point

$$(v^{\text{old}})^\top((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I})v^{\text{old}} \le 2\lambda_1((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I})$$

since v^{old} was recomputed to be a 2-approximate least eigenvector by Theorem 4.10. Our goal is to show that:

$$v^\top\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v \le 3(v^{\text{old}})^\top\left((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v^{\text{old}}.$$

Let v^{ext} be a vector extended to the same length as v^{old}, sharing the same values as v^{old} on the support of V, and 0 otherwise. See Figure 2 for an illustration. Let $\lambda_i^{\text{old}} \stackrel{\text{def}}{=} \lambda_i((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2})$, and $v_i^{\text{old}} \stackrel{\text{def}}{=} v_i((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2})$ denote the true eigenvalues and eigenvectors. Since v^{old} is a constant factor approximation to v_1^{old} in the Rayleigh quotient, Theorem 4.8 implies that $1 - ((v_1^{\text{old}})^\top v^{\text{old}})^2 \le \frac{10\lambda_1^{\text{old}}}{\lambda_2^{\text{old}}}$. Theorem 4.3 and Theorem 4.4 imply that $\lambda_1^{\text{old}} \le 10\beta_\eta + \epsilon_{\text{add}}$, and that

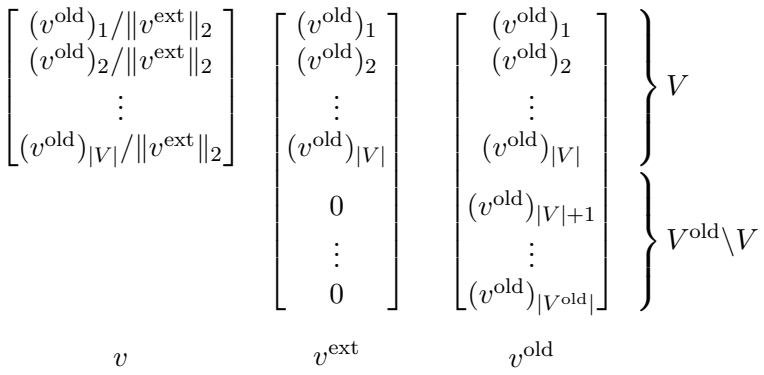

Figure 2: Illustration of v, v^{ext}, and v^{old}.

$\lambda_2^{\text{old}} \ge \frac{\phi^2}{4c}$ where $c = 9$. By our choices of parameters that $\beta_\eta, \epsilon_{\text{add}} \le \frac{\phi^2}{10^5\log^2(m)} \le \frac{\phi^2}{400c\log m}$, we have that $\frac{10\lambda_1^{\text{old}}}{\lambda_2^{\text{old}}} \le \frac{1}{\log m}$. Furthermore, since $(\beta_\eta + 9\lambda_1^{\text{old}})\log m \le \frac{\phi^2}{100\log m}$, by Theorem 5.2,

$$\frac{(v_{\min}^{\text{old}})_i}{\sqrt{(d^{\text{old}})_i}} \ge \left(1 - \frac{1}{\log m}\right)\cdot \max_{i'}\frac{(v_{\min}^{\text{old}})_{i'}}{\sqrt{(d^{\text{old}})_{i'}}}. \tag{17}$$

Next we bound $\|v^{\text{ext}}\|_2 = (v^{\text{old}})_i/(v)_i$, for any $i \in V$, the normalizing factor after deleting some subset of vertices. Let χ_V be the indicator vector that has support V^{old} and is 1 on V and 0 otherwise. By definition,

$$\begin{aligned}
\|v^{\text{ext}}\|_2^2 &= \|v^{\text{old}} \cdot \chi_V\|_2^2 \\
&\geq \frac{1}{2}\left\|(v_{\min}^{\text{old}}(v_{\min}^{\text{old}})^\top v^{\text{old}}) \cdot \chi_V\right\|_2^2 - \left\|(v^{\text{old}} - v_{\min}^{\text{old}}(v_{\min}^{\text{old}})^\top v^{\text{old}}) \cdot \chi_V\right\|_2^2 \\
&\geq \frac{1}{2}((v_{\min}^{\text{old}})^\top v^{\text{old}})^2 \cdot \|v_{\min}^{\text{old}} \cdot \chi_V\|_2^2 - \left(\|v^{\text{old}}\|_2^2 + \left((v_{\min}^{\text{old}})^\top v^{\text{old}}\right)^2 \|v_{\min}^{\text{old}}\|_2^2 - 2\left((v_{\min}^{\text{old}})^\top v^{\text{old}}\right)^2\right).
\end{aligned}$$

We bound the $(v_{\min}^{\text{old}})^\top v^{\text{old}}$ term using $1 - ((v_{\min}^{\text{old}})^\top v^{\text{old}})^2 \leq \frac{10\lambda_1^{\text{old}}}{\lambda_2^{\text{old}}} \leq \frac{1}{\log m}$ that we've proved, and we bound the $\|v^{\text{old}}\|_2$ term by $\|v^{\text{old}}\|_2 = \|v_{\min}^{\text{old}}\|_2 = 1$, so the above equation gives

$$\begin{aligned}
\|v^{\text{ext}}\|_2^2 &\geq \frac{1}{2}\left(1 - \frac{1}{\log m}\right) \cdot \|v_{\min}^{\text{old}} \cdot \chi_V\|_2^2 - \frac{1}{\log m} \\
&\geq \frac{1}{2}\left(1 - \frac{1}{\log m}\right) \cdot \frac{(1 - \frac{1}{\log m})\sum_{i \in V}(d^{\text{old}})_i}{\sum_{i \in V^{\text{old}}}(d^{\text{old}})_i} - \frac{1}{\log m} \\
&\geq \frac{1}{3}, \qquad (18)
\end{aligned}$$

where the second line holds by Equation (17), and the third line holds since if we ever delete more than half the edges we reset (see Line 21), giving us that $\sum_{i \in V}(d^{\text{old}})_i \geq \frac{1}{2}\sum_{i \in V^{\text{old}}}(d^{\text{old}})_i$.

The term that we want to bound is

$$\begin{aligned}
v^\top(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})v &= \epsilon_{\text{add}} + \sum_{e=(j,i) \in G}\left((d^{-1/2})_i(v)_i - \eta_e(d^{-1/2})_j(v)_j\right)^2 \\
&= \epsilon_{\text{add}} + \sum_{e=(j,i) \in G}\frac{1}{\|v^{\text{ext}}\|_2^2}\left((d^{-1/2})_i(v^{\text{old}})_i - \eta_e(d^{-1/2})_j(v^{\text{old}})_j\right)^2.
\end{aligned}$$

Using Equation (18), the above equation gives

$$\begin{aligned}
v^\top(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I})v &\leq 3\epsilon_{\text{add}} + 3\sum_{e=(j,i) \in G}\left((d^{-1/2})_i(v^{\text{old}})_i - \eta_e(d^{-1/2})_j(v^{\text{old}})_j\right)^2 \\
&\leq 3\epsilon_{\text{add}} + 3\sum_{e=(j,i) \in G^{\text{old}}}\left((d^{-1/2})_i(v^{\text{old}})_i - \eta_e(d^{-1/2})_j(v^{\text{old}})_j\right)^2 \\
&= 3(v^{\text{old}})^\top\left((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v^{\text{old}},
\end{aligned}$$

where the second line follows from G undergoing only edge deletions, and the last line follows from d and d^{old} are the same on all vertices $i \in V$.

Combining with (16) that $v^\top\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v \geq 10\lambda_1\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)$, and $(v^{\text{old}})^\top\left((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v^{\text{old}} \leq 2\lambda_1\left((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)$, yields

$$\begin{aligned}
10\lambda_1\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right) &\leq v^\top\left(\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v \\
&\leq 3(v^{\text{old}})^\top\left((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right)v^{\text{old}} \\
&\leq 6\lambda_1\left((\mathbf{D}^{\text{old}})^{-1/2}\mathbf{L}_G^{\text{old}}(\mathbf{D}^{\text{old}})^{-1/2} + \epsilon_{\text{add}}\mathbf{I}\right).
\end{aligned}$$

As such, we have that each time we call RESET in QUERYHEAVY, the smallest eigenvalue of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$ must drop by at least a factor of 1.5. Since the smallest eigenvalue of $\mathbf{D}^{-1/2}\mathbf{L}_G\mathbf{D}^{-1/2} + \epsilon_{\text{add}}\mathbf{I}$ is lower bounded by ϵ_{add}, and upper bounded by $O(1)$, we have that the smallest eigenvalue can drop for at most $O(\log(\epsilon_{\text{add}}^{-1}))$ times.

Correctness and time complexity of Delete and ScaleTau: The correctness of these two operations directly follow from their algorithm description.

In DELETE$(\cdot)$, updating $\mathbf{M}$ requires $O(|F|k) = O(|F|\log m)$ time, since each row of $\mathbf{B}_G$ is 2-sparse. Deleting the edges in F from G, u, and π_u takes $O(|F|)$ time. Finally, it's straightforward to see SCALETAU$(\cdot)$ takes $O(1)$ time.

Correctness of QueryHeavy: If there is an entry e of $\mathbf{B}_G h$ such that $|\mathbf{B}_G\mathbf{D}^{-1/2}g|_e \geq \epsilon$, then at least one of $|\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v}|_e$ and $|\mathbf{B}_G\mathbf{D}^{-1/2}g^{v}|_e$ will be at least $\epsilon/2$. As such, it suffices to find the heavy hitters of $\mathbf{B}_G\mathbf{D}^{-1/2}g^v$ and $\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v}$.

- Heavy hitters of $\mathbf{B}_G\mathbf{D}^{-1/2}g^v$: These are covered in line 9, which finds all the entries of $|\mathbf{B}_G\mathbf{D}^{-1/2}g^v| = \|g^v\|_2 \cdot |u|$ that are greater than $\epsilon/2$.
- Heavy hitters of $\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v}$: These are covered in line 11. If some $e = (i, j)$ satisfies that

$$\left| d_j^{-1/2} g_j^{\perp v} - \eta_e d_i^{-1/2} g_i^{\perp v} \right| = |\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v}|_e \geq \epsilon/2,$$

  then since $|\eta_e| \leq 1 + \beta_\eta \leq 1.5$, we have that either $|d_i^{-1/2} g_i^{\perp v}| \geq \epsilon/6$ or $|d_j^{-1/2} g_j^{\perp v}| \geq \epsilon/6$.

Time complexity of QueryHeavy: The runtime of QUERYHEAVY$(\cdot)$ has the following parts:

- Time to compute vectors g^v and $g^{\perp v}$, and time to go over all vertices in the for loop on Line 10, and these steps take $O(n)$ time in total.
- Time to perform binary search over π_u: $O(\log m)$ time.
- Time to add edges into set I: We bound the size of the set I. There are two places where we add edges to I. In line 9, we add all the heavy hitters of $\mathbf{B}_G\mathbf{D}^{-1/2}g^v$ to I. There can be at most $O(\|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2\epsilon^{-2})$ such heavy hitters. In line 11, we check to see if a vertex j satisfies $|d_j^{-1/2} g_j^{\perp v}| \geq \epsilon/6$, which is exactly when $d_j^{-1}(g_j^{\perp v})^2\epsilon^{-2} \geq 1/36$. For each of these vertices, we have to add d_j entries to I, so the total number of edges added to I is bounded by $O(\|g^{\perp v}\|_2^2\epsilon^{-2})$.

  We first bound the latter norm. By Johnson-Lindenstrauss lemma, $\|\mathbf{M}h\|_2 \approx_{1.01} \|\mathbf{B}_G h\|_2$, and we have also proved that $\mathbf{D} \approx_{30} \mathbf{D}_G$, so

$$\|g^{\perp v}\|_2^2 \leq O(t) \leq O\Big(\phi^{-4}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|\mathbf{D}_G^{1/2} h\|_2^2\Big). \tag{19}$$

  To bound the first norm, see that since $g = g^{\perp v} + g^v$,

$$\begin{aligned}
\|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2 &\leq 2\|\mathbf{B}_G\mathbf{D}^{-1/2}g\|_2^2 + 2\|\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v}\|_2^2 \\
&\leq 2\|\mathbf{B}_G h\|_2^2 + 2\|\mathbf{B}_G\mathbf{D}^{-1/2}\|_2^2\|g^{\perp v}\|_2^2 \\
&\leq 2\|\mathbf{B}_G h\|_2^2 + 8\|g^{\perp v}\|_2^2,
\end{aligned} \tag{20}$$

  where the last step makes use of Theorem 4.4, bounding $\|\mathbf{B}_G\mathbf{D}^{-1/2}\|_2^2$ by 4. Thus, this step takes $O(\phi^{-4}\epsilon^{-2}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\epsilon^{-2}\|\mathbf{D}_G^{1/2} h\|_2^2)$ time.

Adding these terms together, the total amortized time for QueryHeavy$(\cdot)$ is $O(\phi^{-4}\epsilon^{-2}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon^{-2}\epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2 + n + \log m)$.

Correctness and time complexity of Norm. The lower bound follows from

$$\begin{aligned}\|g^v\|_2^2 \cdot \|u\|_2^2 + \|g^{\perp v}\|_2^2 &= \|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2 + \|g^{\perp v}\|_2^2 \\ &\geq \|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2 + \frac{\|\mathbf{B}_G\mathbf{D}^{-1}g^{\perp v}\|_2^2}{4} \geq \frac{\|\mathbf{B}_G h\|_2^2}{8},\end{aligned}$$

where the second step follows from $\lambda_n(\mathbf{D}^{-1/2}\mathbf{B}_G^\top\mathbf{B}_G\mathbf{D}^{-1/2}) \leq 4$ by Theorem 4.4, the third step follows from $\|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2+\|\mathbf{B}_G\mathbf{D}^{-1}g^{\perp v}\|_2^2 \geq (\|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2+\|\mathbf{B}_G\mathbf{D}^{-1}g^{\perp v}\|_2)^2/2 \geq \|\mathbf{B}_G\mathbf{D}^{-1}g\|_2^2/2 = \|\mathbf{B}_G h\|_2^2/2$.

The upper bound follows from

$$\begin{aligned}\|g^v\|_2^2 \cdot \|u\|_2^2 + \|g^{\perp v}\|_2^2 &= \|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2 + \|g^{\perp v}\|_2^2 \\ &\leq 2\|\mathbf{B}_G h\|_2^2 + 9\|g^{\perp v}\|_2^2 \\ &\leq O\Big(\phi^{-4}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2\Big),\end{aligned}$$

where the second step follows from Eq. (20) that $\|\mathbf{B}_G\mathbf{D}^{-1/2}g^v\|_2^2 \leq 2\|\mathbf{B}_G h\|_2^2 + 8\|g^{\perp v}\|_2^2$, and the third step follows from Eq. (19) that $\|g^{\perp v}\|_2^2 \leq O(\phi^{-4}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2)$.

Finally we bound the time complexity of Norm$(\cdot)$. Note that the algorithm can maintain $\|u\|_2$ whenever it re-computes u, and computing $\|g^v\|_2$ and $\|g^{\perp v}\|_2$ takes $O(n)$ time.

Correctness of Sample. In the algorithm our goal is to sample each edge $e = (a_e, b_e)$ with probability p_e/S where $p_e = \min\{1,\ 5\overline{C}_1\big(\|g^v\|_2^2 u_e^2 + \frac{(g_{a_e}^{\perp v})^2}{d_{a_e}} + \frac{(g_{b_e}^{\perp v})^2}{d_{b_e}}\big) + \overline{C}_2 + C_3\overline{\tau}_e\}$ and $S = \sum_{e\in E} p_e$. First note that by the definitions of S_1, S_2, S_3, S on Line 8, the S computed by the algorithm satisfies $S = \sum_{e\in E} p_e$. We first prove that $5(\|g^v\|_2^2 u_e^2 + \frac{(g_{a_e}^{\perp v})^2}{d_{a_e}} + \frac{(g_{b_e}^{\perp v})^2}{d_{b_e}}) \geq (\mathbf{B}_G h)_e^2$, which will imply $p_e \geq \min\big\{1,\ \overline{C}_1 \cdot (\mathbf{B}_G h)_e^2 + \overline{C}_2 + C_3\overline{\tau}_e\big\}$ as required. By definitions $(\mathbf{B}_G h)_e^2 = (\mathbf{B}_G\mathbf{D}^{-1/2}g^v + \mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v})_e^2 \leq 2(\mathbf{B}_G\mathbf{D}^{-1/2}g^v)_e^2 + 2(\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v})_e^2$. Since $(\mathbf{B}_G\mathbf{D}^{-1/2}g^v)_e = \|g^v\|_2^2 u_e^2$ and $(\mathbf{B}_G\mathbf{D}^{-1/2}g^{\perp v})_e = \frac{g_{b_e}^{\perp v}}{d_{b_e}^{1/2}} - \frac{\eta_e g_{a_e}^{\perp v}}{d_{a_e}^{1/2}}$, we have

$$\begin{aligned}(\mathbf{B}_G h)_e^2 &\leq 2\|g^v\|_2^2 u_e^2 + 2(\frac{g_{b_e}^{\perp v}}{d_{b_e}^{1/2}} - \frac{\eta_e g_{a_e}^{\perp v}}{d_{a_e}^{1/2}})^2 \\ &\leq 2\|g^v\|_2^2 u_e^2 + 4\frac{(g_{b_e}^{\perp v})^2}{d_{b_e}} + 5\frac{(g_{a_e}^{\perp v})^2}{d_{a_e}},\end{aligned}$$

where we used that $\eta_e \leq 1 + \beta_\eta < 1.1$.

Next we prove that each edge is indeed sampled with probability p_e/S. For each $j \in [C_0 S]$, and for each edge e, we compute the probability that $x_j = p_e^{-1}\vec{1}_e$ by summing the probability of the three cases of Line 12, 14, and 16:

$$\begin{aligned}\Pr[x_j = p_e^{-1}\vec{1}_e] = \frac{S_1}{S}\cdot\frac{5\overline{C}_1\|g^v\|_2^2 u_e^2}{S_1} + \frac{S_3}{S}\cdot\frac{\overline{C}_2 + C_3\overline{\tau}_e}{S_3} \\ + \frac{S_2}{S}\cdot\Big(\frac{5\overline{C}_1(g_{a_e}^{\perp v})^2(d_{\overline{G}})_{a_e}}{d_{a_e}\cdot S_2}\cdot\frac{1}{(d_{\overline{G}})_{a_e}} + \frac{5\overline{C}_1(g_{b_e}^{\perp v})^2(d_{\overline{G}})_{b_e}}{d_{b_e}\cdot S_2}\cdot\frac{1}{(d_{\overline{G}})_{b_e}}\Big) = \frac{p_e}{S}.\end{aligned}$$

Time complexity of Sample. Sample$(\cdot)$ has $C_0 S$ iterations, where each iteration has the following parts:

- Sample each edge e with probability $\frac{5\overline{C}_1\|g^v\|_2^2 u_e^2}{S_1}$ on Line 12: Since the algorithm maintains the vector u, using a binary tree that stores partial sums of the probabilities, this sampling step can be implemented in time $O(\log m)$.

- Sample each edge e with probability $\frac{(g_{a_e}^{\perp v})^2}{d_{a_e}} + \frac{(g_{b_e}^{\perp v})^2}{d_{b_e}}$ on Line 14: The algorithm first samples a vertex i with probability $\frac{5\overline{C}_1(g_i^{\perp v})^2 \cdot (d_{\overline{G}})_i}{d_i \cdot S_2}$, which can be done in $O(\log n)$ time using a binary tree that stores partial sums of the probabilities. This binary tree can be precomputed in $O(n \log n)$ time and reused for all $C_0 S$ samples. The algorithm then uniformly samples an incident edge of i in $O(\log(d_{\overline{G}})_i)$ time. So the total amortized time of this step is $O(\log n + \frac{n \log n}{C_0 S})$.

- Sample each edge e with probability $\frac{\overline{C}_2 + C_3\overline{\tau}_e}{S_3}$ on Line 16: Since the algorithm maintains the vector τ, using a binary tree that stores partial sums of the probabilities, this sampling step can be implemented in time $O(\log m)$.

Combining these three parts, we have that sampling one edge takes $O(\log m + \frac{n \log n}{C_0 S})$ time. Next we bound S using Eq. (19) and (20):

$$\begin{aligned} S &= \sum_e \left(5\overline{C}_1\big(\|g^v\|_2^2 u_e^2 + \frac{(g_{a_e}^{\perp v})^2}{d_{a_e}} + \frac{(g_{b_e}^{\perp v})^2}{d_{b_e}}\big) + \overline{C}_2 + C_3\overline{\tau}_e \right) \\ &\le O\left(\overline{C}_1\big(\|g^v\|_2^2\|u\|_2^2 + \|g^{\perp v}\|_2^2\big) + \overline{C}_2 m + C_3\|\overline{\tau}\|_1\right) \\ &\le O\Big(\overline{C}_1\phi^{-4}\|\mathbf{B}_G h\|_2^2 + \overline{C}_1\phi^{-4}\epsilon_{\text{add}}\|\mathbf{D}^{1/2}h\|_2^2 + \overline{C}_2 m + C_3\|\overline{\tau}\|_1\Big). \end{aligned}$$

So the total time $C_0 S \cdot O(\log m + \frac{n \log n}{C_0 S})$ is bounded as claimed in the theorem statement. □

7 General Heavy Hitters and Sampler for Two-Sparse Matrices

In this section we present our general heavy hitter and sampler data structure for two-sparse matrices. In Section 8.1 we show that this data structure suffices to implement the HeavyHitter and HeavySampler data structures required by the reduction in [vdBLL+21]. While the reduction in [vdBLL+21] only requires the ability to rescale and delete rows, we prove a stronger data structure that supports row insertions and deletions.

We begin by stating our main theorem, Theorem 7.1 for two-sparse matrices. In the remaining sections we present the proof of Theorem 7.6 in a bottom-up fashion. In Section 7.1, we show that the heavy hitter data structure on balanced lossy expanders (Theorem 6.1) implies a heavy hitter data structure on all balanced lossy graphs (Theorem 7.2). In Section 7.2, we further show that the heavy hitter data structure on balanced lossy graphs (Theorem 7.2) implies a heavy hitter data structure on general lossy graphs (Theorem 7.6). Finally in Section 7.3, we prove our main theorem Theorem 7.1 using the heavy hitter data structure general lossy graphs (Theorem 7.6).

Additional notation. Since all data structures in this section support update operations — including row insertions and deletions, edge insertions and deletions, and vector entry updates — the matrix $\mathbf{A}$ and the lossy graph G can change over time, and as such all bounding parameters W_g, W_η, λ_1 are defined as their maximum values over all times when a data structure operation is called. Throughout this section, we use the superscript $^{(t)}$ to denote the object after the t-th

update, e.g., $\mathbf{A}^{(t)}$ denotes the matrix $\mathbf{A}$ after the t-th update. We omit this superscript when the object is clear from context, using it only when distinguishing between updates is necessary.

Theorem 7.1 (Heavy hitters and sampler on two-sparse matrices)**.** *Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ be a dynamic two-sparse matrix undergoing row insertions and deletions. Let λ_1 be the largest value such that $\|\mathbf{A}h\|_2 \geq \sqrt{\lambda_1}\|h\|_2$ at any time the algorithm calls a query operation with input h, and let $W_A \stackrel{\text{def}}{=} \max_t W_{\mathbf{A}^{(t)}}$ (Theorem 1.1). There is a data structure that w.h.p. supports the following operations:*

- INITIALIZE($\mathbf{A} \in \mathbb{R}^{m \times n}, \overline{\tau} \in \mathbb{R}^m_{\geq 0}$)*: Initializes with a two-sparse matrix $\mathbf{A}$ and a vector $\overline{\tau}$ in amortized $\widetilde{O}(m \log^2(\lambda_1^{-1}) \log^8(W_A))$ time.*
- INSERT($a \in \mathbb{R}^n$)*: Appends two-sparse row a to $\mathbf{A}$ in amortized $\widetilde{O}\left(\log^2(\lambda_1^{-1}) \log^6(W_A)\right)$ time.*
- DELETE($e \in [m]$)*: Deletes the row e in $\mathbf{A}$ in the same time as* INSERT*.*
- SCALETAU($e \in [m], b \in \mathbb{R}$)*: Sets $\overline{\tau}_e \leftarrow b$ in worst-case $O(1)$ time.*
- QUERYHEAVY($h \in \mathbb{R}^n, \epsilon \in (0,1)$)*: Returns $I \subseteq [m]$ containing exactly those i with $|(\mathbf{A}h)_i| \geq \epsilon$ in amortized $\widetilde{O}(\epsilon^{-2}\|\mathbf{A}h\|_2^2 + n \log^2(W_A))$ time.*
- SAMPLE($h \in \mathbb{R}^V, C_0, C_1, C_2, C_3$)*: Let a vector $p \in \mathbb{R}^E$ satisfy*

$$p_e \geq \min\left\{1,\ C_1 \frac{m}{\sqrt{n}} \cdot \frac{(\mathbf{A}h)_e^2}{\|\mathbf{A}h\|_2^2} + C_2 \frac{1}{\sqrt{n}} + C_3 \overline{\tau}_e\right\}.$$

Let $S = \sum_{e \in E} p_e$. Let X be a random variable which equals to $p_e^{-1}\vec{1}_e$ with probability p_e/S for all $e \in E$. This operation returns a random diagonal matrix $\mathbf{R} = C_0^{-1} \sum_{j=1}^{C_0 S} \mathbf{diag}(X_j)$, where X_j are i.i.d. copies of X. The amortized time of this operation and also the output size of $\mathbf{R}$ are bounded by

$$\widetilde{O}\left(C_0 C_1 \frac{m}{\sqrt{n}} + (C_0 C_2 \frac{m}{\sqrt{n}} + n) \cdot \log^2(W_A) + C_0 C_3 \|\overline{\tau}\|_1\right).$$

We note that $\lambda_1 = \lambda_1(\mathbf{A}^\top \mathbf{A})$ suffices for the above theorem to hold, and that we prove the slightly stronger statement only requiring the bound on the Rayleigh quotient of $\mathbf{A}^\top \mathbf{A}$ for the query vectors h, as opposed to all vectors.

7.1 Heavy Hitters on Balanced Lossy Graphs (Reduction to Balanced Lossy Expanders)

In this section we present a heavy hitter data structure for any β_η-balanced lossy graph, built using the data structure of Theorem 6.1. The reduction in this section decomposes the input graph into subgraphs and maintains the following three properties for each subgraphs, as required by Theorem 6.1: (1) Updates are only deletions. (2) The subgraphs are expanders. (3) The degrees of the subgraphs are approximately preserved under updates.

Theorem 7.2 (Heavy hitters on balanced lossy graphs)**.** *Let $\beta_\eta \leq 0.0005$. There is a data structure (Algorithm 3 and 4) that w.h.p. supports the following operations:*

- INITIALIZE($G = (V, E, \eta), \epsilon_{\text{add}} \in (0,1), \phi \in ((2000\beta_\eta)^{1/2}, 1), \overline{\tau} \in \mathbb{R}^m_{\geq 0}$)*: Initializes with a β_η-balanced lossy graph $G = (V, E, \eta)$ with $|V| = n$ and $|E| = m$, parameters ϵ_{add} and ϕ, and a vector $\overline{\tau} \geq 0$ in amortized $O(m\phi^{-2} \log^2(n/\epsilon_{\text{add}}))$ time.*

- DELETE($e \in E$): *Deletes the edge e from G in amortized $O(\phi^{-3}\log(m)\log^2(n/\epsilon_{\text{add}}))$ time.*
- INSERT($a_e \in V, b_e \in V, \eta_e \in [1, 1+\beta_\eta]$): *Inserts a new edge $e = (a_e, b_e)$ with multiplier η_e to G in amortized $O(\phi^{-2}\log(m)\log^2(n/\epsilon_{\text{add}}))$ time.*
- SCALETAU($e \in E, b \in \mathbb{R}_{\geq 0}$): *Sets $\overline{\tau}_e \leftarrow b$ in worst-case $O(1)$ time.*
- QUERYHEAVY($h \in \mathbb{R}^V, \epsilon \in (0,1)$): *Returns a set $I \subseteq E$ containing exactly all $e \in E$ that satisfies $|\mathbf{B}_G h|_e \geq \epsilon$ in amortized $O(\phi^{-4}\epsilon^{-2}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\epsilon^{-2}\|\mathbf{D}_G^{1/2}h\|_2^2 + n\log^2 m)$ time.*
- NORM($h \in \mathbb{R}^V$): *Returns a value L in amortized $O(n\log^2(m))$ time such that*

$$\|\mathbf{B}_G h\|_2^2 \leq L \leq O(\phi^{-4}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2).$$

- SAMPLE($h \in \mathbb{R}^V, C_0, \overline{C}_1, \overline{C}_2, C_3$): *Let a vector $p \in \mathbb{R}^E$ satisfy*

$$p_e \geq \min\left\{1,\ \overline{C_1}\cdot(\mathbf{B}_G h)_e^2 + \overline{C}_2 + C_3\overline{\tau}_e\right\}.$$

Let $S = \sum_{e\in E} p_e$. Let X be a random variable which equals to $p_e^{-1}\vec{1}_e$ with probability p_e/S for all $e \in E$. This operation returns a random diagonal matrix $\mathbf{R} = C_0^{-1}\sum_{j=1}^{C_0 S}\mathbf{diag}(X_j)$, where X_j are i.i.d. copies of X. The amortized time of this operation and also the output size of $\mathbf{R}$ are bounded by

$$O\left(C_0\overline{C}_1\phi^{-4}\log m(\|\mathbf{B}_G h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2) + C_0\overline{C}_2 m\log m + C_0 C_3\|\overline{\tau}\|_1\log m + n\log^3(m)\right).$$

Expander decomposition and pruning. The data structure of Theorem 7.2 uses the following tools of expander decomposition and expander pruning. We note that the expander decomposition and expander pruning results extend naturally to graphs with multi-edges, in the same way that they extend to weighted graphs in Section 4.1 of [SW19].

Lemma 7.3 (Expander decomposition, Theorem 5.1 of [CKL+22] and Theorem 1.2 of [SW19])**.** *There is an algorithm* DECOMPOSE(G) *that takes as input any unweighted, undirected graph G, and w.h.p. in $O(m\log^7(m))$ time computes an edge-disjoint partition of G into graphs $G_0, G_1, \cdots, G_k$ for $k = O(\log n)$ such that each nontrivial connected component X of G_i is a ϕ-expander for $\phi = \Theta(1/\log^3(m))$.*

Lemma 7.4 (Expander pruning, Theorem 1.3 of [SW19])**.** *Let $G = (V, E)$ be a ϕ-expander with m edges. There is a deterministic algorithm with access to adjacency lists of G such that, given an online sequence of $k \leq \phi m/10$ edge deletions in G, can maintain a pruned set $P \subseteq V$ such that the following property holds. Let G_i and P_i be the graph G and the set P after the i-th deletion. We have, for all i,*

1. *$P_0 = \emptyset$ and $P_i \subseteq P_{i+1}$,*
2. *$\text{vol}(P_i) \leq 8i/\phi$ and $|E(P_i, V - P_i)| \leq 4i$, and*
3. *$G_i\{V - P_i\}$ is a $\phi/6$-expander.*

The total time for updating $P_0, \cdots, P_k$ is $O(k\phi^{-2}\log(m))$.

Before proving Theorem 7.2, we first prove some invariants that the algorithm maintains.

Algorithm 3: Heavy hitter on β_η-balanced lossy graphs

Member : Parameters ϵ_{add}, ϕ, and $k_1, \cdots, k_{\lceil \log m \rceil}$.

Graph decomposition $G = \bigcup_{\ell=1}^{\lceil \log m \rceil} \bigcup_{j=1}^{k_\ell} G^{(\ell,j)}$, along with data structures $\text{DS}^{(\ell,j)}$ of Theorem 6.1, initial degrees $d^{(\ell,j)} \in \mathbb{R}^V$, initial number of edges $m_{\text{init}}^{(\ell,j)}$ and a counter $m_{\text{cnt}}^{(\ell,j)}$.

procedure `Initialize`$(G = (V, E, \eta), \epsilon_{\text{add}} \in (0,1), \phi \in ((2000\beta_\eta)^{1/2}, 1), \overline{\tau} \in \mathbb{R}^E_{\geq 0})$**:**

  $\epsilon_{\text{add}} \leftarrow \epsilon_{\text{add}}$, $\phi \leftarrow \phi$. $k_1 \leftarrow 0, \cdots, k_{\lceil \log m \rceil - 1} \leftarrow 0$, and $k_{\lceil \log m \rceil} \leftarrow 1$.

  Set $\overline{\tau} \leftarrow \overline{\tau}$, $G^{(\lceil \log m \rceil, 1)} \leftarrow G$, and call $\textsc{Rebuild}(\lceil \log m \rceil)$.

procedure `Delete`$(e_{\text{start}} \in E)$**:**

  Find $G^{(\ell,j)}$ that contains edge e_{start}, and set $F_{\text{tot}} \leftarrow \emptyset$, $F \leftarrow \{e_{\text{start}}\}$.

  while *F is not empty* **do**

    $F_{\text{tot}} \leftarrow F_{\text{tot}} \cup F$, increment $m_{\text{cnt}}^{(\ell,j)} \leftarrow m_{\text{cnt}}^{(\ell,j)} + |F|$

    $F_{\text{exp}}, F'_{\text{exp}} \leftarrow \textsc{ExpanderPrune}(F, \ell, j)$, delete edges in $F \cup F_{\text{exp}} \cup F'_{\text{exp}}$ from $G^{(\ell,j)}$

    $F_{\text{deg}} \leftarrow \textsc{DegreePrune}(F \cup F_{\text{exp}}, \ell, j)$, delete edges in F_{deg} from $G^{(\ell,j)}$

    $F \leftarrow F_{\text{deg}}$, $F_{\text{tot}} \leftarrow F_{\text{tot}} \cup F_{\text{exp}} \cup F'_{\text{exp}}$

  Call $\text{DS}^{(\ell,j)}.\textsc{Delete}(F_{\text{tot}})$. Call $\textsc{Insert}(a_e, b_e, \eta_e)$ for each edge $e \in F_{\text{tot}} \backslash \{e_{\text{start}}\}$.

  if $m_{\text{cnt}}^{(\ell,j)} \geq (\phi/10) m_{\text{init}}^{(\ell,j)}$ **then**

    Call $\textsc{Insert}(a_e, b_e, \eta_e)$ for all edges $e \in E(G^{(\ell,j)})$, and destruct $G^{(\ell,j)}$ and $\text{DS}^{(\ell,j)}$.

procedure `Insert`$(a_e \in V, b_e \in V, \eta_e \in [1, 1+\beta_\eta])$**:**

  $k_1 \leftarrow k_1 + 1$, and let $G^{(1,k_1)}$ contain the single edge $e = (a_e, b_e)$ with multiplier η_e.

  for $\ell \in [\lceil \log m \rceil]$ **do**

    if *total number of edges in level ℓ is at most 2^ℓ* **then**

      Call $\textsc{Rebuild}(\ell)$, and **break**

    else

      $G^{(\ell+1, j+k_{\ell+1})} \leftarrow G^{(\ell,j)}$ for all $j \in [k_\ell]$

      $k_{\ell+1} \leftarrow k_{\ell+1} + k_\ell$, $k_\ell \leftarrow 0$

procedure `Rebuild`$(\ell \in [\lceil \log m \rceil])$**:**

  Let $G^{(\ell)}$ be the union of all graphs of level ℓ.

  Use Theorem 7.3 to decompose the smoothed graph $\overline{G}^\ell$ of G^ℓ into edge-disjoint 10ϕ-expanders, and let $G^{(\ell,1)}, \cdots, G^{(\ell,k_\ell)}$ be the corresponding lossy graphs.

  for $j \in [k_\ell]$ **do**

    Initialize a data structure of Theorem 6.1: $\text{DS}^{(\ell,j)}.\textsc{Initialize}(G^{(\ell,j)}, \epsilon_{\text{add}}, \overline{\tau}_{E(G^{(\ell,j)})})$.

    Let $d^{(\ell,j)} \in \mathbb{R}^V$ be the degree of $G^{(\ell,j)}$, $m_{\text{init}}^{(\ell,j)} \leftarrow |E(G^{(\ell,j)})|$, $m_{\text{cnt}}^{(\ell,j)} \leftarrow 0$.

procedure `ExpanderPrune`$(F \subset E, \ell, j)$**:**

  Use Theorem 7.4 to find the set of vertices S to be pruned from $G^{(\ell,j)}$ so that it remains a ϕ-expander after deleting vertices in F.

  return $F_{\text{exp}} \leftarrow E_{G^{(\ell,j)}}(S, V \backslash S)$ and $F'_{\text{exp}} \leftarrow E_{G^{(\ell,j)}}(S, S)$

procedure `DegreePrune`$(F \subset E, \ell, j)$**:**

  $F_{\text{deg}} \leftarrow \emptyset$.

  For $e \in F$, compute the degree d'_{a_e}, d'_{b_e} of the two endpoints a_e, b_e in $G^{(\ell,j)}$.

  For $e \in F$, for $v \in \{a_e, b_e\}$, if $d'_v < d_v^{(\ell,j)}/9$ then add to F_{deg} all edges adjacent to v.

  return F_{deg}

Algorithm 4: Heavy hitter on β_η-balanced lossy graphs (continued from Algorithm 3)

procedure `ScaleTau`$(e \in E, b \in \mathbb{R}_{\geq 0})$**:**
  $\overline{\tau}_e \leftarrow b$
  Find the subgraph $G^{(\ell,j)}$ that contains edge e, and call $\mathrm{DS}^{(\ell,j)}$.ScaleTau(e, b).
procedure `QueryHeavy`$(h \in \mathbb{R}^V, \epsilon \in (0,1))$**:**
  for $\ell \in [\lceil \log m \rceil]$ *and* $j \in [k_\ell]$ **do** $I^{(\ell,j)} \leftarrow \mathrm{DS}^{(\ell,j)}$.QueryHeavy$(h, \epsilon)$
  return $I \leftarrow \bigcup_{\ell=1}^{\lceil \log m \rceil} \bigcup_{j=1}^{k_\ell} I^{(\ell,j)}$
procedure `Norm`$(h \in \mathbb{R}^V)$**:**
  for $\ell \in [\lceil \log m \rceil]$ *and* $j \in [k_\ell]$ **do** $L^{(\ell,j)} \leftarrow \mathrm{DS}^{(\ell,j)}$.Norm$(h, \epsilon)$
  return $L \leftarrow \sum_{\ell=1}^{\lceil \log m \rceil} \sum_{j=1}^{k_\ell} L^{(\ell,j)}$
procedure `Sample`$(h \in \mathbb{R}^V, \overline{C}_1, \overline{C}_2, C_3)$**:**
  for $\ell \in [\lceil \log m \rceil]$ *and* $j \in [k_\ell]$ **do**
    $\mathbf{R}^{(\ell,j)} \leftarrow \mathrm{DS}^{(\ell,j)}$.Sample$(h, \overline{C}_1, \overline{C}_2, C_3)$
  return $\mathbf{R}$ which is a concatenation of all $\mathbf{R}^{(\ell,j)}$

Lemma 7.5 (Invariants of Algorithm 3 and 4)**.** *Let $\beta_\eta \leq 0.0005$. Assuming the input graph G to the data structure of Algorithm 3 and 4 is a β_η-balanced lossy graph, and the input parameter $\phi \geq (2000\beta_\eta)^{1/2}$, then the data structure maintains the following invariants after any operation:*

1. **(Edge-disjoint decomposition)** *Each data structure $\mathrm{DS}^{(\ell,j)}$ maintains a subgraph $G^{(\ell,j)}$ that is edge disjoint with each other, and $\bigcup_{\ell=1}^{\lceil \log m \rceil} \bigcup_{j=1}^{k_\ell} E(G^{(\ell,j)}) = E(G)$, where G is maintained by the data structure such that each* Delete(e) *operation deletes the edge e from G, and each* Insert(a_e, b_e, η_e) *operation inserts a new edge (a_e, b_e) with multiplier η_e to G.*
2. **(Size of each level)** *For any $\ell \in [\lceil \log m \rceil]$, the subgraphs of level ℓ satisfy $\sum_{j \in [k_\ell]} |E(G^{(\ell,j)})| \leq 2^\ell$, and $\sum_{j \in [k_\ell]} |V(G^{(\ell,j)})| \leq O(n \log n)$.*
3. **(Properties of subgraphs)** *$\forall \ell \in [\lceil \log m \rceil]$ and $\forall j \in [k_\ell]$, the subgraph $G^{(\ell,j)}$ is a β_η-balanced lossy ϕ-expander. Furthermore, after any $\mathrm{DS}^{(\ell,j)}$.Delete operation, the degree of any vertex v in $G^{(\ell,j)}$ either remains at least $1/9$ of the original degree $d_v^{(\ell,j)}$, or it drops to 0.*

Proof. **Part 1 (Edge-disjoint decomposition).** First note that by expander decomposition (Theorem 7.3), after each Rebuild(ℓ) operation we are guaranteed that every edge of level ℓ is included in one of the edge-disjoint expanders $G^{(\ell,1)}, \cdots, G^{(\ell,k_\ell)}$, and each data structure $\mathrm{DS}^{(\ell,j)}$ maintains an expander $G^{(\ell,j)}$.

Initialize$(\cdot)$ first sets the initial graph G to have level $\lceil \log m \rceil$, and it then calls Rebuild$(\lceil \log m \rceil)$. This Rebuild$(\lceil \log m \rceil)$ ensures that initially $\bigcup_{\ell=1}^{\lceil \log m \rceil} \bigcup_{j=1}^{k_\ell} E(G^{(\ell,j)}) = E(G)$, and all the subgraphs are edge disjoint. Next we prove that we maintain an edge-disjoint decomposition after insertions and deletions.

Each Insert$(\cdot)$ inserts a new edge $e = (a_e, b_e)$ into a new subgraph $G^{(1,k_1)}$ in level 1, so we still have $\bigcup_{\ell=1}^{\lceil \log m \rceil} \bigcup_{j=1}^{k_\ell} E(G^{(\ell,j)}) = E(G)$, this operation then calls Rebuild$(\cdot)$ recursively that still maintain an edge-disjoint decomposition.

Each Delete(e) inserts all the deleted edges except e back to the data structure by using Insert$(\cdot)$. So we still have $\bigcup_{\ell=1}^{\lceil \log m \rceil} \bigcup_{j=1}^{k_\ell} E(G^{(\ell,j)}) = E(G)$.

Part 2 (Size of each level). We first prove the total number of edges in level ℓ is at most 2^ℓ. This bound holds because the algorithm could only add edges to level ℓ during INSERT$(\cdot)$, and it would move all edges from level ℓ to $\ell+1$ if the total number of edges in level ℓ exceeds 2^ℓ (see the if-else-clause from Line 17 to 21 of Algorithm 3).

Next we bound the number of vertices in level ℓ. Note that after each REBUILD(ℓ), Theorem 7.3 implies that $\sum_{j=1}^{k_\ell} |V(G^{(\ell,j)})| \le O(n \log n)$. The algorithm could only add more subgraphs to level ℓ in INSERT$(\cdot)$, and there are two cases: (1) either the total number of edges in level ℓ is bounded by 2^ℓ, and the algorithm rebuilds level ℓ, we again restore $\sum_{j=1}^{k_\ell} |V(G^{(\ell,j)})| \le O(n \log n)$, (2) or the total number of edges in level ℓ exceeds 2^ℓ, and the algorithm moves all graphs in level ℓ to $\ell+1$, and this level becomes empty.

Part 3 (Properties of subgraphs). Every subgraph $G^{(\ell,j)}$ is a β_η-balanced lossy graph because the initial graph is β_η-balanced, and any added edge satisfies $\eta_e \in [1, 1+\beta_\eta]$.

Every subgraph $G^{(\ell,j)}$ remains a ϕ-expander throughout the algorithm because the subgraphs are constructed to be 10ϕ-expanders in REBUILD by Theorem 7.3, and whenever the algorithm deletes an edge from a subgraph in DELETE, we immediately use the pruning procedure of Theorem 7.4 to ensure that it remains a ϕ-expander (see Line 8 and 28). Moreover, the algorithm uses the if-clause on Line 12 to ensure that at most $\phi/10$ fraction of edges are pruned using Theorem 7.4, so the pruning procedure is correct. Finally note that we never add any edge to a subgraph.

Finally note that the DEGREEPRUNE procedure of the algorithm ensures that at the end of each while-loop in DELETE (Line 10), only the vertices that are endpoints of edges in F may have dropped below $1/9$, so when the DELETE operation ends, we must have that the degree of any vertex v in $G^{(\ell,j)}$ either remains at least $1/9$ of the original degree or directly drops to 0. □

Using these invariants, we are ready to prove the main result of this section.

Proof of Theorem 7.2. First note that by the invariants proved in Part 3 of Theorem 7.5, all the requirements of Theorem 6.1 are satisfied by the subgraphs $G^{(\ell,j)}$, so all the function calls to data structures $\mathrm{DS}^{(\ell,j)}$ are correct. The invariants also ensure the correctness of DELETE$(\cdot)$, INSERT$(\cdot)$, and REBUILD$(\cdot)$. Also note that the correctness of SCALETAU$(\cdot)$ is straightforward, and its runtime is bounded by $O(n)$ by Theorem 6.1. It remains to bound the time complexity of all other operations, and prove the correctness of QUERYHEAVY$(\cdot)$, NORM$(\cdot)$, and SAMPLE$(\cdot)$.

Time complexity of Rebuild and Initialize. By Part 2 of Theorem 7.5, $\sum_{j=1}^{k_\ell} |E(G^{(\ell,j)})| \le 2^\ell$, so the expander decomposition algorithm of Theorem 7.3 runs in $O(2^\ell \log^7 m)$ time. Initialization of the data structures $\mathrm{DS}^{(\ell,1)}, \cdots, \mathrm{DS}^{(\ell,k_\ell)}$ of Theorem 6.1 takes $O(2^\ell \phi^{-2} \log^2(n/\epsilon_{\mathrm{add}}))$ time. Since INITIALIZE calls REBUILD$(\lceil \log m \rceil)$, it takes $O(m\phi^{-2} \log^2(n/\epsilon_{\mathrm{add}}))$ time.

Amortized time complexity of Insert. Whenever we perform INSERT$(\cdot)$, we first assign $O\big(\log(m)\phi^{-2} \log^2(n/\epsilon_{\mathrm{add}})\big)$ number of tokens to the edge e that is being inserted. Next we show that when INSERT$(\cdot)$ makes recursive calls to REBUILD$(\cdot)$, it has enough tokens to cover their cost. Whenever we call REBUILD(ℓ) during INSERT, there must have been at least $2^{\ell-1}$ number of edges that just got moved up to level ℓ from level $\ell-1$, and we charge $O(\phi^{-2} \log^2(n/\epsilon_{\mathrm{add}}))$ number of tokens from every such edge. In this way we collect enough tokens to cover the time complexity of REBUILD(ℓ).

Note that every single edge that is moved up from some level $\ell-1$ to level ℓ must have been inserted by some INSERT$(\cdot)$ since the initial edges are all in level $\lceil \log m \rceil$, and each edge can move up at most $\lceil \log m \rceil$ levels, so every edge has enough tokens to pay the charges of REBUILD$(\cdot)$.

Amortized time complexity of Delete. We say that DELETE(e_{start}) is performed on a subgraph $G^{(\ell,j)}$ if $e_{\mathrm{start}} \in E(G^{(\ell,j)})$. The algorithm maintains a counter $m_{\mathrm{cnt}}^{(\ell,j)}$ for each subgraph $G^{(\ell,j)}$, which records the total size of all sets F passed as inputs to EXPANDERPRUNE(F, ℓ, j) (see

Line 7 and 8). We first prove the following key amortization claim: for any $G^{(\ell,j)}$ and any integer t, after t number of DELETE$(\cdot)$ performed on $G^{(\ell,j)}$,

$$m_{\text{cnt}}^{(\ell,j)} \leq 7t. \tag{21}$$

We prove this claim by a token-based amortization argument. We maintain a pool of tokens that pays for each increment of $m_{\text{cnt}}^{(\ell,j)}$ on Line 7. Every time DELETE(e_{start}) is called on an edge $e_{\text{start}} \in E(G^{(\ell,j)})$, we assign 7 tokens to e_{start}. We maintain the invariant that at the beginning of each while-loop with set F (Line 6), every edge in F has at least 7 tokens. We also maintain the invariant that for any vertex that remains in the graph, it has 1 token for each edge deleted from it. These invariants holds initially, since the first while-loop starts with $F = \{e_{\text{start}}\}$. In each iteration of the while-loop (Line 6), tokens are transferred among edges and vertices as follows:

- *Counter increment step.* When $m_{\text{cnt}}^{(\ell,j)}$ is incremented by $|F|$ on Line 7, each edge in F pays 1 token for this increment.
- *Expander pruning step.* In each call to EXPANDERPRUNE(F, ℓ, j) on Line 8 (defined on Line 28), each edge in F pays 6 tokens. Among these, 2 tokens are given to each of the two endpoints of each edge in F, and for every cut edge $e = (u, v) \in E(S, V\backslash S)$ with $u \in S$ and $v \in V\backslash S$, 1 token is given to the vertex v, restoring the second invariant.

  By Theorem 7.4, $|E(S, V\backslash S)|$ is at most 4 times the total number of edges passed as inputs to the expander pruning procedure of Theorem 7.4, so $6 = 2 + 4$ tokens per edge is enough to pay for this step.
- *Degree pruning step.* In each call to DEGREEPRUNE(F, ℓ, j) on Line 9 (defined on Line 31), each vertex v detected on Line 34 with $d'_v < d_v^{(\ell,j)}/9$ pays $8d'_v$ tokens. Among these, for every edge $e = (u, v)$ adjacent to v that is added to F_{deg}, 7 tokens are given to e, and 1 token is given to the other endpoint u of e.

  Since $d'_v < d_v^{(\ell,j)}/9$, and each deleted edge adjacent to v gives 1 token to v, v has accumulated $d_v^{(\ell,j)} - d'_v > (8/9)d_v^{(\ell,j)} > 8d'_v$ number of tokens, so v has enough tokens to pay for this step.

Finally, since each edge in F_{deg} receives 7 tokens, and the next iteration of the while-loop begins with $F \leftarrow F_{\text{deg}}$, the invariant that each edge in F has at least 7 tokens is preserved. This finishes the proof of (21).

Next we use (21) to bound the amortized time complexity of DELETE$(\cdot)$. Consider a fixed DELETE(e_{start}) performed on $G^{(\ell,j)}$, and let $\Delta m_{\text{cnt}}^{(\ell,j)}$ denote the increase in $m_{\text{cnt}}^{(\ell,j)}$ during this operation. In each while-loop with set F (Line 6), $m_{\text{cnt}}^{(\ell,j)}$ is incremented by $|F|$, and the edges in F, F_{exp}, and F'_{exp} are added to F_{tot}. By Theorem 7.4, we have $|F_{\text{exp}} \cup F'_{\text{exp}}| \leq 8|F|/\phi$, so when all while-loops terminate and Line 11 is reached,

$$|F_{\text{tot}}| \leq (1 + 8/\phi) \cdot \Delta m_{\text{cnt}}^{(\ell,j)}.$$

Apart from the if-clause on Line 12 (which is called at most once for each $G^{(\ell,j)}$), the most time-consuming step of this DELETE(e_{start}) are the following two steps: (1) expander pruning steps of Line 8, (2) the calls to DS$^{(\ell,j)}$.DELETE(F_{tot}) and INSERT$(\cdot)$ on Line 11. So by Theorem 7.4, Theorem 6.1 and the time complexity of INSERT$(\cdot)$ proved earlier, the worst-case runtime of this DELETE(e_{start}) is

$$\Delta m_{\text{cnt}}^{(\ell,j)} \cdot \phi^{-2} \log(m) + |F_{\text{tot}}| \cdot O\left(\log(m)\phi^{-2} \log^2(n/\epsilon_{\text{add}})\right) \leq \Delta m_{\text{cnt}}^{(\ell,j)} \cdot O\left(\log(m)\phi^{-3} \log^2(n/\epsilon_{\text{add}})\right).$$

Combining the above equation and Equation (21), we have that apart from the if-clause on Line 12, the amortized time of each DELETE$(\cdot)$ is $O\left(\log(m)\phi^{-3}\log^2(n/\epsilon_{\text{add}})\right)$. Finally, since the if-clause on Line 12 is executed only when $m_{\text{cnt}}^{(\ell,j)} \geq (\phi/10)m_{\text{init}}^{(\ell,j)}$, by Equation (21) this if-clause is only executed after $O(\phi m_{\text{init}}^{(\ell,j)})$ number of DELETE$(\cdot)$ performed on $G^{(\ell,j)}$. Since the if-clause on Line 12 inserts at most $m_{\text{init}}^{(\ell,j)}$ number of edges, the amortized cost of this step is also $O(1/\phi)\cdot O\left(\log(m)\phi^{-2}\log^2(n/\epsilon_{\text{add}})\right) = O\left(\log(m)\phi^{-3}\log^2(n/\epsilon_{\text{add}})\right)$.

Correctness and time complexity of QueryHeavy, Norm, and Sample. By Part 1 of Theorem 7.5, the algorithm maintains $\bigcup_{\ell=1}^{\lceil\log m\rceil}\bigcup_{j=1}^{k_\ell} E(G^{(\ell,j)}) = E(G)$, so the correctness of these three operations directly follows from the correctness of the three operations of $\text{DS}^{(\ell,j)}$ by Theorem 6.1.

Using Theorem 6.1, the runtime of QUERYHEAVY$(\cdot)$ is

$$\begin{aligned}&\sum_{\ell=1}^{\lceil\log m\rceil}\sum_{j=1}^{k_\ell} O\left(\phi^{-4}\epsilon^{-2}\|\mathbf{B}^{(\ell,j)}h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\epsilon^{-2}\|(\mathbf{D}^{(\ell,j)})^{1/2}h\|_2^2 + |V(G^{(\ell,j)})|\right)\\ \leq\ & O\left(\phi^{-4}\epsilon^{-2}\|\mathbf{B}_G h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\epsilon^{-2}\|\mathbf{D}^{1/2}h\|_2^2 + n\log^2 m\right),\end{aligned}$$

where we bound the first term by Part 1 of Theorem 7.5 that $\{G^{(\ell,j)}\}$ forms an edge-disjoint decomposition of G, and we bound the second term by $\sum_{\ell=1}^{\lceil\log m\rceil}\sum_{j=1}^{k_\ell}\mathbf{D}^{(\ell,j)} = \mathbf{D}$ because this decomposition is edge disjoint, and we bound the third term by Part 2 of Theorem 7.5 that $\sum_{j=1}^{k_\ell}|V(G^{(\ell,j)})| \leq n\log n$.

Using Theorem 6.1, the runtime of NORM$(\cdot)$ is

$$\sum_{\ell=1}^{\lceil\log m\rceil}\sum_{j=1}^{k_\ell} O\left(|V(G^{(\ell,j)})|\right) \leq O(n\log^2(m)),$$

which follows from Part 2 of Theorem 7.5 that $\sum_{j=1}^{k_\ell}|V(G^{(\ell,j)})| \leq n\log n$.

Using Theorem 6.1, and denote $m^{(\ell,j)} = |E(G^{(\ell,j)})|$, $n^{(\ell,j)} = |V(G^{(\ell,j)})|$, and $\overline{\tau}^{(\ell,j)} = \overline{\tau}_{E(G^{(\ell,j)})}$, the runtime of SAMPLE$(\cdot)$ is

$$\begin{aligned}&\sum_{\ell=1}^{\lceil\log m\rceil}\sum_{j=1}^{k_\ell} O\left((\overline{C}_1(\phi^{-4}\|\mathbf{B}^{(\ell,j)}h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|(\mathbf{D}^{(\ell,j)})^{1/2}h\|_2^2) + \overline{C}_2 m^{(\ell,j)} + C_3\|\overline{\tau}^{(\ell,j)}\|_1)C_0\log m + n^{(\ell,j)}\log n\right)\\ \leq\ & O\left(C_0\overline{C}_1\phi^{-4}\log m(\|\mathbf{B}_G h\|_2^2 + \epsilon_{\text{add}}\|\mathbf{D}_G^{1/2}h\|_2^2) + C_0\overline{C}_2 m\log m + C_0C_3\|\overline{\tau}\|_1\log m + n\log^3 m\right),\end{aligned}$$

where we again used that $\{G^{(\ell,j)}\}$ forms an edge-disjoint decomposition of G, so $\sum_{\ell=1}^{\lceil\log m\rceil}\sum_{j=1}^{k_\ell}\mathbf{D}^{(\ell,j)} = \mathbf{D}$, and by Part 2 of Theorem 7.5 that $\sum_{j=1}^{k_\ell}|V(G^{(\ell,j)})| \leq n\log n$. □

7.2 Heavy Hitters on General Lossy Graphs (Reduction to Balanced Lossy Graphs)

In this section we present a heavy hitter data structure for general weighted lossy graphs (Theorem 7.6), built using the data structure of Theorem 7.2. The reduction in this section ensures two properties that are required by Theorem 7.2: (1) We decompose a general lossy graph to β_η-balanced lossy graphs. (2) We decompose the weighted graph into unweighted graphs, and implement SCALE$(\cdot)$ using INSERT$(\cdot)$ and DELETE$(\cdot)$.

Theorem 7.6 (Heavy hitters and sampler on general lossy graphs)**.** *Let $G = (V, E, \eta)$ be a dynamic lossy graph undergoing edge insertions and deletions, and let $\mathbf{A}$ denote its incidence matrix. Let $g \in \mathbb{R}^E_{>0}$ be a positive weight vector. Let $m \stackrel{\text{def}}{=} \max_t |E^{(t)}|$, and $n \stackrel{\text{def}}{=} \max_t |V^{(t)}|$. Let $W_\eta \stackrel{\text{def}}{=} \max_{e,t} \eta_e^{(t)}$, and let $W_g \stackrel{\text{def}}{=} \max_t \frac{\max_e g_e^{(t)}}{\min_e g_e^{(t)}}$. Let λ_1 be the largest value such that $\|\mathbf{A}h\|_2 \geq \sqrt{\lambda_1}\|h\|_2$ at any time the algorithm calls a query operation with input h. There is a data structure that w.h.p. supports the following operations:*

- INITIALIZE($\mathbf{A} \in \mathbb{R}^{E\times V}, g \in \mathbb{R}^E_{>0}, \overline{\tau} \in \mathbb{R}^E$)*: Initializes with the incidence matrix $\mathbf{A}$ of a lossy graph $G = (V, E, \eta)$, and two vectors g and $\overline{\tau}$ in amortized time*

$$\widetilde{O}\Big(m \log^2(\lambda_1^{-1}) \log^3(W_\eta) \log^3(W_g)\Big).$$

- INSERT($i \in V, j \in V, \eta \in \mathbb{R}, b \in \mathbb{R}$)*: Appends $\mathbf{A}$ with the row $\vec{1}_i - (1+\eta)\vec{1}_j$, and appends a corresponding entry of b to g in amortized $\widetilde{O}\left(\log^2(\lambda_1^{-1}) \log^2(W_\eta) \log^2(W_g)\right)$ time.*
- DELETE($e \in E$)*: Deletes the row e in both g and $\mathbf{A}$ in the same time as* INSERT*.*
- SCALE($e \in E, b \in \mathbb{R}$)*: Sets $g_e \leftarrow b$ in the same time as* INSERT*.*
- SCALETAU($e \in E, b \in \mathbb{R}$)*: Sets $\overline{\tau}_e \leftarrow b$ in worst-case $O(1)$ time.*
- QUERYHEAVY($h \in \mathbb{R}^V, \epsilon \in (0, 1)$)*: Returns $I \subset E$ containing exactly those e with $|(\mathbf{GA}h)_e| \geq \epsilon$ in amortized $O(\epsilon^{-2}\|\mathbf{GA}h\|_2^2) + \widetilde{O}(n \log(W_\eta) \log(W_g))$ time.*
- SAMPLE($h \in \mathbb{R}^V, C_0, C_1, C_2, C_3$)*: Let a vector $p \in \mathbb{R}^E$ satisfy*

$$p_e \geq \min\left\{1,\ C_1 \frac{|E|}{\sqrt{|V|}} \cdot \frac{(\mathbf{GA}h)_e^2}{\|\mathbf{GA}h\|_2^2} + C_2 \frac{1}{\sqrt{|V|}} + C_3 \overline{\tau}_e\right\}.$$

Let $S = \sum_{e\in E} p_e$. Let X be a random variable which equals to $p_e^{-1}\vec{1}_e$ with probability p_e/S for all $e \in E$. This operation returns a random diagonal matrix $\mathbf{R} = C_0^{-1} \sum_{j=1}^{C_0 S} \mathbf{diag}(X_j)$, where X_j are i.i.d. copies of X. The amortized time of this operation and also the output size of $\mathbf{R}$ are bounded by

$$\widetilde{O}\Big(C_0C_1 \frac{m}{\sqrt{n}} + (C_0C_2 \frac{m}{\sqrt{n}} + n) \cdot \log(W_\eta) \log(W_g) + C_0C_3\|\overline{\tau}\|_1\Big).$$

Before proving Theorem 7.6, we first prove the following key lemma which provides a procedure for decomposing any general lossy graph into β_η-balanced lossy graphs. This lemma first decomposes a general lossy graph into bipartite graphs whose edges are all oriented in the same direction, then partitions the edges in each bipartite graph by their flow multipliers η_e, and finally finds an appropriate vertex scaling factor to ensure all flow multipliers η_e are bounded by $1 + \beta_\eta$.

Lemma 7.7 (Decomposition into balanced lossy graphs)**.** *Consider any vertex set V and any $W_\eta > 1, \beta_\eta > 0$. There exists a map $f : \{(u, v, \eta_e) \mid u, v \in V, \eta_e \leq 1 + W_\eta\} \to [\lceil \log n \rceil \cdot 2\lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil]$ mapping edges into buckets, as well as a diagonal vertex scaling matrix $\mathbf{S}^{(i)} \in \mathbb{R}^{V\times V}_{>0}$ for each bucket $i \in [\lceil \log n \rceil \cdot 2\lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil]$ such that for any lossy graph $G = (V, E, \eta)$ with $\eta_e \leq 1 + W_\eta$, we have:*

1. *Let $E^{(i)} \subseteq E$ denote the edges mapped into i by f. Let $\mathbf{B}^{(i)}$ denote the submatrix of $\mathbf{B}_G$ with rows in $E^{(i)}$. Then the matrix $\mathbf{B}^{(i)} \cdot \mathbf{S}^{(i)}$ is the incidence matrix of a β_η-balanced lossy graph.*

2. $\forall i, k, W_\eta^{-1} \le \mathbf{S}^{(i)}_{k,k} \le 1$.

3. *f is computable in $O(\log n)$ time.*

Proof. For clarity, we construct this mapping in two steps, mapping each edge of form (u, v, η_e) where $u, v \in V$ and $\eta_e \le 1 + W_\eta$ into a bucket $[\lceil \log n \rceil] \times [2\lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil]$. First, label each vertex by an integer from 1 to n. For an edge $e = (u, v)$, let i be the first index in which u and v differ in their binary representations, and assign this edge to the i-th bucket $E^{(i)}$, consisting of the set of edges that would map to this bucket, i.e., the preimage of i in E.

Note that edges mapped to $E^{(i)}$ are bipartite, since edges in $E^{(i)}$ are only between vertices whose i-th bit differs. Let us consider this bipartitioning. Let $L^{(i)}$ be the set of vertices whose i-th bit are 0, and let $R^{(i)}$ be the set of vertices whose i-th bit are 1. Denote the edges in $E^{(i)}$ that are directed from $L^{(i)}$ to $R^{(i)}$ as $E_{L^{(i)} \to R^{(i)}}$, and $R^{(i)}$ to $L^{(i)}$ as $E_{R^{(i)} \to L^{(i)}}$. For $j = 0, 1, \cdots, \lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil$, define the edge sets:

$$E^{(i,j)} \stackrel{\text{def}}{=} \left\{ e \in E_{L^{(i)} \to R^{(i)}} : \eta_e \in [(1+\beta_\eta)^j, (1+\beta_\eta)^{j+1}) \right\}.$$

This partitions $E^{(i)}$, and forms the map f for edges going from $L^{(i)}$ to $R^{(i)}$. More precisely, for any edge $e = (u, v, \eta_e)$, we map this edge to (i, j), where i is the first bit that differs in the bit representations of u and v, and j is $\lfloor \frac{\log(1+\eta_e)}{\log(1+\beta_\eta)} \rfloor$ if the edge u has 0 as its i-th bit.

Next, define the scaling matrix $\mathbf{S}^{(i,j)}$ as follows: for all $v \in R^{(i)}$ set $\mathbf{S}^{(i,j)}_{v,v} = 1$, and for all $v \in L^{(i)}$ set $\mathbf{S}^{(i,j)}_{v,v} = (1+\beta_\eta)^{-j}$. It is easy to see that both conditions hold. Every edge set $E^{(i,j)}$ is constructed as $E^{(i,j)} = \{e \in E_{L^{(i)} \to R^{(i)}} : \eta_e \in [(1+\beta_\eta)^j, (1+\beta_\eta)^{j+1})\}$, and since we define the scaling matrix $\mathbf{S}^{(i,j)}$ to have $(1+\beta_\eta)^{-j}$ on entries corresponding to $v \in L$, after scaling all the flow multipliers are between 1 and $1 + \beta_\eta$.

Likewise, define $E^{(i,j)}$ for $j = \lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil + 1, \lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil + 2, \cdots, 2\lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil$ for the edges going from $R^{(i)}$ to $L^{(i)}$. This gives us the full map. For any edge $e = (u, v, \eta_e)$, we map this edge to:

$$f(e) = \begin{cases} (i, \lfloor \frac{\log \eta_e}{\log(1+\beta_\eta)} \rfloor), & \text{if } e \in E_{L^{(i)} \to R^{(i)}}, \\ (i, \lceil \frac{\log(W_\eta)}{\log(1+\beta_\eta)} \rceil + \lfloor \frac{\log \eta_e}{\log(1+\beta_\eta)} \rfloor, & \text{if } e \in E_{R^{(i)} \to L^{(i)}}. \end{cases} \tag{22}$$

where i is the first bit where u and v differ. □

Param	ϕ	β_η	ϵ_{add}
Value	$\log^{-3}(m)$	$\log^{-11}(m)$	$m^{-1} W_\eta^{-2} W_g^{-2} \lambda_1 \log^{-1}(W_\eta) \log^{-1}(W_g) \log^{-12}(m)$

Table 2: Choice of parameters for Theorem 7.6, where W_η, W_g, λ_1 are defined in the theorem statement.

Now we are ready to prove Theorem 7.6.

Proof of Theorem 7.6. In the proof we show how to perform the operations Initialize, Scale, ScaleTau, QueryHeavy, and Sample using the heavy hitter data structure for balanced lossy graphs in Theorem 7.2. In this proof we will use parameters $\beta_\eta = 1/\log^{11} m$, $\phi = 1/\log^3 m$, and $\epsilon_{\text{add}} = m^{-1} W_\eta^{-2} W_g^{-2} \lambda_1 \log^{-1}(W_\eta) \log^{-1}(W_g) \log^{-12}(m)$, as summarized in Table 2. We also denote $g_{\min} = \min_{e' \in E} g_{e'}$ and $g_{\max} = \max_{e' \in E} g_{e'}$.

Initialization. Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $g \in \mathbb{R}^m_{>0}$ denote the initial inputs, where $\mathbf{A}$ is the incidence matrix of a lossy graph $G = (V, E, \eta)$. We first use Theorem 7.7 with parameter β_η to decompose the edges of G into $E = \bigcup_{j=1}^{T} E^{(j)}$, where

$$T = O\big(\log(m)\log(W_\eta)\beta_\eta^{-1}\big) = O\big(\log^{12}(m)\log(W_\eta)\big).$$

Let $\mathbf{A}^{(j)}$ denote the submatrix of $\mathbf{A}$ with rows in $E^{(j)}$, let $\mathbf{S}^{(j)}$ be the vertex scaling matrix for $\mathbf{A}^{(j)}$ given by Theorem 7.7.

Next for every edge set $E^{(j)}$, we decompose it into $E^{(j)} = \bigcup_{k=1}^{\lceil \log(W_g)\rceil} E^{(j,k)}$, where $E^{(j,k)}$ include all $e \in E^{(j)}$ such that

$$g_{\min} \cdot 2^{k-1} \le g_e \le g_{\min} \cdot 2^k.$$

Note that the decomposition covers all $e \in E^{(j)}$ since every g_e satisfies $g_{\min} \le g_e \le g_{\min} \cdot 2^{\lceil \log(W_g)\rceil} = g_{\max}$ because we defined $W_g = \frac{g_{\max}}{g_{\min}}$. Let $\mathbf{A}^{(j,k)}$ denote the submatrix of $\mathbf{A}$ with rows in $E^{(j,k)}$. Let $G^{(j,k)}$ denote the subgraph of G with edges in $E^{(j,k)}$ and multipliers $\eta_e^{(j,k)} = (1+\eta_e)\cdot \mathbf{S}^{(j)}_{a_e,a_e} - 1$ for all $e = (a_e, b_e) \in E^{(j,k)}$, which is equivalent to $G^{(j,k)}$ being an unweighted lossy graph with incidence matrix $\mathbf{A}^{(j,k)} \cdot \mathbf{S}^{(j)}$. Let $\mathbf{D}^{(j,k)} = \mathbf{D}_{G^{(j,k)}}$. Let $g^{(j,k)}$ and $\overline{\tau}^{(j,k)}$ be the subvector of g and $\overline{\tau}$ with entries in $E^{(j,k)}$.

For every $j \in [T]$ and every $k \in [\lceil \log(W_g)\rceil]$, we initialize a heavy hitter data structure $\mathrm{DS}^{(j,k)}$ of Theorem 7.2 for the lossy graph $G^{(j,k)}$, with parameters ϵ_{add} and ϕ, and vector $\overline{\tau}^{(j,k)}$.

By Theorem 7.7, computing the first decomposition takes $O(m\log(m))$ time. Computing the second decomposition takes $O(n)$ time. Initializing the $T \cdot \lceil \log(W_g)\rceil$ data structures takes $T\log(W_g)\cdot O(m\phi^{-2}\log^2(n/\epsilon_{\text{add}}))$ time by Theorem 7.2. So the total initialization time is

$$\begin{aligned} P &= O\Big(m\phi^{-2}\log^2(n/\epsilon_{\text{add}})\log(m)\log(W_\eta)\log(W_g)\beta_\eta^{-1}\Big) \\ &= O\Big(m\log^{20}(m)\log^2(\lambda_1^{-1})\log(W_\eta)^3\log(W_g)^3\Big), \end{aligned}$$

where we used the choice of parameters $\epsilon_{\text{add}} = m^{-1}W_\eta^{-2}W_g^{-2}\lambda_1\log^{-1}(W_\eta)\log^{-1}(W_g)\log^{-12}(m)$, and $\beta_\eta = 1/\log^{11} m$.

Scale. Given $i \in [m]$ and $b \in \mathbb{R}$, let a_i and b_i denote the two endpoints of edge i. We first find the set $E^{(j,k)}$ such that $i \in E^{(j,k)}$, and update g_i and $g_i^{(j,k)}$ to be b. Next we find $k' \in [1, \lceil \log(W_g)\rceil]$ such that the new scalar b satisfies

$$g_{\min} \cdot 2^{k'-1} \le b \le g_{\min} \cdot 2^{k'}.$$

If $k' = k$, then we don't change anything else.

If $k' \neq k$, then we call $\mathrm{DS}^{(j,k)}.\textsc{Delete}(i)$, and $\mathrm{DS}^{(j,k')}.\textsc{Insert}(a_i, b_i, \eta_i^{(j,k)})$, and we also delete $g_i^{(j,k)}$ from $g^{(j,k)}$, and add a new entry of value b to $g^{(j,k')}$ that corresponds to the newly added edge. This takes $O\left(\phi^{-3}\log(m)\log^2(n/\epsilon_{\text{add}})\right) = O(\log^{12}(m)\log^2(\lambda_1^{-1})\log(W_\eta)^2\log(W_g)^2)$ time by Theorem 7.2.

Insert. Given a new edge $e = (u, v, \eta_e)$, and its edge weight b, we use the map f in theorem 7.7 to find the bucket $E^{(j)}$ for $j = f(u, v, \eta_e)$ that e belongs to. Next, we find $k \in [1, \lceil \log(W_g)\rceil]$ such that the new scalar b satisfies

$$g_{\min} \cdot 2^{k-1} \le b \le g_{\min} \cdot 2^k.$$

and we call $\mathrm{DS}^{(j,k)}.\textsc{Insert}(u, v, \eta_e)$. Note that the guarantee of theorem 7.7 gives us that adding this edge to $E^{(j)}$ maintains the property that the subset of edges $E^{(j)}$ form a β_η-balanced lossy graph.

Note here that as the graph evolves, $g_{\min}$ may change. We keep this notation for ease of understanding, but it is easy to implement this data structure efficiently without actually changing each set, simply by defining an offset term.

This takes $O\left(\phi^{-2}\log(m)\log^2(n/\epsilon_{\text{add}})\right) = O(\log^9(m)\log^2(\lambda_1^{-1})\log(W_\eta)^2\log(W_g)^2)$ time by Theorem 7.2.

Delete. Given $e \in [m]$, we find the set $E^{(j,k)}$ that $e \in E^{(j,k)}$, then we call $\text{DS}^{(j,k)}$.DELETE(i).

This takes $O\left(\phi^{-3}\log(m)\log^2(n/\epsilon_{\text{add}})\right) = O(\log^{12}(m)\log^2(\lambda_1^{-1})\log(W_\eta)^2\log(W_g)^2)$ time by Theorem 7.2.

ScaleTau. Given $e \in [m]$ and $b \in \mathbb{R}$, we find the set $E^{(j,k)}$ such that $e \in E^{(j,k)}$, and call $\text{DS}^{(j,k)}$.SCALETAU(e, b). This takes $O(1)$ time by Theorem 7.2.

QueryHeavy. Given query vector $h \in \mathbb{R}^n$ and error parameter $\epsilon \in (0,1)$, we compute $h^{(j)} = (\mathbf{S}^{(j)})^{-1} \cdot h$ for all $j \in [T]$ and $\epsilon^{(k)} = \epsilon \cdot 2^{-k} g_{\min}^{-1}$ for all $k \in [\lceil \log(W_g) \rceil]$. We then call $\text{DS}^{(j,k)}$.QUERYHEAVY$(h^{(j)}, \epsilon^{(k)})$ for all $j \in [T]$ and $k \in [\lceil \log(W_g) \rceil]$. By Theorem 7.2, the time of the function call $\text{DS}^{(j,k)}$.QUERYHEAVY is

$$\begin{aligned}
&O\left(\phi^{-4}(\epsilon^{(k)})^{-2}\|(\mathbf{A}^{(j,k)}\mathbf{S}^{(j)})h^{(j)}\|_2^2 + \phi^{-4}\epsilon_{\text{add}}(\epsilon^{(k)})^{-2}\|(\mathbf{D}^{(j,k)})^{1/2}h^{(j)}\|_2^2 + n\log^2 m\right) \\
\leq\ &O\left(\phi^{-4}\epsilon^{-2}\|\mathbf{G}^{(j,k)}\mathbf{A}^{(j,k)}h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\epsilon^{-2}g_{\max}^2 m W_\eta^2 \cdot \|h\|_2^2 + n\log^2 m\right), \qquad (23)
\end{aligned}$$

where we bound the first term using the definition of $h^{(j)}$ and $\epsilon^{(k)}$, and that $g_e^{(j,k)} = \Theta(2^k g_{\min})$, and we bound the second term using that $G^{(j,k)}$ is β_η-balanced, so each diagonal entry of $\mathbf{D}^{(j,k)}$ is at most $m(1+\beta_\eta) \leq O(m)$, and $\mathbf{S}_{i,i}^{(j)} \geq W_\eta^{-1}$ by Theorem 7.7, and $\epsilon^{(k)} \geq \epsilon \cdot g_{\max}^{-1}$.

So the total runtime of this operation is

$$\begin{aligned}
&\sum_{j=1}^{T}\sum_{k=1}^{\log(W_g)} O\left(\phi^{-4}\epsilon^{-2}\|\mathbf{G}^{(j,k)}\mathbf{A}^{(j,k)}h\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\epsilon^{-2}g_{\max}^2 m W_\eta^2 \cdot \|h\|_2^2 + n\log^2 m\right) \\
\leq\ &O\Big(\phi^{-4}\epsilon^{-2}\|\mathbf{G}\mathbf{A}h\|_2^2 + n\log^{14}(m)\log(W_\eta)\log(W_g)\Big), \qquad (24)
\end{aligned}$$

where we bound the first term using the fact that $E^{(j,k)}$'s form an edge-disjoint decomposition of E, we bound the second term by $\|h\|_2^2 \leq \|\mathbf{A}h\|_2^2 \cdot \lambda_1^{-1} \leq \|\mathbf{G}\mathbf{A}h\|_2^2 \cdot g_{\min}^{-2} \cdot \lambda_1^{-1}$ and our choice of $\epsilon_{\text{add}} = m^{-1}W_\eta^{-2}W_g^{-2}\lambda_1 \cdot T^{-1}\log^{-1}(W_g)$, and lastly we bound the third term by $T = O\big(\log(m)\log(W_\eta)\beta_\eta^{-1}\big)$, $\beta_\eta = 1/\log^{11} m$.

Sample. Given a vector $h \in \mathbb{R}^n$ and parameters C_1, C_2, C_3, we first compute $h^{(j)} = (\mathbf{S}^{(j)})^{-1} \cdot h$ call $L^{(j,k)} = \text{DS}^{(j,k)}$.NORM$(h^{(j)})$ for all $j \in [T]$ and $k \in [\lceil \log(W_g) \rceil]$. We then compute

$$L = \sum_{j=1}^{T}\sum_{k=1}^{\lceil \log(W_g) \rceil} 2^{2k} g_{\min}^2 \cdot L^{(j,k)}.$$

By Theorem 7.2 each returned value $L^{(j,k)}$ satisfies

$$\|(\mathbf{A}^{(j,k)}\mathbf{S}^{(j)})h^{(j)}\|_2^2 \leq L^{(j,k)} \leq O\left(\phi^{-4}\|(\mathbf{A}^{(j,k)}\mathbf{S}^{(j)})h^{(j)}\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|(\mathbf{D}^{(j,k)})^{1/2}h\|_2^2\right).$$

Using a similar argument as how we proved Eq. (23),

$$\|\mathbf{G}\mathbf{A}h\|_2^2 \leq L \leq O\left(\phi^{-4}\|\mathbf{G}\mathbf{A}h\|_2^2 + \phi^{-4}\epsilon_{\text{add}} \cdot m g_{\max}^2 W_\eta^2 \|h\|_2^2\right),$$

then using $\|h\|_2^2 \le \|\mathbf{G}\mathbf{A}h\|_2^2 \cdot g_{\min}^{-2} \cdot \lambda_1^{-1}$ and our choice of $\epsilon_{\text{add}} \le m^{-1}W_\eta^{-2}W_g^{-2}\lambda_1$,

$$\|\mathbf{G}\mathbf{A}h\|_2^2 \le L \le O\left(\phi^{-4}\|\mathbf{G}\mathbf{A}h\|_2^2\right).$$

Next we define

$$\overline{C}_1^{(k)} = C_1 \cdot \frac{m}{\sqrt{n}} \cdot \frac{2^{2k}g_{\min}^2}{\phi^4 L}, \quad \overline{C}_2 = \frac{C_2}{\sqrt{n}},$$

and for all $j \in [T]$ and $k \in [\lceil \log(W_g) \rceil]$, we call $\text{DS}^{(j,k)}.\textsc{Sample}(h^{(j)}, \overline{C}_1^{(k)}, \overline{C}_2, C_3)$ to obtain a diagonal matrix $\mathbf{R}^{(j,k)}$ of size $|E^{(j,k)}| \times |E^{(j,k)}|$. We concatenate all $\mathbf{R}^{(j,k)}$ to form a diagonal matrix $\mathbf{R}$ of size $m \times m$. By Theorem 7.2, every $e \in E^{(j,k)}$ is sampled with probability

$$\begin{aligned}
p_e &\ge \min\left\{1,\ \overline{C_1}^{(k)} \cdot (\mathbf{A}^{(j,k)}\mathbf{S}^{(j)}h^{(j)})_e^2 + \overline{C}_2 + C_3\overline{\tau}_e\right\} \\
&= \min\left\{1,\ C_1 \cdot \frac{m}{\sqrt{n}} \cdot \frac{2^{2k}g_{\min}^2}{\phi^4 L} \cdot (\mathbf{A}^{(j,k)}h)_e^2 + \frac{C_2}{\sqrt{n}} + C_3\overline{\tau}_e\right\} \\
&\ge \min\left\{1,\ C_1\frac{m}{\sqrt{n}} \cdot \frac{(\mathbf{G}\mathbf{A}h)_e^2}{\|\mathbf{G}\mathbf{A}h\|_2^2} + C_2\frac{1}{\sqrt{n}} + C_3\overline{\tau}_e\right\},
\end{aligned}$$

where the second step follows from the definition that $h^{(j)} = (\mathbf{S}^{(j)})^{-1} \cdot h$ and the definition of $\overline{C}_1^{(k)}$, and the third step follows from $L \le O(\phi^{-4}\|\mathbf{G}\mathbf{A}h\|_2^2)$ and $g_e^{(j,k)} = \Theta(2^k g_{\min})$.

Finally we bound the runtime of $\textsc{Sample}(\cdot)$. The dominating term of the runtime is the time to call $\text{DS}^{(j,k)}.\textsc{Sample}(h^{(j)}, \overline{C}_1^{(k)}, \overline{C}_2, C_3)$, which by Theorem 7.2 is bounded by

$$O\Big((\overline{C}_1^{(k)}(\phi^{-4}\|\mathbf{A}^{(j,k)}\mathbf{S}^{(j)}h^{(j)}\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|(\mathbf{D}^{(j,k)})^{1/2}h\|_2^2) + \overline{C}_2 m + C_3\|\overline{\tau}^{(j,k)}\|_1)C_0\log m + n\log^3(m)\Big).$$

Using the same argument as how we proved Eq. (23) and (24),

$$\sum_{j=1}^{T}\sum_{k=1}^{\lceil \log(W_g)\rceil} 2^{2k}g_{\min}^2 \cdot \left(\phi^{-4}\|\mathbf{A}^{(j,k)}\mathbf{S}^{(j)}h^{(j)}\|_2^2 + \phi^{-4}\epsilon_{\text{add}}\|(\mathbf{D}^{(j,k)})^{1/2}h\|_2^2\right) \le O\left(\phi^{-4}\|\mathbf{G}\mathbf{A}h\|_2^2\right).$$

Using this equation and that $L \ge \|\mathbf{G}\mathbf{A}h\|_2^2$, $\overline{C}_1^{(k)} = C_1 \cdot \frac{m}{\sqrt{n}} \cdot \frac{2^{2k}g_{\min}^2}{\phi^4 L}$, $\overline{C}_2 = \frac{C_2}{\sqrt{n}}$, and $T = O\big(\log^{10}(m)\log(W_\eta)\big)$, the total time of all $\text{DS}^{(j,k)}.\textsc{Sample}$ calls is at most

$$O\left(C_0C_1\frac{m}{\sqrt{n}}\phi^{-8} + (C_0C_2\frac{m}{\sqrt{n}} + n) \cdot \log(W_\eta)\log(W_g)\log^{13}(m) + C_0C_3\|\overline{\tau}\|_1\log(m)\right),$$

the claimed time complexity in the theorem statement then follows from $\phi = \log^{-3}(m)$. □

7.3 Heavy Hitters on Two-Sparse Matrices (Reduction to General Lossy Graphs)

In this section we prove our main theorem Theorem 7.1. We prove Theorem 7.1 through a standard reduction from two-sparse matrices to lossy graphs. The proof of the following lemma closely follows the reduction of [Hoc04], and we include it here for completeness.

Lemma 7.8 (Reduction from two-sparse matrices to lossy graphs)**.** *Given any matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ that has at most two non-zero entries per row, there exists a lossy graph $G = (V, E, \eta)$ with $|E| = m$ and $|V| = 2n$, and a diagonal edge weight matrix $\mathbf{G} \in \mathbb{R}^{m\times m}$ that satisfy the following properties:*

1. *For any vector* $h \in \mathbb{R}^n$, $\mathbf{A} \cdot h = \mathbf{G}\overline{\mathbf{A}} \cdot \begin{bmatrix} h \\ -h \end{bmatrix}$, *where* $\overline{\mathbf{A}} \in \mathbb{R}^{m \times 2n}$ *is the incidence matrix of* G.

2. *The flow multipliers satisfy* $\max_{e \in [m]} \eta_e \leq W_{\mathbf{A}}^2$ *(Theorem 1.1), and the edge weights satisfy* $\max_{e \in [m]} \max(|\mathbf{G}_{e,e}|, \frac{1}{|\mathbf{G}_{e,e}|}) \leq W_{\mathbf{A}}$.

Proof. We construct $\overline{\mathbf{A}}$ in two steps. First build $\widehat{\mathbf{A}}$, a matrix with a positive and negative entry in each row and second we build $\overline{\mathbf{A}}$, the adjacency matrix, where the positive entry is 1. First, we define $\widehat{\mathbf{A}}$ to be a matrix of size $m \times 2n$, where every row of $\mathbf{A}$ corresponds to a row of $\widehat{\mathbf{A}}$. There are three cases for every row of $\mathbf{A}$: (1) The row contains two non-zero entries of the same sign. (2) The row contains one positive entry and one negative entry. (3) The row contains one non-zero entry. Next we show how the corresponding rows of $\widehat{\mathbf{A}}$ are constructed for each of these three cases:

- For every row $e \in [m]$ of $\mathbf{A}$ that has two non-zero entries of the same sign, denote these two entries as $\mathbf{A}_{e,i} = \alpha$ and $\mathbf{A}_{e,j} = \beta$. Construct the e-th row of $\widehat{\mathbf{A}}$ as follows: Let $\widehat{\mathbf{A}}_{e,i} = \alpha$, $\widehat{\mathbf{A}}_{e,n+j} = -\beta$, and let all other entries be 0.
- For every row $e \in [m]$ of $\mathbf{A}$ that has one positive entry and one negative entry, denote these two entries as $\mathbf{A}_{e,i} = \alpha > 0$ and $\mathbf{A}_{e,j} = -\beta < 0$. Construct the e-th row of $\overline{\mathbf{A}}$ as follows: Let $\widehat{\mathbf{A}}_{e,i} = \alpha$, $\widehat{\mathbf{A}}_{e,j} = -\beta$, and let all other entries be 0.
- For every row $e \in [m]$ of $\mathbf{A}$ that has exactly one non-zero entry, denote this entry as $\mathbf{A}_{e,i} = \alpha$, and let $\sigma = \text{sign}(\alpha) \in \{1, -1\}$. Construct the e-th row of $\widehat{\mathbf{A}}$ as follows: Let $\widehat{\mathbf{A}}_{e,i} = \alpha/2$, $\widehat{\mathbf{A}}_{e,n+i} = -\alpha/2$, and let all other entries be 0.

It is easy to see that $\mathbf{A} \cdot h = \widehat{\mathbf{A}} \cdot \begin{bmatrix} h \\ -h \end{bmatrix}$, and that every row of $\widehat{\mathbf{A}} \in \mathbb{R}^{m \times 2n}$ has exactly one positive entry and one negative entry. See Figure 3 for an illustration of the construction of $\widehat{\mathbf{A}}$.

$$\mathbf{A} = \begin{bmatrix} \alpha_1 & & \beta_1 \\ & -\alpha_2 & -\beta_2 \\ -\beta_3 & & \alpha_3 \\ \alpha_4 & & \end{bmatrix}, \quad \widehat{\mathbf{A}} = \left[\begin{array}{ccc|ccc} \alpha_1 & & & & & -\beta_1 \\ & -\alpha_2 & & & & \beta_2 \\ -\beta_3 & & \alpha_3 & & & \\ \alpha_4/2 & & & & -\alpha_4/2 & \end{array}\right]$$

Figure 3: An example of a two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $\alpha_1, \alpha_2, \alpha_3, \alpha_4, \beta_1, \beta_2, \beta_3 \geq 0$, and the matrix $\widehat{\mathbf{A}} \in \mathbb{R}^{m \times 2n}$ corresponding to it.

We will next build $\overline{\mathbf{A}}$ from $\widehat{\mathbf{A}}$. For every $e \in [m]$, denote these two entries as $\widehat{\mathbf{A}}_{e,i} = \alpha > 0$ and $\widehat{\mathbf{A}}_{e,j} = -\beta < 0$, and define a diagonal weight matrix $\mathbf{G} \in \mathbb{R}^{m \times m}$ and a vector $\eta \in \mathbb{R}^m$ as follows:

$$\mathbf{G}_{e,e} = \begin{cases} \alpha & \text{if } \beta \geq \alpha, \\ -\beta & \text{if } \beta < \alpha. \end{cases} \quad \eta_e = \begin{cases} -\frac{\beta}{\alpha} & \text{if } \beta \geq \alpha, \\ -\frac{\alpha}{\beta} & \text{if } \beta < \alpha. \end{cases}$$

We then define $\overline{\mathbf{A}} \in \mathbb{R}^{m \times 2n}$ as $\overline{\mathbf{A}} \stackrel{\text{def}}{=} \mathbf{G}^{-1} \cdot \widehat{\mathbf{A}}$, and it's straightforward to see each row of $\overline{\mathbf{A}}$ contains exactly two non-zero entries: one is 1, and the other is $-\eta_e \in (-W_{\mathbf{A}}^2, -1]$. Also note that

by the definition of $W_{\mathbf{A}}$ in Theorem 1.1, $\alpha, \beta, \alpha^{-1}, \beta^{-1} \leq W_{\mathbf{A}}$, so $\eta_e \leq W_{\mathbf{A}}^2$, $\|\mathbf{G}\|_\infty \leq W_{\mathbf{A}}$, and $\|\mathbf{G}^{-1}\|_\infty \leq W_{\mathbf{A}}$. Lastly, since $\mathbf{A} \cdot h = \widehat{\mathbf{A}} \cdot \begin{bmatrix} h \\ -h \end{bmatrix}$, and $\overline{\mathbf{A}} = \mathbf{G}^{-1}\widehat{\mathbf{A}}$, we have that $\mathbf{A} \cdot h = \mathbf{G}\overline{\mathbf{A}} \cdot \begin{bmatrix} h \\ -h \end{bmatrix}$ as desired. □

Now we prove Theorem 7.1 using Theorem 7.6 and Theorem 7.8.

Proof of Theorem 7.1. Using theorem 7.8, whenever we are given a two-sparse matrix $\mathbf{A}$, we instead consider the problem on the lossy incidence matrix $\overline{\mathbf{A}}$, with query vectors $\begin{bmatrix} h \\ -h \end{bmatrix}$, and edge weights $\mathbf{G}$. We now bound the parameters of the new problem:

- $\log(W_\eta) \leq O(\log(W_{\mathbf{A}}))$ and $\log(W_g) \leq O(\log(W_{\mathbf{A}}))$ by Theorem 7.8.
- λ_1 is Note that

$$\|\mathbf{A}h\|_2 \geq \sqrt{\lambda_1}\|h\|_2 \implies \left\|\mathbf{G}\overline{\mathbf{A}} \cdot \begin{bmatrix} h \\ -h \end{bmatrix}\right\|_2 \geq \sqrt{\frac{1}{2}\lambda_1}\left\|\begin{bmatrix} h \\ -h \end{bmatrix}\right\|_2$$

.

It is easy to see that the operations of the two data structuress correspond exactly, since $\mathbf{A} \cdot h = \mathbf{G}\overline{\mathbf{A}} \cdot \begin{bmatrix} h \\ -h \end{bmatrix}$. □

8 Two-Sparse Linear Programs and Generalized Min-Cost Flows

The goal of this section is to prove our main results, Theorem 1.2 and Theorem 1.4, by implementing the IPM algorithm of [vdBLL+21] using the data structure we developed in Section 7. To make this section self-contained, we restate the relevant definitions and theorems from [vdBLL+21] that are used in our proofs. In particular, we restate several of their intermediate theorems to carefully track the dependence on the magnitude parameter W and the error parameter δ in our specific settings of two-sparse LPs and generalized min-cost flows.

This section is structured as follows. In Section 8.1, we first restate the formal definitions of the three data structures required by the IPM algorithm of [vdBLL+21], and show how to implement them for two-sparse matrices using our data structure from Theorem 7.1. In Section 8.2, we prove that using these three data structures, we can implement the algorithm of [vdBLL+21] to approximately follow the central path for two-sparse LPs. In Section 8.3, we describe the standard procedures for constructing an initial central point, and for rounding a final central point into an approximate LP solution. Finally in Section 8.4, we combine all the results of this section to prove our main results, Theorem 1.2 and Theorem 1.4.

8.1 Data Structures for the IPM

In this section we show how to use the data structure we developed in Theorem 7.1 to build the the heavy hitter, sampler and inverse maintenance data structures required by the IPM algorithm of [vdBLL+21].

Heavy Hitter. We first restate the formal definition of the heavy hitter data structure of [vdBLL+21].

Definition 8.1 (Heavy hitter, Definition 3.1 of [vdBLN+20])**.** *For $c \in \mathbb{R}^m$ and $P, Q \in \mathbb{R}_{>0}$ with $nP \geq \|c\|_1 \geq P$, we call a data structure with the following procedures a (P, c, Q)-*HeavyHitter *data structure:*

- Initialize($\mathbf{A} \in \mathbb{R}^{m \times n}, g \in \mathbb{R}^m_{>0}$)*: Let $\mathbf{A}$ be a matrix with $c_i \geq \mathrm{nnz}(a_i)$, $\forall i \in [m]$ and $P \geq \mathrm{nnz}(\mathbf{A})$. The data structure initializes in $O(P)$ time.*
- Scale($i \in [m], b \in \mathbb{R}$)*: Sets $g_i \leftarrow b$ in $O(c_i)$ time.*
- QueryHeavy($h \in \mathbb{R}^n, \epsilon \in (0, 1)$)*: Returns $I \subset [m]$ containing exactly those i with $|(\mathbf{GA}h)_i| \geq \epsilon$ in $O(\epsilon^{-2}\|\mathbf{GA}h\|_c^2 + Q)$ time.*

Using Theorem 7.1 we have the following (P, c, Q)-HeavyHitter data structure for two-sparse matrices.

Corollary 8.2 (Heavy hitters on two-sparse matrices)**.** *There exists a (P, c, Q)-*HeavyHitter *data structure for any two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ that succeeds with high probability with*

$$
\begin{aligned}
P &= \widetilde{O}\Big(m \log^2(W_g \lambda_{\min}(\mathbf{A}^\top \mathbf{A})^{-1}) \log^8(W_g W_{\mathbf{A}})\Big), \\
c &= \widetilde{O}\left(\log^2(W_g \lambda_{\min}(\mathbf{A}^\top \mathbf{A})^{-1}) \log^6(W_g W_{\mathbf{A}})\right) \cdot \mathbb{1}_m, \\
Q &= \widetilde{O}\left(n \log^2(W_g W_{\mathbf{A}})\right),
\end{aligned}
$$

where $W_{\mathbf{A}}$ is defined in Theorem 1.1, and W_g is the ratio of the largest to smallest non-zero entry in g.

Proof. We let W'_g denote the ratio of the largest non-zero entry in any g throughout the algorithm to smallest non-zero entry in any g throughout the algorithm. We can without loss of generality assume that $W'_g \leq W_g^2$, because otherwise the largest non-zero entry of g at some time t is less than the smallest non-zero entry of g at some other time t', and we can simply re-build the data structure when this happens.

Let $g_{\min}$ be the minimum value of the vector g throughout the algorithm. We implement the three operations using Theorem 7.1 as follows:

- Initialize: Given $\mathbf{A}$ and g, we let $\overline{g} \stackrel{\text{def}}{=} \frac{g}{g_{\min}}$, and initialize a data structure of Theorem 7.1 with $\overline{\mathbf{A}} \stackrel{\text{def}}{=} \overline{\mathbf{G}} \cdot \mathbf{A}$.
- Scale: Given i and b, we call the data structure of Theorem 7.1 to first delete row i from $\overline{\mathbf{A}}$, then insert a new row $\frac{b}{g_{\min}} \cdot a_i$, where a_i denotes the i-th row of $\mathbf{A}$.
- QueryHeavy: Given query vector h and parameter ϵ, we call the QueryHeavy operation of the data structure of Theorem 7.1 with h and $\overline{\epsilon} \stackrel{\text{def}}{=} \frac{\epsilon}{g_{\min}}$. Note that this is correct because $|(\mathbf{GA}h)_i| \geq \epsilon$ if and only if $|(\overline{\mathbf{A}}h)_i| \geq \overline{\epsilon}$, and $\overline{\epsilon}^{-2}\|\overline{\mathbf{A}}h\|_2^2 = \epsilon^{-2}\|\mathbf{GA}h\|_2^2$.

Next we bound the parameters λ_1 and $W_{\overline{\mathbf{A}}}$ that show up in the time complexity of Theorem 7.1. First note that during the algorithm we always have $\overline{g} \geq 1$ and $\max_i g_i \leq W'_g$. At any time a query is performed we have that

$$
\lambda_1(\overline{\mathbf{A}}^\top \overline{\mathbf{A}}) = \lambda_1(\mathbf{A}^\top \overline{\mathbf{G}}^2 \mathbf{A}) \leq W'_g \cdot \lambda_1(\mathbf{A}^\top \mathbf{A}).
$$

We also have that

$$W_{\overline{\mathbf{A}}} = \max_{i\in[n],e\in[m]:\overline{\mathbf{A}}_{e,i}\neq 0} \left(\max(|\overline{\mathbf{A}}_{e,i}|, \frac{1}{|\overline{\mathbf{A}}_{e,i}|})\right) \leq W'_g \cdot W_A.$$

The claimed time bounds of this Corollary then directly follows from Theorem 7.1. □

Sampler. Next we restate the formal definition of the sampler data structure of [vdBLL+21], which is defined using the following definition of a valid sampling distribution.

Definition 8.3 (Valid sampling distribution, definition 4.13 in [vdBLL+21])**.** *Given vector $\delta_r, \mathbf{A}, g, \overline{\tau}$, we say that a random diagonal matrix $\mathbf{R} \in \mathbb{R}^{m\times m}$ is C_{valid}-valid if it satisfies the following properties, for $\overline{\mathbf{A}} = \overline{\mathbf{T}}^{1/2}\mathbf{G}\mathbf{A}$. We assume that $C_{\text{valid}} \geq C_{\text{norm}}$.*

- *(Expectation) We have that $\mathbb{E}[\mathbf{R}] = \mathbf{I}$.*
- *(Variance) For all $i \in [m]$, we have that $\text{Var}[\mathbf{R}_{ii}(\delta_r)_i] \leq \frac{\eta|(\delta_r)_i|}{C^2_{\text{valid}}}$ and $\mathbb{E}[\mathbf{R}^2_{ii}] \leq 2\sigma(\overline{\mathbf{A}})_i^{-1}$.*
- *(Covariance) For all $i \neq j$, we have that $\mathbb{E}[\mathbf{R}_{ii}\mathbf{R}_{jj}] \leq 2$.*
- *(Maximum) With probability at least $1 - n^{-10}$ we have that $\|\mathbf{R}\delta_r - \delta_r\|_\infty \leq \frac{\eta}{C^2_{\text{valid}}}$.*
- *(Matrix approximation) We have that $\overline{\mathbf{A}}^\top\mathbf{R}\overline{\mathbf{A}} \approx_\eta \overline{\mathbf{A}}^\top\overline{\mathbf{A}}$ with probability at least $1 - n^{-10}$.*

Using the notion of a valid sampling distribution, we now restate the definition of the sampler of [vdBLL+21].

Definition 8.4 (Sampler data structure, Definition 6.3 in [vdBLL+21])**.** *We call a data structure a (P, c, Q)-HeavySampler data structure if it supports the following operations:*

- Initialize($\mathbf{A} \in \mathbb{R}^{m\times n}, g \in \mathbb{R}^m_{>0}, \overline{\tau} \in \mathbb{R}^m_{>0}$)*: Let $\mathbf{A}$ be a matrix with $c_i \geq \text{nnz}(a_i)$, $\forall i \in [m]$. The data structure initializes in $O(P)$ time.*
- Scale(i, a, b) *: Sets $g_i \leftarrow a$ and $\overline{\tau}_i \leftarrow b$ in $O(c_i)$ amortized time.*
- Sample($h \in \mathbb{R}^m$) *: Returns a random diagonal matrix $\mathbf{R} \in \mathbb{R}^{m\times m}$ that satisfies Theorem 8.3 for $\delta_r = \mathbf{G}\mathbf{A}h$ with $\|\delta_r\|_2 \leq m/n$ and $\overline{\tau} \approx_{1/2} \sigma(\overline{\mathbf{T}}^{1/2}\mathbf{G}\mathbf{A})$ in $O(Q)$ expected time. Furthermore, $\mathbb{E}[\text{nnz}(\mathbf{R}\mathbf{A})] = O(Q)$.*

Our data structure of Theorem 7.1 supports the proportional sampling of Lemma 4.42 of [vdBLL+21], and the output $\mathbf{R}$ satisfies Theorem 8.3 when choosing $C_0 = 100C^4_{\text{valid}}\gamma^{-2}\log m$, where $\gamma = \frac{\epsilon^2}{C^2\log(Cm/\epsilon^2)}$, $\epsilon = \frac{1}{4C\log(m/n)}$, and $C \geq 100$ is a constant. Similar to how [vdBLL+21] proved its sampler for graphs in Corollary F.4, we also use Corollary 4.42 of [vdBLL+21] and the ScaleTau and Sample operations of the data structure of Theorem 7.1 to obtain the following sampler for two-sparse matrices.

Corollary 8.5 (Sampler for two-sparse matrices)**.** *There exists a (P, c, Q)-HeavySampler data structure for any two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m\times n}$ where*

$$P = \widetilde{O}\Big(m\log^2(W_g\lambda_{\min}(\mathbf{A}^\top\mathbf{A})^{-1})\log^8(W_gW_{\mathbf{A}})\Big),$$
$$c = \widetilde{O}\left(\log^2(W_g\lambda_{\min}(\mathbf{A}^\top\mathbf{A})^{-1})\log^6(W_gW_{\mathbf{A}})\right)\cdot\mathbb{1}_m,$$
$$Q = \widetilde{O}\left((\frac{m}{\sqrt{n}}+n)\cdot\log^2(W_gW_{\mathbf{A}})\right),$$

where $W_{\mathbf{A}}$ is defined in Theorem 1.1, and W_g is the ratio of the largest to smallest non-zero entry in g.

Inverse Maintenance. Finally we restate the formal definition of the inverse maintenance data structure of [vdBLL+21].

Definition 8.6 (Inverse maintenance, Definition 6.2 of [vdBLN+20])**.** *We call a data structure a (P, c, Q)-*InverseMaintenance *data structure, if it supports the following operations:*

- Initialize($\mathbf{A} \in \mathbb{R}^{m \times n}, v \in \mathbb{R}^m, \bar{\sigma} \in \mathbb{R}^m$) *The data structure initializes in $O(P)$ time for $\bar{\sigma} \geq \frac{1}{2}\sigma(\mathbf{V}^{1/2}\mathbf{A})$ and $\|\bar{\sigma}\|_1 = O(n)$.*
- Scale($i \in [m], a, b$)*: Set $v_i \leftarrow a$ and $\bar{\sigma}_i \leftarrow b$ in $O(c_i)$ time.*
- Solve($\bar{v} \in \mathbb{R}^m, b, \epsilon \in (0,1)$)*: Assume $\bar{\sigma} \geq \frac{1}{2}\sigma(\mathbf{V}^{1/2}\mathbf{A})$ and the given $\bar{v}$ satisfies $\mathbf{A}^\top \mathbf{V}\mathbf{A} \approx_{1/2} \mathbf{A}^\top \overline{\mathbf{V}}\mathbf{A}$. Then* Solve *returns $\mathbf{H}^{-1}b$ for $\mathbf{H} \approx_\epsilon \mathbf{A}^\top \overline{\mathbf{V}}\mathbf{A}$ in $O\big((Q + \mathrm{nnz}(\overline{\mathbf{V}}\mathbf{A})) \cdot \log \epsilon^{-1}\big)$ time. Furthermore, for the same $\bar{v}$ and ϵ, the algorithm uses the same $\mathbf{H}$.*

For the complexity bounds one may further assume the following stability assumption: Let $\bar{v}^{(1)}, \bar{v}^{(1)}, \ldots$ be the sequence of inputs given to Solve*, then there exists a sequence $\tilde{v}^{(1)}, \tilde{v}^{(2)}, \ldots$ such that for all $t > 0$:*

$$\bar{v}^{(t)} \in (1 \pm 1/(100 \log n))\tilde{v}^{(t)} \text{ and } \|(\tilde{v}^{(t)})^{-1}(\tilde{v}^{(t)} - \tilde{v}^{(t+1)}\|_{\bar{\sigma}} = O(1).$$

We show that there exists such an inverse maintenance data structure for two-sparse matrices.

Theorem 8.7 (Inverse maintenance data structure for two-sparse matrices)**.** *There exists a (P, c, Q)-*InverseMaintenance *data structure for any two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ where $c_i = O(1)$, $P = \widetilde{O}(m)$, and $Q = \widetilde{O}(n \log(\kappa))$ where $\kappa = \frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{V}\mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{V}\mathbf{A})}$.*

Our proof uses the following fast Laplacian solver by [ST04] (see Theorem 8.8), where we use the stronger version that holds for any symmetric diagonally dominant (SDD) matrix. We will also use the M-matrix scaling theorem by [DS08] (see Theorem 8.9), which shows the stronger statement that given any matrix $\mathbf{M} = \mathbf{A}^\top \mathbf{A}$ where $\mathbf{A}$ has at most two non-zeros entries per row, there is an algorithm that can output a scaling matrix $\mathbf{D}$ such that $\mathbf{DMD}$ is SDD. Note that the off-diagonal entries of $\mathbf{M}$ can be positive.

Theorem 8.8 (Fast Laplacian solver, [ST04])**.** *There is an algorithm that takes as input a two-sparse matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, a non-negative diagonal matrix $\mathbf{D} \in \mathbb{R}^{m \times m}$, a vector $b \in \mathbb{R}^n$, and a parameter $\epsilon > 0$, such that $\mathbf{A}^\top \mathbf{D}\mathbf{A}$ is a symmetric diagonally dominant (SDD) matrix, and the algorithm returns a vector $\overline{x} = \mathbf{Z} \cdot b$ such that $\mathbf{Z} \approx_\epsilon (\mathbf{A}^\top \mathbf{D}\mathbf{A})^\dagger$. The algorithm runs in $\widetilde{O}(m \log(\epsilon^{-1}))$ time, and succeeds with high probability. Furthermore, for the same $\mathbf{A}$, $\mathbf{D}$, and ϵ, the algorithm uses the same $\mathbf{Z}$.*

Theorem 8.9 (M-Matrix Scaling, Theorem 4.5 of [DS08])**.** *There is an algorithm that takes as input any M-matrix $\mathbf{M} \in \mathbb{R}^{n \times n}$ and its factorization $\mathbf{M} = \mathbf{A}^\top \mathbf{A}$, where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is two-sparse, along with upper and lower bounds $\lambda_{\max}$ and $\lambda_{\min}$ for the eigenvalues of the matrix $\mathbf{A}$, and the algorithm returns a positive diagonal matrix $\mathbf{D} \in \mathbb{R}^{n \times n}$ such that the matrix $\mathbf{DMD}$ is a symmetric diagonally dominant (SDD) matrix. This algorithm runs in expected time $\widetilde{O}(m \log \kappa)$, where $\kappa \stackrel{\text{def}}{=} \lambda_{\max}/\lambda_{\min}$.*

Now we are ready to prove Theorem 8.7 using Theorem 8.8 and Theorem 8.9.

Proof of Theorem 8.7. In the INITIALIZE operation, the algorithm computes a random diagonal matrix $\mathbf{S} \in \mathbb{R}^{m \times m}$ where for each $\in [m]$, $\mathbf{S}_{i,i}$ is independently set to $\frac{1}{p_i}$ with probability $p_i = \min\{1, \Theta(\overline{\sigma}_i \log n)\}$, and it is set to 0 otherwise. The guarantees of leverage score sampling implies that $\mathbf{A}^\top \mathbf{V}^{1/2} \mathbf{S} \mathbf{V}^{1/2} \mathbf{A} \approx_{0.1} \mathbf{A}^\top \mathbf{V} \mathbf{A}$, and with high probability $\mathbf{S}$ has $O(n \log n)$ non-zero entries on its diagonal. This can be computed in $\widetilde{O}(m)$ time.

In the SCALE operation with input $i \in [m]$ and $a, b \in \mathbb{R}$, we re-sample the i-th entry of $\mathbf{S}$: We again set $\mathbf{S}_{i,i} = \frac{1}{p_i}$ with probability $p_i = \min\{1, \Theta(\overline{\sigma}_i \log n)\}$, and set it to 0 otherwise. This step takes $O(1)$ time.

In a SOLVE operation, since we maintained $\mathbf{S}$, we can use $(\mathbf{A}^\top \mathbf{V}^{1/2} \mathbf{S} \mathbf{V}^{1/2} \mathbf{A})^{-1}$ as a preconditioner for $\mathbf{A}^\top \overline{\mathbf{V}} \mathbf{A}$. First note that using Theorem 8.9 and Theorem 8.8 we can solve any linear system with $\mathbf{A}^\top \mathbf{V}^{1/2} \mathbf{S} \mathbf{V}^{1/2} \mathbf{A}$ to constant accuracy in $\widetilde{O}(n \log \kappa)$ time, where $\kappa = \frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{V} \mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{V} \mathbf{A})} = \Theta(\frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{V}^{1/2} \mathbf{S} \mathbf{V}^{1/2} \mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{V}^{1/2} \mathbf{S} \mathbf{V}^{1/2} \mathbf{A})})$. We then use preconditioned Richardson iterations

$$x^{(k+1)} = x^{(k)} + (\mathbf{A}^\top \mathbf{V}^{1/2} \mathbf{S} \mathbf{V}^{1/2} \mathbf{A})^{-1} (b - \mathbf{A}^\top \overline{V} \mathbf{A} x^{(k)}).$$

It takes $O(\log \epsilon^{-1})$ iterations to solve to ϵ accuracy. So the total runtime of SOLVE is $\widetilde{O}\big((n \log \kappa + \mathrm{nnz}(\overline{\mathbf{V}} \mathbf{A})) \log \epsilon^{-1}\big)$. □

8.2 Approximately Following the Central Path

In this section we prove that using the heavy hitter (Theorem 8.2), sampler (Theorem 8.5), and inverse maintenance (Theorem 8.7) data structures, we can implement the algorithm of [vdBLL+21] to approximately follow the central path for two-sparse LPs. This path following algorithm takes an initial central point as input and produces another central point with improved parameters. We leave the problems of finding the initial central point, and rounding the final central point to an LP solution to next sections.

Recall that we consider an LP of the following form, where $\mathbf{A} \in \mathbb{R}^{m \times n}$ is a two-sparse matrix:

$$\begin{aligned} \min_{x \in \mathbb{R}^m} \ & c^\top x \\ \text{s.t. } & \mathbf{A}^\top x = b \\ & \ell \le x \le u \end{aligned}$$

Path following algorithm from [vdBLL+21]. We first restate the guarantees of the PATHFOLLOWING algorithm of [vdBLL+21]. We use the following definitions from Section 4 of [vdBLL+21]. For the two-sided constraints $\ell_i \le x_i \le u_i$ for all $i \in [m]$, barrier function is defined as

$$\phi(x) = \sum_{i \in [m]} \phi_i(x_i), \text{ where } \phi_i(x_i) = -\log(x_i - \ell_i) - \log(u_i - x_i).$$

We also use the definition of regularized Lewis weights for a matrix. Define $p = 1 - \frac{1}{4 \log(4m/n)}$. For every matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, its regularized ℓ_p-Lewis weights $w(\mathbf{A}) \in \mathbb{R}^m_{>0}$ is defined as the solution of

$$w(\mathbf{A}) = \sigma(\mathbf{W}^{\frac{1}{2} - \frac{1}{p}} \mathbf{A}) + \frac{n}{m}, \text{ where } \mathbf{W} \stackrel{\text{def}}{=} \mathbf{diag}(w(\mathbf{A})).$$

Given any $x \in \mathbb{R}^m$, define the central path weights as

$$\tau(x) = w\Big(\mathbf{diag}(\phi''(x)^{-\frac{1}{2}}) \mathbf{A}\Big).$$

We use the following centrality definition from [vdBLL+21].

Definition 8.10 (ϵ-centered point, Definition 4.7 of [vdBLL+21]). *We say that $(x, s, \mu) \in \mathbb{R}^m \times \mathbb{R}^m \times \mathbb{R}^m_{>0}$ is ϵ-centered for $\epsilon \in (0, 1/80]$ if the following properties hold, where C is a constant such that $C \geq 100$, and $C_{\text{norm}} \stackrel{\text{def}}{=} C/(1-p)$, $\gamma \stackrel{\text{def}}{=} \frac{\epsilon^2}{C^2 \log(Cm/\epsilon^2)}$.*

1. *(Approximate centrality)* $\left\| \frac{s+\mu\tau(x)\phi'(x)}{\mu\tau(x)\sqrt{\phi''(x)}} \right\|_\infty \leq \epsilon$.
2. *(Dual Feasibility) There exists a vector $z \in \mathbb{R}^n$ with $\mathbf{A}z + s = c$.*
3. *(Approximate Feasibility)* $\|\mathbf{A}^\top x - b\|_{(\mathbf{A}^\top(\mathbf{T}(x)\Phi''(x))^{-1}\mathbf{A})^{-1}} \leq \epsilon\gamma/C_{\text{norm}}$.

The PathFollowing algorithm of [vdBLL+21] satisfies the following guarantees.

Theorem 8.11 (PathFollowing, Lemma 4.12 and Theorem 6.1 of [vdBLL+21]). *Consider a linear program with $\mathbf{A} \in \mathbb{R}^{m \times n}$, $\ell, u, c \in \mathbb{R}^m$, and $b \in \mathbb{R}^n$:*

$$\Pi : \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x.$$

Let $\epsilon = \frac{1}{4C\log(m/n)}$ for a large enough constant C. Given an ϵ-centered initial point $(x^{(\text{init})}, s^{(\text{init})}, \mu^{(\text{init})})$ for Π and a target $\mu^{(\text{final})}$, there exists an algorithm PathFollowing$(\mathbf{A}, b, \ell, u, c, \mu, \mu^{(\text{final})})$ *that returns an ϵ-centered point $(x^{(\text{final})}, s^{(\text{final})}, \mu^{(\text{final})})$.*

*Assume there exists a (P, c, Q)-*HeavyHitter *(Theorem 8.1), a (P, c, Q)-*InverseMaintenance *(Theorem 8.6), and a (P, c, Q)-*HeavySampler *(Theorem 8.4), then the total time of* PathFollowing *is*

$$\widetilde{O}\left(\left(\sqrt{P\|c\|_1} + \sqrt{n}\left(Q + n \cdot \max_i \text{nnz}(a_i)\right)\right) \log \frac{\mu^{(\text{init})}}{\mu^{(\text{final})}}\right).$$

Furthermore, the algorithm PathFollowing also guarantees that the parameters along the central path are bounded.

Lemma 8.12 (Parameter changes of PathFollowing, Lemma 4.46 of [vdBLL+21]). *For $\mathbf{A} \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^n, c \in \mathbb{R}^m$ and $\ell, u \in \mathbb{R}^m$, assume that the point $x^{(\text{init})} = (\ell + u)/2$ is feasible, i.e. $\mathbf{A}^\top x^{(\text{init})} = b$. Let W be the ratio of the largest to smallest entry of $\phi''(x^{(\text{init})})^{1/2}$, and let W' be the ratio of the largest to smallest entry of $\phi''(x)^{1/2}$ encountered in* PathFollowing. *Then:*

$$\log W' = \widetilde{O}\left(\log W + \log(1/\mu^{(\text{final})}) + \log\|c\|_\infty\right).$$

Approximately following the central path for two-sparse LPs. Next we use the data structures for two-sparse matrices to efficiently implement the PathFollowing algorithm.

Theorem 8.13 (Path following for two-sparse LPs). *Consider a linear program with $\mathbf{A} \in \mathbb{R}^{m \times n}$ that has at most two non zero entries per row, $\ell, u, c \in \mathbb{R}^m$, and $b \in \mathbb{R}^n$:*

$$\Pi : \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x.$$

Let $\epsilon = \frac{1}{4C\log(m/n)}$ for a large enough constant C. Given an ϵ-centered initial point $(x^{(\text{init})}, s^{(\text{init})}, \mu^{(\text{init})})$ for Π and a target $\mu^{(\text{final})}$, the algorithm PathFollowing$(\mathbf{A}, b, \ell, u, c, \mu^{(\text{init})}, \mu^{(\text{final})})$ *returns an ϵ-centered point $(x^{(\text{final})}, s^{(\text{final})}, \mu^{(\text{final})})$ with high probability in time*

$$\widetilde{O}\left(m \cdot \left(\log(\lambda_{\min}(\mathbf{A}^\top \mathbf{A})^{-1}) + \log(W_\phi) + \log(\frac{1}{\mu^{(\text{final})}}) + \log\|c\|_\infty + \log(W_{\mathbf{A}})\right)^{10} \log\frac{\mu^{(\text{init})}}{\mu^{(\text{final})}}\right)$$
$$+ \widetilde{O}\left(n^{1.5} \cdot \left(\log(\frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{A})}) + \log(W_\phi) + \log(\frac{1}{\mu^{(\text{final})}}) + \log\|c\|_\infty + \log(W_{\mathbf{A}})\right)^{2} \log\frac{\mu^{(\text{init})}}{\mu^{(\text{final})}}\right).$$

where $W_{\mathbf{A}}$ is defined in Theorem 1.1, and W is the ratio of the largest to smallest entry of $\phi''(x^{(\text{init})})^{1/2}$.

Proof. Using Theorem 8.11, we can implement the PathFollowing algorithm using the three data structures for two-sparse matrices: heavy hitter (Theorem 7.6), sampler (Theorem 8.5), and inverse maintenance (Theorem 8.7), and this gives a runtime of

$$\widetilde{O}\left(\left(m \cdot \log^2(W_g \lambda_{\min}(\mathbf{A}^\top \mathbf{A})^{-1}) \log^8(W_g W_{\mathbf{A}}) + n^{1.5} \cdot \left(\log^2(W_g W_{\mathbf{A}}) + \log(\kappa)\right)\right) \log\frac{\mu^{(\text{init})}}{\mu^{(\text{final})}}\right),$$

where W_g is the ratio of the largest to smallest non-zero entry in all scaling vectors g of the data structures, and $\kappa = \frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{V}\mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{V}\mathbf{A})}$, where $\mathbf{V}$ is the scaling matrix of InverseMaintenance.

Let W' be the ratio of the largest to smallest entry of $\phi''(x)^{1/2}$ encountered in PathFollowing, note that $W_g \le O(W')$, and the ratio of the largest to smallest entry of $\mathbf{V}$ of InverseMaintenance is also at most $O(W')$, so

$$\kappa = \frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{V}\mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{V}\mathbf{A})} \le O\left(\frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{A}) \max_i v_i}{\lambda_{\max}(\mathbf{A}^\top \mathbf{A}) \min_i v_i}\right) \le O\left(\frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{A})} \cdot W'\right).$$

Theorem 8.12 implies that $\log W' = \widetilde{O}\left(\log(W_\phi) + \log(1/\mu^{(\text{final})}) + \log\|c\|_\infty\right)$, where W_ϕ is the ratio of the largest to smallest entry of $\phi''(x^{(\text{init})})^{1/2}$.

So the total runtime is bounded by

$$\widetilde{O}\left(m \cdot \left(\log(\lambda_{\min}(\mathbf{A}^\top \mathbf{A})^{-1}) + \log(W_\phi) + \log(\frac{1}{\mu^{(\text{final})}}) + \log\|c\|_\infty + \log(W_{\mathbf{A}})\right)^{10} \log\frac{\mu^{(\text{init})}}{\mu^{(\text{final})}}\right)$$
$$+ \widetilde{O}\left(n^{1.5} \cdot \left(\log(\frac{\lambda_{\max}(\mathbf{A}^\top \mathbf{A})}{\lambda_{\min}(\mathbf{A}^\top \mathbf{A})}) + \log(W_\phi) + \log(\frac{1}{\mu^{(\text{final})}}) + \log\|c\|_\infty + \log(W_{\mathbf{A}})\right)^{2} \log\frac{\mu^{(\text{init})}}{\mu^{(\text{final})}}\right). \square$$

8.3 Initial and Final Point of LP

In this section we first show how to obtain an initial central point as required by the PathFollowing algorithm of Theorem 8.13.

Lemma 8.14 (LP initialization, Section 8.2 of [vdBLL+21])**.** *Given any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, any vector $b \in \mathbb{R}^n$, any vector $c \in \mathbb{R}^n$, any lower bound and upper bound vectors $\ell \le u \in \mathbb{R}^m$, and any accuracy parameter $\delta \in (0, 0.1)$, consider the following linear program:*

$$\min_{\substack{\mathbf{A}^\top x = b \\ \ell \le x \le u}} c^\top x. \tag{25}$$

Let $W \stackrel{\text{def}}{=} \max\left\{\|c\|_\infty, \|\mathbf{A}\|_\infty, \|b\|_\infty, \|u\|_\infty, \|\ell\|_\infty, \frac{\max_{i\in[m]}(u_i-\ell_i)}{\min_{i\in[m]}(u_i-\ell_i)}\right\}$, $\delta' \stackrel{\text{def}}{=} \frac{\delta}{10mW^2}\cdot\min\{1,\|c\|_1\}$, $\Xi \stackrel{\text{def}}{=} \max_{i\in[m]}|u_i-\ell_i|$, *and define the initial points by*

$$x^{(\text{init})} \stackrel{\text{def}}{=} (\ell+u)/2, \quad \beta \stackrel{\text{def}}{=} \|b-\mathbf{A}^\top x^{(\text{init})}\|_\infty/\Xi+1, \quad \widetilde{x}^{(\text{init})} \stackrel{\text{def}}{=} \frac{1}{\beta}|b-\mathbf{A}^\top x^{(\text{init})}|.$$

Finally, define a vector $\sigma = \text{sign}(b-\mathbf{A}^\top x^{(\text{init})}) \in \mathbb{R}^n$.

There exists a modified linear program $\min_{\substack{\overline{\mathbf{A}}^\top x=\overline{b}\\ \overline{\ell}\le x\le\overline{u}}} \overline{c}^\top x$ *with*

$$\overline{\mathbf{A}} = \begin{bmatrix}\mathbf{A}\\ \beta\cdot\mathbf{diag}(\sigma)\end{bmatrix},\ \overline{b}=b,\ \overline{c}=\begin{bmatrix}c\\ \frac{2\|c\|_1}{\delta'}\cdot\mathbb{1}_n\end{bmatrix},\ \overline{u}=\begin{bmatrix}u\\ 2\widetilde{x}^{(\text{init})}\end{bmatrix},\ \overline{\ell}=\begin{bmatrix}\ell\\ 0\end{bmatrix},$$

that satisfies the following guarantees:

1. *Define* $\overline{x}^{(\text{init})} \stackrel{\text{def}}{=} \begin{bmatrix}x^{(\text{init})}\\ \widetilde{x}^{(\text{init})}\end{bmatrix}$. *The point* $(\overline{x}^{(\text{init})}, \overline{c}, \mu)$ *is* ϵ*-centered for* $\mu = \frac{4m\|c\|_1\Xi}{\epsilon\delta'}$.

2. *Assume the linear program* (25) *has a feasible solution. For any* $\overline{x}^{(\text{final})} \stackrel{\text{def}}{=} \begin{bmatrix}x^{(\text{final})}\\ \widetilde{x}^{(\text{final})}\end{bmatrix}$ *that satisfies* $\overline{\mathbf{A}}^\top x^{(\text{final})} = \overline{b}$, $\overline{\ell} \le \overline{x}^{(\text{final})} \le \overline{u}$, *and* $\overline{c}^\top\overline{x}^{(\text{final})} \le \min_{\substack{\overline{\mathbf{A}}^\top x=\overline{b}\\ \overline{\ell}\le x\le\overline{u}}}\overline{c}^\top x+\delta$, *the point* $x^{(\text{final})}$ *satisfies*

$$c^\top x^{(\text{final})} \le \min_{\substack{\mathbf{A}^\top x=b\\ \ell\le x\le u}} c^\top x+\delta, \quad \textit{and} \quad \|\mathbf{A}^\top x^{(\text{final})}-b\|_\infty \le \delta.$$

3. $\overline{\mathbf{A}}$ *satisfies* $\sigma_{\min}(\overline{\mathbf{A}}) \ge 1$.

Our lemma differs from the initialization techniques of [vdBLL+21] in that we also need to lower bound the least singular value of $\overline{\mathbf{A}}$. For completeness we include a proof in Section A.

Next we restate the following lemma from [vdBLL+21] that shows how to round an ϵ-centered point produced by the PathFollowing algorithm into an approximate LP solution.

Lemma 8.15 (Final point, Lemma 4.11 of [vdBLL+21])**.** *Given an* ϵ*-centered point* (x,s,μ) *where* $\epsilon \le 1/80$, *we can compute a point* $(x^{(\text{final})}, s^{(\text{final})})$ *satisfying*

1. $\mathbf{A}^\top x^{(\text{final})} = b$, $s^{(\text{final})} = \mathbf{A}y+c$ *for some* y.
2. $c^\top x^{(\text{final})} - \min_{\substack{\mathbf{A}^\top x=b\\ \ell\le x\le u}} c^\top x \lesssim n\mu$.

The algorithm takes $O(\text{nnz}(\mathbf{A}))$ *time plus the time for solving a linear system on* $\mathbf{A}^\top\mathbf{D}\mathbf{A}$ *where* $\mathbf{D}$ *is a diagonal matrix.*

8.4 Proof of Main Theorems

In this section we prove our main results Theorem 1.2, Theorem 1.4, and Theorem 1.3.

Proof of Theorem 1.2. Given any LP $\min_{\substack{\mathbf{A}^\top x=b\\ \ell\le x\le u}} c^\top x$, we use Theorem 8.14 to construct a modified LP instance with $(\overline{\mathbf{A}}, \overline{b}, \overline{c}, \overline{\ell}, \overline{u})$ where $\overline{\mathbf{A}} = \begin{bmatrix}\mathbf{A}\\ \beta\mathbf{diag}(\sigma)\end{bmatrix}$ for $\beta > 1$ and $\mathbf{diag}(\sigma)$ a diagonal matrix

with 1s and -1s on the diagonal, together with an initial point $(\overline{x}^{(\text{init})}, \overline{c}, \mu^{(\text{init})})$ that is ϵ-centered. The new inputs are bounded by

$$\mu^{(\text{init})} \leq \text{poly}(m) \cdot \text{poly}(W) \cdot \delta^{-1}, \quad \beta \leq \text{poly}(m) \cdot \text{poly}(W),$$
$$\lambda_{\min}(\overline{\mathbf{A}}^\top \overline{\mathbf{A}}) \geq 1, \quad \lambda_{\max}(\overline{\mathbf{A}}^\top \overline{\mathbf{A}}) \leq \beta^2 \|\mathbf{A}\|_2^2 \leq \text{poly}(m) \cdot \text{poly}(W).$$

And since $x^{(\text{init})} = (\ell + u)/2$ and $\phi_i(x_i) = -\log(x_i - \ell_i) - \log(u_i - x_i)$, the entries of $\phi''(x^{(\text{init})})^{1/2}$ are $\frac{8}{(u_i - \ell_i)^2}\ W_\phi$, so

$$W_\phi \leq \text{poly}\left(\frac{\max_i(u_i - \ell_i)}{\min_i(u_i - \ell_i)}\right) \leq \text{poly}(W).$$

We set $\mu^{(\text{final})} = \frac{\delta}{\text{poly}(m)}$, and using Theorem 8.13, we can solve the modified LP $(\overline{\mathbf{A}}, \overline{b}, \overline{c}, \overline{\ell}, \overline{u})$ to obtain a ϵ-centered point $(x^{(\text{final})}, s^{(\text{final})}, \mu^{(\text{final})})$ with high probability in time

$$\widetilde{O}\left((m + n^{1.5}) \cdot \log^{10}(\frac{W}{\delta})\right).$$

Then using Theorem 8.15 we convert it into a solution $x^{(\text{final})}$ that satisfies

$$\|\mathbf{A}^\top x^{(\text{final})} - b\|_\infty \leq \delta \quad \text{and} \quad \ell \leq x^{(\text{final})} \leq u \quad \text{and} \quad c^\top x^{(\text{final})} \leq \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x + \delta. \qquad \square$$

Next we prove Theorem 1.4, which directly follows from Theorem 1.2.

Proof of Theorem 1.4. Given any lossy graph $G = (V, E, \gamma)$, costs $c \in \mathbb{R}^E$, capacities $u \in \mathbb{R}^E_{\geq 0}$, demands $d \in \mathbb{R}^V$, and $\delta > 0$, our goal is to solve the LP

$$\min_{\substack{\mathbf{B}_G^\top f = d \\ 0 \leq f \leq u}} c^\top f.$$

As a corollary of Theorem 1.2, we immediately have that we can solve this LP to δ error in time

$$\widetilde{O}\left((m + n^{1.5}) \cdot \log^{10}(\frac{W}{\delta})\right),$$

where $W \stackrel{\text{def}}{=} \max(W_{\mathbf{B}_G}, \|c\|_\infty, \|d\|_\infty, \|u\|_\infty, \frac{\max_e u_e}{\min_e u_e})$, $W_{\mathbf{B}_G} \stackrel{\text{def}}{=} \max_{e,i:(\mathbf{B}_G)_{e,i} \neq 0}(\max(|(\mathbf{B}_G)_{e,i}|, \frac{1}{|(\mathbf{B}_G)_{e,i}|}))$.

Note that by definition $W_{\mathbf{B}_G} \leq \max(\|\gamma\|_\infty, \|\gamma^{-1}\|_\infty)$, and this upper bound together with $\mathbf{B}_G^\top f = d$ implies that $\|d\|_\infty \leq O(\min\{1, \|\gamma\|_\infty\} \cdot \|u\|_\infty)$. So we can define

$$W' \stackrel{\text{def}}{=} \max\left(\|\gamma\|_\infty, \|\gamma^{-1}\|_\infty, \|c\|_\infty, \|u\|_\infty, \frac{\max_e u_e}{\min_e u_e}\right)$$

where $W' \leq O(W^2)$, and the algorithm runs in time $\widetilde{O}\left((m + n^{1.5}) \cdot \log^{10}(\frac{W'}{\delta})\right)$. $\square$

Next we prove Theorem 1.3, which also directly follows from Theorem 1.2, and we use the technique of [DS08] to make the flow feasible for the special case of lossy maxflow.

Proof of Theorem 1.3. Given any lossy graph $G = (V, E, \gamma)$, a source vertex $s \in V$ and a sink vertex $t \in V$, capacities $u \in \mathbb{R}^E_{\geq 0}$, and $\delta > 0$, our goal is to solve the LP

$$\min_{\substack{\mathbf{B}_{G\setminus\{s,t\}}^\top f = 0 \\ \mathbb{0} \leq f \leq u}} (\mathbf{B}_G^\top f)_t.$$

As a corollary of Theorem 1.2, this LP can be solved to δ error in time

$$\widetilde{O}\left((m + n^{1.5}) \cdot \log^{10}(\frac{W}{\delta})\right),$$

where $W \stackrel{\text{def}}{=} \max(W_{\mathbf{B}_G}, \|u\|_\infty, \frac{\max_e u_e}{\min_e u_e})$, and $W_{\mathbf{B}_G} \stackrel{\text{def}}{=} \max_{e,i:(\mathbf{B}_G)_{e,i}\neq 0}(\max(|(\mathbf{B}_G)_{e,i}|, \frac{1}{|(\mathbf{B}_G)_{e,i}|}))$, and by definition $W_{\mathbf{B}_G} \leq \max(\|\gamma\|_\infty, \|\gamma^{-1}\|_\infty)$.

In the special case of lossy maxflow with $\gamma \leq \mathbb{1}_m$, we further apply Lemma 3.2 of [DS08] to convert this δ-approximately feasible flow into an exactly feasible flow in $\widetilde{O}(m)$ additional time. □

9 Conclusion and Open Problems

In this paper we provide a randomized $\widetilde{O}((m + n^{1.5}) \cdot \text{polylog}(\frac{W}{\delta}))$ time algorithm for solving two-sparse LPs. As a corollary, we obtain nearly-linear time algorithms for the generalized maximum flow and generalized minimum cost flow problems on moderately dense graphs where $m \geq n^{1.5}$.

Perhaps the most immediate open problem is whether our running times can be further improved. A key open problem is to close the gap between our running times for lossy flow and state-of-the-art running times for maximum flow, namely, improving the $\widetilde{O}((m+n^{1.5}) \cdot \text{polylog}(\frac{W}{\delta}))$ runtime to an almost-linear runtime of $m^{1+o(1)} \cdot \text{polylog}(\frac{W}{\delta})$.

Another natural direction for future research is to further improve our dependence on the conditioning of the problem, e.g., improving the exponent in the $\text{polylog}(\frac{W}{\delta})$ factors, and expressing the dependence to scale-invariant condition measures rather than the current $W_{\mathbf{A}} = \max_{i,j:\mathbf{A}_{i,j}\neq 0}(\max(|\mathbf{A}_{i,j}|, \frac{1}{|\mathbf{A}_{i,j}|}))$.

Finally, investigating further spectral characterizations of lossy graphs and finding further applications of the spectral results proved in this paper are interesting directions for future work.

Acknowledgements

Thank you to the reviewers for their helpful anonymous feedback. This research was done in part during the authors' visit to the Simons Institute for the Theory of Computing at UC Berkeley. Shunhua Jiang is supported by the ERC Starting Grant #101039914. Lawrence Li is supported by grants awarded to Sushant Sachdeva — an NSERC Discovery grant RGPIN-2025-06976 and an Ontario Early Researcher Award (ERA) ER21-16-283. Aaron Sidford was funded in part by a Microsoft Research Faculty Fellowship, NSF CAREER Award CCF-1844855, NSF Grant CCF1955039, and a PayPal research award.

References

[AALOG18] Vedat Levi Alev, Nima Anari, Lap Chi Lau, and Shayan Oveis Gharan. Graph Clustering using Effective Resistance. In Anna R. Karlin, editor, *9th Innovations in*

Theoretical Computer Science Conference (ITCS 2018), volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 41:1–41:16, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[AJSS19] AmirMahdi Ahmadinejad, Arun Jambulapati, Amin Saberi, and Aaron Sidford. Perron-frobenius theory in nearly linear time: positive eigenvectors, m-matrices, graph kernels, and other applications. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '19, page 1387–1404, USA, 2019. Society for Industrial and Applied Mathematics.

[AMO+93] Ravindra K Ahuja, Thomas L Magnanti, James B Orlin, et al. *Network flows: theory, algorithms, and applications*, volume 1. Prentice hall Englewood Cliffs, NJ, 1993.

[AMV20] Kyriakos Axiotis, Aleksander Madry, and Adrian Vladu. Circulation Control for Faster Minimum Cost Flow in Unit-Capacity Graphs . In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 93–104, Los Alamitos, CA, USA, November 2020. IEEE Computer Society.

[AMV22] Kyriakos Axiotis, Aleksander Madry, and Adrian Vladu. Faster sparse minimum cost flow by electrical flow localization. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 528–539, 2022.

[BBST24] Aaron Bernstein, Joakim Blikstad, Thatchaphol Saranurak, and Ta-Wei Tu. Maximum Flow by Augmenting Paths in $n^{2+o(1)}$ Time . In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2056–2077, Los Alamitos, CA, USA, October 2024. IEEE Computer Society.

[BCK+23] Jan Van Den Brand, Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, Sushant Sachdeva, and Aaron Sidford. A Deterministic Almost-Linear Time Algorithm for Minimum-Cost Flow . In *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 503–514, Los Alamitos, CA, USA, November 2023. IEEE Computer Society.

[BGS22] Aaron Bernstein, Maximilian Probst Gutenberg, and Thatchaphol Saranurak. Deterministic Decremental SSSP and Approximate Min-Cost Flow in Almost-Linear Time . In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 1000–1008, Los Alamitos, CA, USA, February 2022. IEEE Computer Society.

[CK24] Julia Chuzhoy and Sanjeev Khanna. Maximum bipartite matching in $n^{2+o(1)}$ time via a combinatorial algorithm. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 83–94, New York, NY, USA, 2024. Association for Computing Machinery.

[CKL+22] Li Chen, Rasmus Kyng, Yang P Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Maximum flow and minimum-cost flow in almost-linear time. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 612–623. IEEE, 2022.

[CKL+23] Li Chen, Rasmus Kyng, Yang P. Liu, Richard Peng, Maximilian Probst Gutenberg, and Sushant Sachdeva. Almost-linear-time algorithms for maximum flow and minimum-cost flow. *Commun. ACM*, 66(12):85–92, November 2023.

[CKM+11] Paul Christiano, Jonathan A. Kelner, Aleksander Madry, Daniel A. Spielman, and Shang-Hua Teng. Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs. In *Proceedings of the Forty-Third Annual ACM Symposium on Theory of Computing*, STOC '11, page 273–282, New York, NY, USA, 2011. Association for Computing Machinery.

[CLM+16] Michael B Cohen, Yin Tat Lee, Gary Miller, Jakub Pachocki, and Aaron Sidford. Geometric median in nearly linear time. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 9–21, 2016.

[CM91] Edith Cohen and Nimrod Megiddo. Improved algorithms for linear inequalities with two variables per inequality. In *Proceedings of the twenty-third annual ACM symposium on Theory of Computing*, pages 145–155, 1991.

[CMSV17] Michael B. Cohen, Aleksander Madry, Piotr Sankowski, and Adrian Vladu. Negative-weight shortest paths and unit capacity minimum cost flow in $\tilde{O}(m^{10/7} \log W)$ time: (extended abstract). In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '17, page 752–771, USA, 2017. Society for Industrial and Applied Mathematics.

[DKN+24] Daniel Dadush, Zhuan Khye Koh, Bento Natura, Neil Olver, and László A. Végh. A strongly polynomial algorithm for linear programs with at most two nonzero entries per row or column. In *Proceedings of the 56th Annual ACM Symposium on Theory of Computing*, STOC 2024, page 1561–1572, New York, NY, USA, 2024. Association for Computing Machinery.

[DS08] Samuel I Daitch and Daniel A Spielman. Faster approximate lossy generalized flow via interior point algorithms. In *Proceedings of the fortieth annual ACM symposium on Theory of computing*, pages 451–460, 2008.

[FW02] Lisa K. Fleischer and Kevin D. Wayne. Fast and simple approximation schemes for generalized flow. *Mathematical Programming*, 91(2):215–238, 2002.

[GJO97] Donald Goldfarb, Zhiying Jin, and James B. Orlin. Polynomial-time highest-gain augmenting path algorithms for the generalized circulation problem. *Mathematics of Operations Research*, 22(4):793–802, 1997.

[GKL+21] Grzegorz Gluch, Michael Kapralov, Silvio Lattanzi, Aida Mousavifar, and Christian Sohler. Spectral clustering oracles in sublinear time. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1598–1617. SIAM, 2021.

[GLP22] Yu Gao, Yang P. Liu, and Richard Peng. Fully dynamic electrical flows: Sparse maxflow faster than goldberg-rao. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 516–527, 2022.

[GPT88] A.V. Goldberg, S.A. Plotkin, and E. Tardos. Combinatorial algorithms for the generalized circulation problem. In *[Proceedings 1988] 29th Annual Symposium on Foundations of Computer Science*, pages 432–443, 1988.

[HN94] Dorit S. Hochbaum and Joseph (Seffi) Naor. Simple and fast algorithms for linear and integer programs with two variables per inequality. *SIAM Journal on Computing*, 23(6):1179–1192, 1994.

[Hoc04] Dorit S Hochbaum. Monotonizing linear programs with up to two nonzeroes per column. *Operations Research Letters*, 32(1):49–58, 2004.

[JL84] William Johnson and Joram Lindenstrauss. Extensions of lipschitz maps into a hilbert space. *Contemporary Mathematics*, 26:189–206, 01 1984.

[Kar22] Adam Karczmarz. Improved strongly polynomial algorithms for deterministic MDPs, 2VPI feasibility, and discounted all-pairs shortest paths. In *Proceedings of the 2022 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 154–172. SIAM, 2022.

[KLS20] Tarun Kathuria, Yang P. Liu, and Aaron Sidford. Unit capacity maxflow in almost $o(m^{4/3})$ time. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 119–130, 2020.

[KW92] Jacek Kuczyński and Henryk Woźniakowski. Estimating the largest eigenvalue by the power and lanczos algorithms with a random start. *SIAM journal on matrix analysis and applications*, 13(4):1094–1122, 1992.

[LS14] Yin Tat Lee and Aaron Sidford. Path finding methods for linear programming: Solving linear programs in $\tilde{O}$(vrank) iterations and faster algorithms for maximum flow. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 424–433, 2014.

[LS20] Yang P. Liu and Aaron Sidford. Faster energy maximization for faster maximum flow. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 803–814, New York, NY, USA, 2020. Association for Computing Machinery.

[Mad13] Aleksander Madry. Navigating central path with electrical flows: From flows to matchings, and back. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 253–262, 2013.

[Mad16] Aleksander Madry. Computing Maximum Flow with Augmenting Electrical Flows . In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 593–602, Los Alamitos, CA, USA, October 2016. IEEE Computer Society.

[Meg83] Nimrod Megiddo. Towards a genuinely polynomial algorithm for linear programming. *SIAM Journal on Computing*, 12(2):347–353, 1983.

[ST04] Daniel A Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing (STOC)*, pages 81–90, 2004.

[SW19] Thatchaphol Saranurak and Di Wang. Expander decomposition and pruning: Faster, stronger, and simpler. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2616–2635. SIAM, 2019.

[Vai89] P.M. Vaidya. Speeding-up linear programming using fast matrix multiplication. In *30th Annual Symposium on Foundations of Computer Science*, pages 332–337, 1989.

[VDBCK+24] Jan Van Den Brand, Li Chen, Rasmus Kyng, Yang P. Liu, Simon Meierhans, Maximilian Probst Gutenberg, and Sushant Sachdeva. Almost-linear time algorithms for decremental graphs: Min-cost flow and more via duality. In *2024 IEEE 65th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 2010–2032, 2024.

[vdBGJ+22] Jan van den Brand, Yu Gao, Arun Jambulapati, Yin Tat Lee, Yang P. Liu, Richard Peng, and Aaron Sidford. Faster maxflow via improved dynamic spectral vertex sparsifiers. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, page 543–556, New York, NY, USA, 2022. Association for Computing Machinery.

[vdBLL+21] Jan van den Brand, Yin Tat Lee, Yang P Liu, Thatchaphol Saranurak, Aaron Sidford, Zhao Song, and Di Wang. Minimum cost flows, mdps, and ℓ_1-regression in nearly linear time for dense instances. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, pages 859–869, 2021.

[vdBLN+20] Jan van den Brand, Yin-Tat Lee, Danupon Nanongkai, Richard Peng, Thatchaphol Saranurak, Aaron Sidford, Zhao Song, and Di Wang. Bipartite matching in nearly-linear time on moderately dense graphs. In *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 919–930. IEEE, 2020.

[vdBLSS20] Jan van den Brand, Yin Tat Lee, Aaron Sidford, and Zhao Song. Solving tall dense linear programs in nearly linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 775–788, New York, NY, USA, 2020. Association for Computing Machinery.

[Vég14] László A Végh. A strongly polynomial algorithm for generalized flow maximization. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 644–653, 2014.

[Way02] Kevin D. Wayne. A polynomial combinatorial algorithm for generalized minimum cost flow. *Mathematics of Operations Research*, 27(3):445–459, 2002.

A Missing Proofs

A.1 Proof of LP Initialization

Proof of Theorem 8.14. **Part 1.** By definition we have $\overline{\mathbf{A}}^\top \overline{x}^{(\text{init})} = \overline{b}$, $\overline{\mathbf{A}} 0_n + \overline{c} = \overline{c}$, $\ell \le x^{(\text{init})} \le u$, and $0 \le \widetilde{x}^{(\text{init})} \le 2\widetilde{x}^{(\text{init})}$, so the point is feasible.

Next we bound centrality. Recall that the barrier for the interval $[\ell, u]$ is $\phi(x) = -\log(x-\ell) - \log(u-x)$, and its derivatives are $\phi'(x) = -\frac{1}{x-\ell} + \frac{1}{u-x}$, and $\phi''(x) = \frac{1}{(x-\ell)^2} + \frac{1}{(u-x)^2} \ge \frac{1}{(u-\ell)^2}$. For the old constraints, we have $\frac{1}{(u_i-\ell_i)^2} \ge \frac{1}{\Xi^2}$. For the new constraints, we have

$$\frac{1}{(2\widetilde{x}_i^{(\text{init})})^2} \ge \frac{1}{(2\|b - \mathbf{A}^\top x^{(\text{init})}\|_\infty/\beta)^2} \ge \frac{1}{4\Xi^2}.$$

Since $\tau(x) \geq \frac{n}{m}$, and since $\phi'(\overline{x}^{(\text{init})}) = 0$, we can bound centrality by

$$\left\| \frac{\overline{c} + \mu\tau(\overline{x}^{(\text{init})})\phi'(\overline{x}^{(\text{init})})}{\mu\tau(\overline{x}^{(\text{init})})\sqrt{\phi''(\overline{x}^{(\text{init})})}} \right\|_\infty \leq \max\left\{ \frac{\|c\|_\infty}{\mu\frac{n}{m}\frac{1}{\Xi}}, \frac{\frac{2\|c\|_1}{\delta'}}{\mu\frac{n}{m}\frac{1}{2\Xi}} \right\}$$
$$\leq \frac{4\|c\|_1 \Xi m}{\mu n \delta'} \leq \epsilon,$$

where the last step follows from $\mu = \frac{4m\|c\|_1\Xi}{\epsilon\delta'}$.

Part 2. First note that for any vector $x \in \mathbb{R}^m$ that is feasible for the original LP, the vector $\overline{x} = \begin{bmatrix} x \\ 0_n \end{bmatrix} \in \mathbb{R}^{m+n}$ is feasible for the modified LP. This means

$$\min_{\substack{\overline{\mathbf{A}}^\top \overline{x} = \overline{b} \\ \overline{\ell} \leq \overline{x} \leq \overline{u}}} \overline{c}^\top \overline{x} \leq \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x.$$

Using this we can bound the objective value as follows:

$$c^\top x^{(\text{final})} \leq \min_{\substack{\overline{\mathbf{A}}^\top \overline{x} = \overline{b} \\ \overline{\ell} \leq \overline{x} \leq \overline{u}}} \overline{c}^\top \overline{x} + \delta - \frac{2\|c\|_1}{\delta'} \cdot 1_n^\top \widetilde{x}^{(\text{final})}$$
$$\leq \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x + \delta,$$

where the second step follows from the equation above, and that $\widetilde{x}^{(\text{final})} \geq 0$ since $\overline{x}^{(\text{final})}$ is feasible.

Next we bound the error in feasibility. First note that

$$c^\top x^{(\text{final})} + \frac{2\|c\|_1}{\delta'} \cdot 1_n^\top \widetilde{x}^{(\text{final})} \leq \min_{\substack{\overline{\mathbf{A}}^\top x = \overline{b} \\ \overline{\ell} \leq x \leq \overline{u}}} \overline{c}^\top x + \delta \leq \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x + \delta.$$

Since $\ell \leq x^{(\text{final})} \leq u$, and the same bounds hold for the x in $\min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x$,

$$c^\top x^{(\text{final})} \geq \sum_{i=1}^m \min\{c_i\ell_i, c_i u_i\}, \quad \min_{\substack{\mathbf{A}^\top x = b \\ \ell \leq x \leq u}} c^\top x \leq \sum_{i=1}^m \max\{c_i\ell_i, c_i u_i\}.$$

Combining this and the inequality above yields

$$\sum_{i=1}^m \min\{c_i\ell_i, c_i u_i\} + \frac{2\|c\|_1}{\delta'} \cdot 1_n^\top \widetilde{x}^{(\text{final})} \leq \sum_{i=1}^m \max\{c_i\ell_i, c_i u_i\} + \delta.$$

Next note that for an $i \in [m]$, $\max\{c_i\ell_i, c_i u_i\} - \min\{c_i\ell_i, c_i u_i\} = |c_i| \cdot (u_i - \ell_i) \leq |c_i| \cdot \Xi$, which implies

$$\frac{2\|c\|_1}{\delta'} \cdot 1_n^\top \widetilde{x}^{(\text{final})} \leq \sum_{i=1}^m |c_i| \cdot \Xi + \delta$$
$$\implies 1_n^\top \widetilde{x}^{(\text{final})} \leq \delta' \cdot (\frac{\Xi}{2} + \frac{\delta}{2\|c\|_1}).$$

Finally, since $\overline{x}^{(\text{final})}$ is feasible, $\mathbf{A}^\top x^{(\text{final})} + \beta \cdot \mathbf{diag}(\sigma) \cdot \widetilde{x}^{(\text{final})} = b$, and so

$$\begin{aligned}
\|\mathbf{A}^\top x^{(\text{final})} - b\|_\infty &= \beta \cdot \|\widetilde{x}^{(\text{final})}\|_\infty \\
&\leq (\|b - \mathbf{A}^\top x^{(\text{init})}\|_\infty / \Xi + 1) \cdot \delta' \cdot (\frac{\Xi}{2} + \frac{\delta}{2\|c\|_1}) \\
&\leq \left(\|b\|_\infty \Xi^{-1} + m\|\mathbf{A}\|_\infty \max\{\|u\|_\infty, \|\ell\|_\infty\}\Xi^{-1} + 1\right) \cdot \delta' \cdot (\frac{\Xi}{2} + \frac{\delta}{2\|c\|_1}) \\
&\leq \delta,
\end{aligned}$$

where the last step follows from $\delta' \stackrel{\text{def}}{=} \frac{\delta}{10mW^2} \cdot \min\{1, \|c\|_1\}$.

Part 3. By definition, $\overline{\mathbf{A}}^\top \overline{\mathbf{A}} = \mathbf{A}^\top \mathbf{A} + \beta^2 I \succeq I$. □

A.2 Equivalent Statement of Small Rayleigh Quotient Condition

Lemma A.1. *Let $\mathbf{M}$ be any PSD matrix. For any $c \geq 0$, if v is a unit vector that satisfies $1 - \left(v^\top v_1(\mathbf{M})\right)^2 \leq \frac{c \cdot \lambda_1(\mathbf{M})}{\lambda_n(\mathbf{M})}$, then $v^\top \mathbf{M} v \leq (1+c) \cdot \lambda_1(\mathbf{M})$.*

Proof.

$$\begin{aligned}
v^\top \mathbf{M} v &= \sum_{i \in [n]} \lambda_i(\mathbf{M})(v^\top v_i(\mathbf{M}))^2 \\
&\leq \lambda_1(\mathbf{M}) + \sum_{i=2}^{n} \lambda_n(\mathbf{M})(v^\top v_i(\mathbf{M}))^2 = \lambda_1(\mathbf{M}) + \lambda_n(\mathbf{M}) \left(1 - (v^\top v_1(\mathbf{M}))^2\right) \leq (1+c)\lambda_1(\mathbf{M}) \,. \square
\end{aligned}$$