2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS) | 978-1-6654-5519-0/22/$31.00 ©2022 IEEE | DOI: 10.1109/FOCS54457.2022.00052

On Weighted Graph Sparsification by Linear Sketching

Yu Chen
University of Pennsylvania
chenyu2@cis.upenn.edu

Sanjeev Khanna
University of Pennsylvania
sanjeev@cis.upenn.edu

Huan Li
University of Pennsylvania
huanli@cis.upenn.edu

***Abstract*—A seminal work of [Ahn-Guha-McGregor, PODS'12] showed that one can compute a cut sparsifier of an *unweighted* undirected graph by taking a near-linear number of linear measurements on the graph. Subsequent works also studied computing other graph sparsifiers using linear sketching, and obtained near-linear upper bounds for spectral sparsifiers [Kapralov-Lee-Musco-Musco-Sidford, FOCS'14] and first non-trivial upper bounds for spanners [Filtser-Kapralov-Nouri, SODA'21]. All these linear sketching algorithms, however, only work on *unweighted* graphs, and are extended to weighted graphs by weight grouping, a non-linear operation not implementable in, for instance, general turnstile streams.**

In this paper, we initiate the study of *weighted* graph sparsification by linear sketching by investigating a natural class of linear sketches that we call *incidence sketches*, in which each measurement is a linear combination of the weights of edges incident on a *single* vertex. This class captures *all* aforementioned linear sketches for unweighted sparsification. It also covers linear sketches implementable in the *simultaneous communication model*, where edges are distributed across n machines. Our results are:

1) **Weighted cut sparsification: We give an algorithm that computes a $(1+\epsilon)$-cut sparsifier using $\tilde{O}(n\epsilon^{-3})$linear measurements, which is nearly optimal. This also implies a turnstile streaming algorithm with $\tilde{O}(n\epsilon^{-3})$ space. Our algorithm is achieved by building a so-called "weighted edge sampler" for each vertex.**
2) **Weighted spectral sparsification: We give an algorithm that computes a $(1+\epsilon)$-spectral sparsifier using $\tilde{O}(n^{6/5}\epsilon^{-4})$ linear measurements. This also implies a turnstile streaming algorithm with $\tilde{O}(n^{6/5}\epsilon^{-4})$ space. Key to our algorithm is a novel analysis of how the effective resistances change under *vertex sampling*. Complementing our algorithm, we then prove a superlinear lower bound of $\Omega(n^{21/20-o(1)})$ measurements for computing some $O(1)$-spectral sparsifier using incidence sketches.**
3) **Weighted spanner computation: We first show that any $o(n^2)$ linear measurements can only recover a spanner of stretch that in general depends *linearly* on $\frac{w_{\max}}{w_{\min}}$. We thus focus on graphs with $\frac{w_{\max}}{w_{\min}} = O(1)$ and study the stretch's dependence on n. On such graphs, the algorithm in [Filtser-Kapralov-Nouri, SODA'21] can obtain a spanner of stretch $\tilde{O}(n^{\frac{2}{3}(1-\alpha)})$ using $\tilde{O}(n^{1+\alpha})$ measurements for any $\alpha \in [0,1]$. We prove that, for incidence sketches, this tradeoff is *optimal up to an* $n^{o(1)}$ *factor for all* $\alpha < 1/10$.**

We prove both our lower bounds by analyzing the "effective resistances" in certain *matrix-weighted* graphs, where we develop a number of new tools for reasoning about such graphs – most notably (i) a matrix-weighted analog of the widely used *expander decomposition* of ordinary graphs, and (ii) a proof that a random *vertex-induced subgraph* of a matrix-weighted expander is also an expander. We believe these tools are of independent interest.

***Index Terms*—streaming algorithms, graph sparsification, linear sketch, spectral graph theory**

I. Introduction

Graph sparsification is a process that reduces the number of edges in a dense graph significantly while preserving certain useful properties. Besides being an interesting problem in its own right, graph sparsification has also been used as a fundamental building block in many modern graph algorithms such as maximum flow and minimum cut algorithms [4], [20], [31], [32], solvers for graph structured linear systems [6], [14], [34], and graph clustering [1], [5].

In addition to designing fast algorithms in the classic computational model, a rich body of work has also studied graph sparsification by *linear sketching*. In this setting, we can only access the input graph by taking *linear measurements*, each of which returns a linear combination of the edge weights, and the goal is then to compute a sparsifier of the input graph using as few measurements as possible. To state the previously known results in this setting, let us first recall the definitions of three extensively studied graph sparsifiers that we will study in this work.

Definition I.1 (Cut sparsifiers)**.** Given a weighted graph $G = (V, E, w)$ and a parameter $\epsilon \in (0,1)$, another weighted graph $H = (V, F, w')$ with $F \subseteq E$ is called a $(1+\epsilon)$-*cut sparsifier* of G if for every cut $(S, V-S)$, its weight $w_G(S, V-S)$ in G and its weight $w_H(S, V-S)$ in H satisfy that $(1-\epsilon)w_G(S, V-S) \leq w_H(S, V-S) \leq (1+\epsilon)w_G(S, V-S)$.

Definition I.2 (Spectral sparsifiers)**.** Given a weighted graph $G = (V, E, w)$ with Laplacian matrix L_G and a parameter $\epsilon \in (0,1)$, another weighted graph $H = (V, F, w')$ with Laplacian matrix L_H and $F \subseteq E$ is called a $(1+\epsilon)$-*spectral* sparsifier of G if for every vector $x \in \mathbb{R}^n$ we have $(1-\epsilon)x^T L_G x \leq x^T L_H x \leq (1+\epsilon)x^T L_G x$.

Definition I.3 (Spanners)**.** Given a weighted graph $G = (V, E, w)$ and a parameter $t \geq 1$ (called the *stretch*), another weighted graph $H = (V, F, w')$ with $F \subseteq E$ is called a t-*spanner* of G if for every vertex pair u, v, the shortest path

This work was supported in part by NSF awards CCF-1763514, CCF-1934876, and CCF-2008304.

978-1-6654-5519-0/22/$31.00 ©2022 IEEE
DOI 10.1109/FOCS54457.2022.00052

length $d_G(u,v)$[1] between u, v in G and the shortest path length $d_H(u,v)$ in H satisfy $d_G(u,v) \leq d_H(u,v) \leq t \cdot d_G(u,v)$.

A seminal work by Ahn, Guha, and McGregor [2] showed that one can compute a $(1+\epsilon)$-cut sparsifier of an unweighted graph using $\tilde{O}(n\epsilon^{-2})$ linear measurements, which is nearly optimal. Then subsequent works by Kapralov, Lee, Musco, Musco, and Sidford [17] and Kapralov, Mousavifar, Musco, Musco, Nouri, Sidford, and Tardos [19] showed that one can also compute a $(1+\epsilon)$-spectral sparsifier of an unweighted graph using $\tilde{O}(n\epsilon^{-2})$ linear measurements. Finally, a recent work by Filtser, Kapralov, and Nouri [10] showed that one can compute an $\tilde{O}(n^{\frac{2}{3}})$-spanner of an unweighted graph using $\tilde{O}(n)$ linear measurements. [10] also showed that one can compute an $\tilde{O}(n^{\frac{2}{3}(1-\alpha)})$-spanner of an unweighted graph using $\tilde{O}(n^{1+\alpha})$ linear measurements for any $\alpha \in [0,1]$, and conjectured that this tradeoff might be close to optimal.

In all of these works [2], [10], [17], [19], the authors also showed that their linear sketching algorithms can also be applied to computing graph sparsifiers of weighted graphs in *dynamic streams*, where the input graph is given by a stream of *insertions and deletions of weighted edges*, and the goal is to compute a sparsifier of the input graph using a small amount of space. In all these works, this is achieved by grouping the edge weights geometrically, and then applying the linear sketching algorithm for *unweighted graphs* to the subgraph induced by edges in each weight group. Note that this approach crucially requires that if an edge e is inserted with weight w_e, then any subsequent deletion of the edge e again reveals its weight and it must be *identical* to w_e. This is crucial to ensuring that each edge e is inserted into or deleted from the same geometric group.

However, the operation of grouping edges by weight is not linear, and as a result, the above approach for extending unweighted sketches to weighted ones is not implementable in the more general *turnstile streams*, where the graph is given as a stream of *arbitrary edge weight updates*. Surprisingly, little seems to be known about graph sparsification in this setting.

A. Our results

In this paper, we initiate the study of *weighted* graph sparsification by linear sketching. To state our results, we need to introduce some notation first. Let $w \in \mathbb{R}^{\binom{n}{2}}_{\geq 0}$ denote the weights of the edges of the input graph, where $w_e = 0$ means that there is no edge in the edge slot e. A *linear sketch of* $N = N(n)$ *measurements* consists of a (random) *sketching matrix* $\Phi \in \mathbb{R}^{N \times \binom{n}{2}}$, and a (randomized) *recovery algorithm* $\mathcal{A}$ that takes as input $\Phi w \in \mathbb{R}^N$ and outputs a sparsifier of the graph with edge weights w. Note that, by definition, the linear sketch is *non-adaptive*.

We focus on a natural class of linear sketches that we call *incidence sketches*, in which each row of the sketching matrix Φ is supported[2] on edges incident on a *single* vertex (which could be different for different rows). This class captures all linear sketches that are implementable in a distributed computing setting, where the edges are stored across n machines such that machine i has all edges incident on the i^{th} vertex (a.k.a.*simultaneous communication model*). Moreover, it also covers *all* aforementioned linear sketches used in previous works for unweighted cut sparsification [2], spectral sparsification [18], [19], and spanner computation [10].

We now present our results for computing these three kinds of sparsifiers in weighted graphs. When describing our results, we use $w_{\max}$ and $w_{\min}$ to denote the largest and the smallest non-zero edge weights, respectively, and always assume $w_{\max} \geq 1 \geq w_{\min}$. We also write $\tilde{O}(\cdot)$ to hide $\text{polylog}(n, \epsilon^{-1}, \frac{w_{\max}}{w_{\min}})$ factors.

a) Weighted cut sparsification.: We design an incidence sketch with a near-linear number of measurements for computing a $(1+\epsilon)$-cut sparsifier of a weighted graph.

Theorem I.1 (Algorithm for weighted cut sparsification)**.** *For any $\epsilon \in (0,1)$, there exists an incidence sketch with random sketching matrix $\Phi_1 \in \mathbb{R}^{N_1 \times \binom{n}{2}}$ satisfying $N_1 \leq \tilde{O}(n\epsilon^{-3})$ and a recovery algorithm $\mathcal{A}_1$, such that for any $w \in \mathbb{R}^{\binom{n}{2}}_{\geq 0}$, $\mathcal{A}_1(\Phi_1 w)$ returns, with probability $1 - \frac{1}{\text{poly}(n)}$, a $(1+\epsilon)$-cut sparsifier of the graph with edge weights w.*

Thus, we achieve a similar performance as the linear sketch for unweighted graphs given in [2], which uses $O(n\epsilon^{-2}\text{poly}(\log n))$ measurements. It is well known that even to detect the connectivity of a graph, $\Omega(n)$ linear measurements are needed. Therefore, our upper bound in Theorem I.1 is nearly optimal in n.

Similar to [2], [10], [13], [17], [19], by using Nisan's well known pseudorandom number generator [30], we can turn our linear sketching algorithm to a low space streaming algorithm.

Corollary I.2 (of Theorem I.1)**.** *There is a single pass turnstile streaming algorithm with $\tilde{O}(n\epsilon^{-3})$ space that, at any given point of the stream, recovers a $(1+\epsilon)$-cut sparsifier of the current graph with high probability.*

Note that in the turnstile model, the stream consists of arbitrary edge weight updates.

b) Weighted spectral sparsification.: We design an incidence sketch with about $n^{6/5}$ measurements for computing a $(1+\epsilon)$-spectral sparsifier of a weighted graph.

Theorem I.3 (Algorithm for weighted spectral sparsification)**.** *For any $\epsilon \in (0,1)$, there exists an incidence sketch with random sketching matrix $\Phi_2 \in \mathbb{R}^{N_2 \times \binom{n}{2}}$ satisfying $N_2 \leq \tilde{O}(n^{6/5}\epsilon^{-4})$ and a recovery algorithm $\mathcal{A}_2$, such that for any $w \in \mathbb{R}^{\binom{n}{2}}_{\geq 0}$, $\mathcal{A}_2(\Phi_2 w)$ returns, with probability $1 - \frac{1}{\text{poly}(n)}$, a $(1+\epsilon)$-spectral sparsifier of the graph with edge weights w.*

[1]Note that in a weighted graph, the length of a path is defined as the total edge weight along the path.

[2]Recall that the support of a vector is the set of indices at which it is non-zero.

Similar to the cut sparsification case, we have the following corollary:

Corollary I.4 (of Theorem I.3)**.** *There is a single pass turnstile streaming algorithm with $\tilde{O}(n^{6/5}\epsilon^{-4})$ space that, at any given point of the stream, recovers a $(1+\epsilon)$-spectral sparsifier of the current graph with high probability.*

We complement this result by showing that a superlinear number of measurements are indeed necessary for any incidence sketch to recover some $O(1)$-spectral sparsifier.

Theorem I.5 (Lower bound for weighted spectral sparsification)**.** *There exist constants $\epsilon, \delta \in (0,1)$ such that any incidence sketch of N measurements that computes a $(1+\epsilon)$-spectral sparsifier with probability $\geq 1-\delta$ on any w must satisfy $N \geq n^{21/20-o(1)}$.*

Note that this is in sharp contrast to the unweighted case, where a near-linear number of incidence sketch measurements are sufficient for computing an $O(1)$-spectral sparsifier [18], [19]. Theorem I.5 also draws a distinction between spectral sparsification and cut sparsification, as for the latter a near-linear number of measurements are enough even in the weighted case

c) Weighted spanner computation.: We first show that any $o(n^2)$ linear measurements can only recover a spanner of stretch that in general depends *linearly* on $\frac{w_{\max}}{w_{\min}}$. This differs fundamentally from the case of cut or spectral sparsification, where we can recover an $O(1)$-sparsifier, whose error is completely independent of $\frac{w_{\max}}{w_{\min}}$, using a sublinear-in-$n^2$ number of measurements. Specifically, we prove the following proposition.

Proposition I.6. *Any linear sketch (not necessarily an incidence sketch) of N measurements that computes an $o(\frac{w_{\max}}{w_{\min}})$-spanner with probability $\geq .9$ on any w must satisfy $N \geq \Omega(n^2)$.*

This proposition is a consequence of that edge weights are *proportional* to the edge lengths. More specifically, consider a complete graph where we weight a uniformly random edge by $w_{\min}$ and all other $\binom{n}{2}-1$ edges by $w_{\max}$. Then, while we can ignore the $w_{\min}$-weight edge in an $O(1)$-cut or spectral sparsifier, we have to include it in any $o(\frac{w_{\max}}{w_{\min}})$-spanner, since otherwise the shortest path length between its two endpoints would have been blown up by at least a factor of $\frac{2w_{\max}}{w_{\min}}$. Now note that, as the weight of this edge is smaller than all other edges, in order to find it we have to essentially recover all entries of the edge weight vector w, which inevitably requires $\Omega(n^2)$ linear measurements[3].

In light of the above proposition, we turn our focus to graphs with $\frac{w_{\max}}{w_{\min}} = O(1)$, and study the stretch's optimal dependence on n. On such graphs, the approach in [10] is able to obtain the following tradeoff between the stretch of the spanner and the number of linear measurements needed.

[3]In our actual proof of the proposition, we have to add some random Gaussian noise to each edge's weight for the lower bound to carry out.

Theorem I.7 (Algorithm for weighted spanner computation [10])**.** *For any constant $\alpha \in [0,1]$, there exists an incidence sketch with random sketching matrix $\Phi_3 \in \mathbb{R}^{N_3 \times \binom{n}{2}}$ satisfying $N_3 \leq \tilde{O}(n^{1+\alpha})$ and a recovery algorithm $\mathcal{A}_3$, such that for any $w \in \mathbb{R}^{\binom{n}{2}}_{\geq 0}$ with $\frac{w_{\max}}{w_{\min}} \leq O(1)$, $\mathcal{A}_3(\Phi_3 w)$ returns, with probability $1-\frac{1}{\text{poly}(n)}$, an $\tilde{O}(n^{\frac{2}{3}(1-\alpha)})$-spanner of the graph with edge weights w.*

[10] also conjectured that on unweighted graphs, to obtain a spanner of stretch $O(n^{\frac{2}{3}-\epsilon})$ for any constant $\epsilon > 0$, a superlinear number of measurements are needed for any linear sketch (in other words, the tradeoff is optimal at $\alpha = 0$). We make progress on this question by showing that this is indeed true for a natural class of linear sketches (i.e. incidence sketches) on "almost" unweighted graphs (i.e. those with $\frac{w_{\max}}{w_{\min}} = O(1)$). In fact, we show that in such a setting, the tradeoff obtained in the above theorem is *optimal for all* $\alpha < 1/10$.

Theorem I.8 (Lower bound for weighted spanner computation)**.** *For any constant $\alpha \in (0, 1/10)$, there exist constants $C \geq 1, \delta \in (0,1)$ such that any incidence sketch of N measurements that computes an $o(n^{\frac{2}{3}(1-\alpha)})$-spanner with probability $\geq 1-\delta$ on any w with $\frac{w_{\max}}{w_{\min}} \leq C$ must satisfy $N \geq n^{1+\alpha-o(1)}$.*

B. Roadmap

The rest of this paper is structured as follows. Section II gives an overview of our algorithms for weighted cut and spectral sparsification. Then in Section III, we give an overview of our lower bound for weighted spectral sparsification. Note that we do *not* give a separate overview of our lower bound for weighted spanner computation, because the ideas are similar to the ones described in Section III. The proofs of our main results are omitted here due to space limitation and are included in the full version.

II. Overview of Weighted Cut and Spectral Sparsification Algorithms

A. Overview of the algorithm for weighted cut sparsification

a) Recap of the unweighted cut sparsification algorithm in [2].: At a high-level, the approach taken is to reduce cut sparsification to (repeatedly) recovering a spanning forest of a subgraph of the input graph, obtained by sampling edges *uniformly* at some rate $p \in (0,1)$ known beforehand. This task is then further reduced to the task of sampling an edge connecting S to $\bar{S}$ for an arbitrary subset S of vertices, as this can be used to create a spanning forest by growing connected components.

Now to implement this latter task, in the sketching phase, we apply an ℓ_0-sampler sketch to the incidence vector of each vertex u (i.e. each column of the edge-vertex incidence matrix) in the *sub-sampled graph*. Then in the recovery phase, in order to recover an edge going out of a vertex set S, we add up the sketches of the vertices inside S. By linearity, this summed sketch is taken over the sum of the incidence

vectors of vertices inside S, and the latter contains exactly the edges going out of S, since the edges inside cancel out. As a result, we can recover an edge going out of S, and create a spanning forest of the sub-sampled graph. Note that this approach crucially utilizes the fact that the edges are sampled uniformly at a rate that is known *beforehand*. This means that we can sample all $\binom{n}{2}$ edge slots beforehand, and apply the linear sketch only to the sampled edge slots.

b) Our approach for weighted cut sparsification.: We also reduce the task to recovering a spanning forest in a sub-sampled graph. However, the latter graph is now obtained by sampling edges *non-uniformly*. Specifically, we need to recover a spanning forest in a subgraph obtained by sampling each edge e with probability $\min\{w_e p, 1\}$ for some parameter $p \in (0,1)$ that is known beforehand. Therefore, in order to apply the idea as in the unweighted case, we will now need to design a variant of ℓ_0-sampler that, given a vector $x \in \mathbb{R}^N_{\geq 0}$ and a parameter $p \in (0,1)$, recovers a nonzero entry of x after each entry $i \in [N]$ is sampled with probability $\min\{x_i p, 1\}$. We call such a sampler "weighted edge sampler".

Note that the edge weights are *not* known to us beforehand, so we cannot sample the edge slots with our desired probabilities as in the unweighted graph case. We instead build such a weighted edge sampler using a *rejection sampling* process, in which we sample edges uniformly at $\tilde{O}(1)$ geometric rates, but use ℓ_1*-samplers* to try recovering edges at each rate, and only output a recovered edge e if the sampling rate $\approx w_e p$. We then show that with high probability, we can efficiently find a desired edge.

Roughly, our analysis involves proving that there exists a geometric rate q such that, after uniformly sampling edges at rate q, the total weight of edges e satisfying $w_e p \approx q$ accounts for a large portion of that of all sampled edges. As a result, by using a few independent ℓ_1-samplers, we can find one such edge with high probability.

B. Overview of the algorithm for weighted spectral sparsification

As in previous linear sketches for unweighted graphs [17], [19], the key task is to recover edges with $\tilde{\Omega}(1)$ effective resistances (or in weighted case, $\tilde{\Omega}(1)$-leverage scores), which we refer to as *heavy edges*. The high-level idea used in previous works is to (i) compute, for each vertex pair s,t, a set of vertex potentials $x_{s,t} \in \mathbb{R}^n$ induced by an electrical flow from s to t, and then (ii) apply an ℓ_2-heavy hitter to $B_G x_{s,t} \in \mathbb{R}^{\binom{n}{2}}$ to try recovering the edge (s,t), where B_G is the edge-vertex incidence matrix of G. They achieve (i) by simulating an iterative refinement process in [28]. To achieve (ii), they make a key observation that

$$\|B_G x_{s,t}\|_2^2 = x_{s,t}^T B_G^T B_G x_{s,t} = x_{s,t}^T L_G x_{s,t}$$

is the energy of $x_{s,t}$, and the entry of $B_G x_{s,t}$ indexed by edge (s,t) is $(B_G x_{s,t})_{(s,t)} = b_{s,t}^T x = x_s - x_t$. Therefore by the energy minimization characterization of effective resistances, whenever the effective resistance between s,t is $b_{s,t}^T L_G^\dagger b_{s,t} \geq \tilde{\Omega}(1)$, we have

$$(B_G x_{s,t})_{(s,t)}^2 \geq \tilde{\Omega}(1) \|B_G x_{s,t}\|_2^2,$$

and hence the entry (s,t) is an ℓ_2-heavy hitter.

However, when the graph is weighted, we are only allowed to access the graph through linear measurements on its weight vector w_G. As a result, we can only apply ℓ_2-heavy hitters to $W_G B_G x_{s,t}$, whose squared ℓ_2-norm is $x_{s,t}^T B_G^T W_G^2 B_G x_{s,t}$. Now notice that $B_G^T W_G^2 B_G$ is the Laplacian matrix of a "squared" graph (call it G^{sq}), which has the same edges as G, but whose edges are weighted by w_e^2 as opposed to w_e. Therefore, we will be recovering edges that are heavy in G^{sq} instead of in G if we apply the same approach as in previous works. Unfortunately, a heavy edge in G is not in general heavy in G^{sq}, since the energy on the edges with very large weights will blow up when we square the edge weights (i.e. $w_e^2(x_u - x_v)^2 \gg w_e(x_u - x_v)^2$), and hence make the total energy grow unboundedly.

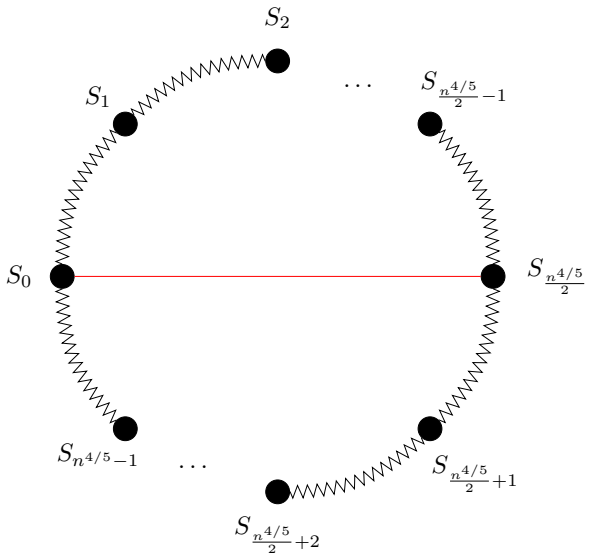

Figure 1. A block cycle graph on n vertices. Each S_i represents a block of $n^{1/5}$ vertices connected by a clique and each zigzag represents the edges of a complete bipartite graph between adjacent blocks. The red edge represents the crossing edge. All edges along the cycle have weights $n^{2/5}$, and the crossing edge has weight 1.

To see an intuitive example, suppose G is a "block cycle graph" on n vertices whose edges are generated as follows (see also Figure 1):

1) Partition the vertices into $n^{4/5}$ blocks $S_0, \ldots, S_{n^{4/5}-1}$, each with $n^{1/5}$ vertices.
2) For each $0 \leq i < n^{4/5}$, add on S_i a complete graph of $n^{1/5}$ vertices with edge weights $n^{2/5}$, i.e. $n^{2/5} K_{n^{1/5}}$.
3) For each $0 \leq i < n^{4/5}$, add on (S_i, S_{i+1}) a complete bipartite graph of $2n^{1/5}$ vertices with edge weights $n^{2/5}$ and bipartition (S_i, S_{i+1})[4], i.e. $n^{2/5} K_{n^{1/5}, n^{1/5}}$.

[4]Note that we consider $i+1$ as 0 when $i = n^{4/5} - 1$.

4) Finally, add a "crossing edge" e^* of weight 1 between a randomly chosen vertex pair s, t.

We note that, in this construction, the crossing edge e^* spans $\Omega(n^{4/5})$ consecutive blocks, Note that, typically, the crossing edge e^* spans $\Omega(n^{4/5})$ consecutive blocks, and therefore has effective resistance (and also leverage score) $\Omega(1)$.

Proposition II.1. *If $s \in S_i, t \in S_j$ such that $\min\{|i-j|, n^{4/5} - |i-j|\} \geq \Omega(n^{4/5})$, then the effective resistance of e^* satisfies $r_{e^*} \geq \Omega(1)$.*

Proof. Let s, t be the endpoints of the crossing edge e^*. By the energy minimization characterization of effective resistances, it suffices to show that there is a set of vertex potentials whose normalized energy with respect to s, t is $O(1)$. Specifically, consider the set of potentials $x \in \mathbb{R}^n$ such that $x_u = \frac{i}{n^{4/5}}$ for all $u \in S_i$. Then we have $x_s - x_t = \Theta(1)$, and the total energy is $\sum_{e=(u,v)} w_e(x_u - x_v)^2 = n^{6/5} \cdot n^{2/5}(\Theta(n^{-4/5}))^2 + \Theta(1) = \Theta(1)$, as desired. □

However, in the squared graph G^{sq}, all edge weights along the cycle are blown up by a factor of $n^{2/5}$, and thus e^* only has leverage score $O(n^{-2/5})$ in G^{sq}. To recover in a vector x every entry with ℓ_2-contribution $\geq n^{-2/5}\|x\|_2^2$, one will need an $\Omega(n^{-2/5})$ factor blowup in the number of linear measurements, resulting in a total of $n^{7/5}$ measurements needed to recover e^*.

We can in fact improve the number of linear measurements needed for recovering e^* to $\tilde{O}(n^{6/5})$ using the *vertex sampling* trick, an idea first used in [10] for sketching spanners. Namely, consider sampling a vertex set $C \subset V$ by including each vertex with probability $n^{-1/5}/100$, and looking at the vertex-induced subgraph $G^{\text{sq}}[C]$. Then one can show that, conditioned on $e^* \in G^{\text{sq}}[C]$, with constant probability, the two endpoints of e^* will be disconnected in $G^{\text{sq}}[C] \backslash e^*$. As a result, the leverage score of e^* becomes 1 in $G^{\text{sq}}[C]$, and we can recover e^* by recovering heavy edges in $G^{\text{sq}}[C]$, which, as will show, can be done using $\tilde{O}(|C|) \approx \tilde{O}(n^{4/5})$ measurements. Since $e^* \in G[C]$ with probability $\approx \frac{1}{n^{2/5}}$, repeating this sampling process independently for $\tilde{O}(n^{2/5})$ times allows us to recover e^* in at least one vertex induced subgraph. This results in a linear sketch of $\tilde{O}(n^{6/5})$ measurements.

What if we slightly increase each block's size to $n^{1/5+\delta}$ and decrease the edge weights along the cycle to $n^{2/5-3\delta}$? While one can still verify that the crossing edge e^* has leverage score $\Omega(1)$, applying the same vertex sampling process as above will not disconnect the endpoints of e^* with $\Omega(1)$ probability. However, one can alternatively show that, with constant probability, the number of edges along the cycle reduces by a factor of $n^{2/5}$. Since now the energy of each edge only blows up by a factor smaller than $n^{2/5}$ in G^{sq}, this will also make the leverage score of e^* become $\Omega(1)$ in $G^{\text{sq}}[C]$, and thus we can apply the same linear sketch of $\tilde{O}(n^{6/5})$ measurements.

The above warm-up seems to suggest that the sampling rate of $\approx n^{-1/5}$ is a sweet spot for recovering heavy edges in any graphs with the block cycle structure. Indeed, we prove a key vertex sampling lemma showing that in *any* weighted graph G, a heavy edge e in G is also likely heavy in a vertex-induced subgraph of G^{sq} obtained by sampling vertices at rate $\approx n^{-1/5}$. This is proved by carefully analyzing the structures of the edges of different weights after vertex sampling, and then explicitly constructing a set of vertex potentials with small total energy in the induced subgraph. Finally, by integrating this lemma into an iterative refinement process in [28] (as the authors did in [17], [19]) and a spectral sparsification algorithm in [21], we are able to recover a spectral sparsifier of G using $\tilde{O}(n^{6/5})$ linear measurements.

We note that this method of recovering heavy edges by vertex sampling is inspired by the one used in [17] for spanners. However, for spectral sparsification, the correctness of such a method follows from fairly different reasoning, and the proof is arguably more involved.

III. Overview of Lower Bound for Weighted Spectral Sparsification

In this section we give an overview of our lower bound for weighted spectral sparsification (Theorem I.5). We prove our lower bound on a family of hard instances that turn out to have the exact same structure as the one in Figure 1, which we used to illustrate the difficulty of recovering spectral sparsifiers in weighted graphs. Specifically, our hard instances are weighted "block cycle graphs" plus an extra *crossing edge* that is included with probability $1/2$. In a block cycle graph, the vertices are partitioned into blocks that are arranged in a cyclic manner. Each block is a complete graph, and the vertices of adjacent blocks are connected by a complete bipartite graph. Here we draw all edge weights from Gaussian distributions and permute the vertices uniformly at random. On such graphs, for a suitable choice of edge weights, computing a spectral sparsifier essentially boils down to detecting the presence/absence of the crossing edge. We then show that for the latter task, the success probability of any incidence sketch can be bounded by the "effective resistance" of the crossing edge in a *matrix-weighted* graph, where the matrix weights are in turn determined by the sketching matrix.

We note that this family of hard instances has a similar structure to the ones conjectured in [10]. However, instead of using Bernoulli distributions on the edges as suggested in [10], we use Gaussian distributions, which makes it easier to build the connection to effective resistances.

In order to show that the effective resistance is small for any incidence sketch with a limited number of measurements, we develop a number of new tools for analyzing such matrix-weighted graphs. Most importantly, we present (i) a matrix-weighted analog of the widely used expander decomposition of ordinary graphs [12], [15], [33], and (ii) a proof that a random vertex-induced subgraph of a matrix-weighted expander is also an expander with high probability. We highly recommend reading Section III-C to get intuition on why these two techniques are useful, where we use the ordinary graph version of (i) and (ii) to prove a lower bound for a simple class of sketches.

The rest of this section is structured as follows. In Section III-A we describe the distribution from which we generate our hard instances. In Section III-B we explain how we bound the success probability of an incidence sketch by the effective resistance in a matrix-weighted graph. In Section III-C we prove, as a warm-up, a lower bound for a simple class of sketches, where we only need to analyze the effective resistances in *ordinary graphs*. Finally in Section III-D we outline our proof for arbitrary incidence sketches, which requires analyzing matrix-weighted graphs.

A. The hard distribution

We first state how we generate the input weighted graph $G = (V, E, w)$. Let n be the number of vertices and define $s \stackrel{\text{def}}{=} n^{1/5}$ and $\ell \stackrel{\text{def}}{=} n^{4/5}$. We choose a random permutation $\pi : 1..n \to 1..n$ and construct a block cycle graph as follows. The i-th block (where $0 \le i < \ell$) consists of vertices $\pi(si + 1), \ldots, \pi(si + s)$. For simplicity we denote the a-th vertex in the i-th block (i.e. $\pi(si + a)$) as $u_{i,a}$. The block index i will always be modulo ℓ implicitly. We then add a complete graph to each block, and a complete bipartite graph between each pair of adjacent blocks. Namely, for each $0 \le i < \ell$, we add a graph G_i with edges connecting $u_{i,a}, u_{i,b}$ for all $a < b \in \{1, \ldots, s\}$, and add another bipartite graph $G_{i,i+1}$ with edges connecting $u_{i,a}, u_{i+1,b}$ for all $a, b \in \{1, \ldots, s\}$. Finally, with probability $1/2$, we add an edge between vertices $\pi(1)$ and $\pi(n/2+1)$ (assume n is even). We refer to this edge as the *crossing edge* with respect to π and any other edge in G as a *non-crossing edge* with respect to π. We will omit "with respect to π" when the underlying permutation π is clear.

We next describe how the edge weights are determined. The weights of all non-crossing edges are drawn independently from $\mathcal{N}(8n^{2/5}, n^{4/5} \log^{-1} n)$ (the Gaussian distribution with mean $8n^{2/5}$ and variance $n^{4/5} \log^{-1} n$). The weight of the crossing edge is drawn from the standard Gaussian $\mathcal{N}(0, 1)$. If the crossing edge has negative weight, we say the input is *invalid*, and accept any sketch as a valid sketch. Our goal will be to detect the presence/absence of the crossing edge with high probability.

In the following, we will call the conditional distribution on the presence of the crossing edge the Yes *distribution*, and call the conditional distribution on the absence of the crossing edge the No *distribution*. We then show that with high probability, the effective resistance of the crossing edge is large, and therefore any linear sketch for computing spectral sparsifiers must distinguish between the two distributions with good probability.

Proposition III.1. *With probability at least $1 - 1/n$, all non-crossing edges have weights in the range $[4n^{2/5}, 12n^{2/5}]$, and as a result the effective resistance between vertices $\pi(1)$ and $\pi(n/2 + 1)$ is at least $1/48$ in a* No *instance.*

Proposition III.2. *Any linear sketch that can compute a 1.0001-spectral sparsifier with probability 0.9 can distinguish between the* Yes *and* No *distributions with probability 0.6.*

The first proposition follows from an application of the Chernoff bound. The proof of the second proposition is deferred to the full version.

In the following, we will assume, for ease of our analysis, that the sketch will be given the permutation π *after* computing the linear sketch. That is, the recovery algorithm $\mathcal{A}$ takes as input both Φw and π. We will show that even with this extra piece of information, any incidence sketch with $n^{21/20-\epsilon}$ measurements for constant $\epsilon > 0$ cannot distinguish between the Yes and No distributions with high probability.

B. A bound on the success probability via effective resistance

We first show that for our lower bound instance, any incidence sketch can be reduced to a more restricted class of linear sketches by only increasing the number of measurements by an $O(\log n)$ factor. Specifically, let us fix an arbitrary orientation of the edges, and consider sketches taken over the weighted signed edge-vertex incidence matrix $B^w \in \mathbb{R}^{\binom{n}{2} \times n}$, where the latter is given by

$$B^w_{eu} = \begin{cases} w_e & e \in E \text{ and } u \text{ is } e\text{'s head} \\ -w_e & e \in E \text{ and } u \text{ is } e\text{'s tail} \\ 0 & \text{otherwise.} \end{cases}$$

That is, the algorithm must choose a (random) sketching matrix $\Phi \in \mathbb{R}^{k \times \binom{n}{2}}$ with the e^{th} column $\phi_e \in \mathbb{R}^k$ corresponding to the edge slot e. The sketch obtained is then $\Phi B^w \in \mathbb{R}^{k \times n}$. Notice that the total number of measurements in ΦB^w is kn, as each vertex applies the sketching matrix Φ to its incident edges. Let us call this class of sketch *signed sketches*. By Yao's minimax principle [35], to prove a lower bound for distinguishing the Yes and No distributions, it suffices to focus on deterministic sketches. The proof of the proposition below appears in the full version.

Proposition III.3 (Reduction to signed sketches)**.** *Consider any incidence sketch of N measurements with a deterministic sketching matrix $\Phi \in \mathbb{R}^{N \times \binom{n}{2}}$ and a recovery algorithm $\mathcal{A}$ that, given Φw and π, distinguishes between the* Yes *and* No *distributions with probability at least 0.6. Then there exists a signed sketch with a sketching matrix $\Phi' \in \mathbb{R}^{k \times \binom{n}{2}}$, where $k = O(1) \cdot \max\{1, \frac{N \log n}{n}\}$, and a recovery algorithm $\mathcal{A}'$ that, given $\Phi' B^w$ and π, distinguishes between the* Yes *and* No *distributions with probability at least 0.55.*

Let us now fix a sketching matrix $\Phi \in \mathbb{R}^{k \times \binom{n}{2}}$ and aim to obtain an upper bound on the success probability of any signed sketch using Φ. For notational convenience, let us write $(\Phi B^w)_{\text{yes}}$ to denote ΦB^w conditioned on the presence of the crossing edge and $(\Phi B^w)_{\text{no}}$ to denote ΦB^w conditioned on the absence of the crossing edge. We will also write $(\Phi B^w)_{\pi,\text{yes}}$ or $(\Phi B^w)_{\pi,\text{no}}$ to denote an extra conditioning on the permutation being π in addition to the presence/absence of the crossing edge. Then to bound the success probability of any signed sketch using Φ , it suffices to show that the total variation distance (TV-distance) between $(\Phi B^w)_{\text{yes}}$ and $(\Phi B^w)_{\text{no}}$ is small.

To state our upper bound on the TV-distance, we need to first introduce some notation. For an edge (u, v), define $b_{uv} \in \mathbb{R}^{nk}$ by writing it as a block vector (with block size k) as follows:

$$b_{uv} = \begin{pmatrix} 0 \\ \vdots \\ \phi_{uv} \\ 0 \\ \vdots \\ -\phi_{uv} \\ 0 \\ \vdots \end{pmatrix} \begin{matrix} \\ \\ u^{\text{th}} \text{ block} \\ \\ \\ v^{\text{th}} \text{ block} \\ \\ \end{matrix} \in \mathbb{R}^{nk}, \tag{1}$$

where $\phi_{uv} \in \mathbb{R}^k$ is the column of Φ corresponding to the edge slot (u, v). For a permutation π, we then define $L_\pi = \sum_{\text{non-crossing } (u,v)} n^{4/5} \log^{-1} n \, b_{uv} b_{uv}^T$. The following proposition is essentially a consequence of the result in [8], which bounds the TV-distance between multivariate Gaussians with the same mean. We give its proof in the full version. Note that we use † to denote taking the Moore-Penrose pseudoinverse of a matrix.

Proposition III.4. *For any permutation π such that $b_{\pi(1)\pi(n/2+1)}$ is in the range[5] of L_π,*

$$d_{\text{TV}}((\Phi B^w)_{\pi,\text{yes}}, (\Phi B^w)_{\pi,\text{no}}) \leq O(1) \cdot \min\left\{1, b_{\pi(1)\pi(n/2+1)} L_\pi^\dagger b_{\pi(1)\pi(n/2+1)}\right\}.$$

Our plan is then to show that $b_{\pi(1)\pi(n/2+1)} L_\pi^\dagger b_{\pi(1)\pi(n/2+1)}$ is small on average for every choice of a signed sketch with $k = n^{1/20-\epsilon}$ for constant $\epsilon > 0$.

Note that if $k = 1$ and each $\phi_{uv} \in \{0, 1\}$, then L_π is exactly the Laplacian matrix of the graph (call it $\mathcal{H}_\pi$) that is formed by the non-crossing edges (u, v) such that $\phi_{uv} = 1$, where each edge is weighted by $n^{4/5} \log^{-1} n$. Therefore, $b_{\pi(1)\pi(n/2+1)} L_\pi^\dagger b_{\pi(1)\pi(n/2+1)}$ is the effective resistance between $\pi(1)$ and $\pi(n/2 + 1)$ in $\mathcal{H}_\pi$, if $\phi_{\pi(1)\pi(n/2+1)} \neq 0$ (otherwise $b_{\pi(1)\pi(n/2+1)}$ is the zero vector).

In fact, to get a quick intuition as to why we should expect the effective resistance between $\pi(1)$ and $\pi(n/2 + 1)$ to be small, let us assume $\phi_{uv} = 1$ for all edge slots (u, v). Then, for any permutation π, $\mathcal{H}_\pi$ is the graph formed by *all* non-crossing edges, each weighted by $n^{4/5} \log^{-1} n$. Note that these weights are about $n^{2/5}$ times larger than the weights $\Theta(n^{2/5})$ in the actual input graph (Proposition III.1). As a result, the effective resistance between $\pi(1)$ and $\pi(n/2+1)$ is about $n^{2/5}$ times smaller than the effective resistance between them in the input graph (the former roughly equals $n^{-2/5}$).

When $k > 1$, we can view L as the Laplacian of a *matrix-weighted* graph (again, call it $\mathcal{H}_\pi$) formed by the non-crossing edges, where each edge (u, v) has a $k \times k$ matrix weight $n^{4/5} \log^{-1} n \, \phi_{uv} \phi_{uv}^T$. Now $b_{\pi(1)\pi(n/2+1)} L_\pi^\dagger b_{\pi(1)\pi(n/2+1)}$ can be seen as the (generalized) effective resistance between $\pi(1)$ and $\pi(n/2 + 1)$ in $\mathcal{H}_\pi$.

[5]Recall that the range of a symmetric matrix is the linear span of its columns.

C. Warm-up: one-row signed sketches have small TV-distance

As a warm-up, we show that for any signed sketch, in the case that $k = 1$ and the sketching matrix Φ has $0/1$ entries, we have, for any constant $\epsilon > 0$,

$$\mathbb{E}_\pi \left[d_{\text{TV}}\left((\Phi B^w)_{\pi,\text{yes}}, (\Phi B^w)_{\pi,\text{no}}\right)\right] \leq \frac{1}{n^{1/5-O(\epsilon)}}. \tag{2}$$

By Proposition III.4, we know that $d_{\text{TV}}\left((\Phi B^w)_{\pi,\text{yes}}, (\Phi B^w)_{\pi,\text{no}}\right)$ can be bounded by the effective resistance between $\pi(1), \pi(n/2 + 1)$ in $\mathcal{H}_\pi$ if $\phi_{\pi(1)\pi(n/2+1)} = 1$, and is zero otherwise. Here $\mathcal{H}_\pi$ is formed by the non-crossing edges whose $\phi_{uv} = 1$, where each edge (u, v) has scalar weight $n^{4/5} \log^{-1} n$. We can focus on the Φ's whose number of nonzero entries is at least $n^{9/5+\epsilon}$, since otherwise

$$\Pr_\pi\left[\phi_{\pi(1)\pi(n/2+1)} = 1\right] = \frac{\text{nnz}(\Phi)}{\binom{n}{2}} \leq \frac{1}{n^{1/5-O(\epsilon)}},$$

and we would already have our desired result (2).

Our proof of (2) will rely on decomposing $\mathcal{H}_\pi$ into expanders with large minimum degree. Since $\mathcal{H}_\pi$'s edges all have the same weight $n^{4/5} \log^{-1} n$, it is more convenient to work with the *unweighted version of* $\mathcal{H}_\pi$, which we denote by H_π. We now briefly review the definition of unweighted expanders, as well as state a known expander decomposition lemma that we will utilize.

Definition III.1 (Expander)**.** An unweighted graph $H = (V, E)$ is a ζ-expander for some $\zeta \in [0, 1]$ if its conductance is at least ζ, namely, for every nonempty $S \subset V$, we have

$$|E(S, V - S)| \geq \zeta \cdot \min\left\{\text{vol}(S), \text{vol}(V - S)\right\},$$

where $\text{vol}(S)$ is the total degree of vertices in S.

Note that in the lemma below, we slightly abuse the notion of "regular graphs". Specifically, we will say a graph is regular if its minimum vertex degree $d_{\min}$ is not much smaller than the average degree d.

Lemma III.5 (Almost regular expander decomposition, see e.g. [16])**.** *Given an unweighted graph $H = (V, E)$ with average degree $d \geq 16$, there exists a subgraph $I = (U, F)$ where $U \subseteq V$ and $F \subseteq E$ such that I is a $\frac{1}{16 \log n}$-expander with minimum degree $d_{\min} \geq \frac{d}{16}$.*

We will also need the following lemma, which shows that a random vertex-induced subgraph of an expander with large minimum degree is almost certainly an expander. We give the proof of this lemma in the full version. To the best of our knowledge, even this result was not known before.

Lemma III.6 (Expanders are preserved under vertex sampling)**.** *There exists a $\theta = \theta(n) = n^{o(1)}$ with the following property. Consider an unweighted $\frac{1}{16 \log n}$-expander $H = (V, E)$ with minimum degree $d_{\min} \geq 4 \cdot 10^6 \cdot \theta(n)$. For any $s \geq \frac{4 \cdot 10^6}{d_{\min}} \cdot \theta(n) \cdot n$, let $C \subseteq V$ be a uniformly random*

vertex subset of size s. Then with probability at least $1-1/n^7$, the vertex-induced subgraph $H[C]$ is a $\frac{1}{n^{o(1)}}$-expander with minimum degree at least $\frac{s}{2n}\cdot d_{\min}$.

Proof of (2) using Lemmas III.5,III.6. As argued above we can assume w.l.o.g. that $\mathrm{nnz}(\Phi) \geq n^{9/5+\epsilon}$. We want to obtain, for each edge slot e satisfying $\phi_e = 1$, conditioned on e being the crossing edge w.r.t. π, an upper bound (call it u_e) on the typical effective resistance between the endpoints of e in the graph $\mathcal{H}_\pi$. In other words, conditioned on e being the crossing edge, u_e should be an upper bound on the effective resistance between the endpoints of e in $\mathcal{H}_\pi$ with high probability over π. Then the total variation distance between $(\Phi B^w)_{\text{yes}}$ and $(\Phi B^w)_{\text{no}}$ can be bounded by

$$\mathbb{E}_\pi\left[d_{\text{TV}}\left((\Phi B^w)_{\pi,\text{yes}}, (\Phi B^w)_{\pi,\text{no}}\right)\right] \leq O(1)\cdot\frac{1}{\binom{n}{2}}\sum_{e:\phi_e=1} u_e. \tag{3}$$

To obtain the u_e's, let us define the unweighted graph $H_\phi = (V, E_\phi)$ where E_ϕ contains *all* edges e whose $\phi_e = 1$ (including the ones not present in the input graph, i.e. $|E_\phi| = \mathrm{nnz}(\Phi)$). Now consider the following process, where we repeatedly delete an expander subgraph from H_ϕ and obtain u_e's for the edges in the expander.

1) While $|E_\phi| \geq 10^9 n^{9/5+\epsilon}$:
   a) Find a subgraph $I = (U, F)$ of $H_\phi = (V, E_\phi)$ that is a $\frac{1}{16\log n}$-expander with minimum degree $d_{\min} \geq \frac{|E_\phi|}{8n}$ (existence is guaranteed by Lemma III.5).
   b) For each edge $f \in F$, let $u_f \leftarrow \left(\frac{10^9 n^{9/5+\epsilon}}{|E_\phi|}\right)^2$.
   c) Delete the edges in F from H_ϕ by letting $E_\phi \leftarrow E_\phi \setminus F$.
2) Let $u_f \leftarrow 1$ for all f in the remaining E_ϕ.

To show that u_e's are valid upper bounds, let us consider a fixed iteration of the while loop. For $i = 0, \ldots, n^{4/5}-1$, let U_i denote the vertices in I that are in the i^{th} block of the input block cycle graph:

$$U_i \stackrel{\text{def}}{=} U \cap \left\{\pi(n^{1/5}i+1), \ldots, \pi(n^{1/5}i+n^{1/5})\right\}.$$

Then by Chernoff bounds, with probability at least $1-1/n^5$ over the random choice of π, we have $|U_i| \geq \frac{|U|}{2n^{4/5}} \geq \frac{4\cdot 10^6}{d_{\min}}\cdot|U|^{1+\epsilon}$. Then by invoking Lemma III.6, with probability at least $1-1/n^4$ over π, all vertex-induced subgraphs $I[U_i \cup U_{i+1}]$ are $\frac{1}{n^{o(1)}}$-expanders with minimum degree at least $\frac{|E_\phi|}{16n^{9/5}}$. Using this fact, we obtain the following claim, whose proof appears in the full version.

Claim III.7. For each edge $f \in F$, conditioned on f being the crossing edge, with probability at least $1-1/n^2$ over π, the effective resistance between the endpoints of f in $\mathcal{H}_\pi$ is at most u_f.

Now let us divide the above process for obtaining u_e's into $O(\log n)$ phases, where in phase $i \in \{1, \ldots, O(\log n)\}$, we have $|E_\phi| \in (\binom{n}{2}/2^i, \binom{n}{2}/2^{i-1}]$. Then we have

$$\begin{aligned}\sum_{e:\phi_e=1} u_e &= \sum_{e:\phi_e=1}\left(\frac{10^9 n^{9/5+\epsilon}}{|E_\phi|}\right)^2\\ &\leq n^{O(\epsilon)}\cdot\sum_{i=1}^{O(\log n)}\frac{\binom{n}{2}}{2^i}\cdot\left(\frac{2^i}{n^{1/5}}\right)^2\\ &\leq n^{8/5+O(\epsilon)}\sum_{1}^{O(\log n)} 2^i \leq n^{9/5+O(\epsilon)}\end{aligned}$$

where in the first line we have used $n^{O(\epsilon)}$ to hide the constant factors, and the last inequality holds since in the last phase we have $2^i \leq n^{1/5}$. Plugging this into (3) finishes the proof. □

D. The general case: proof of Theorem I.5

Note that even though for $k = 1$, the TV-distance is $\tilde{O}(n^{-1/5})$, this does not imply that k must be large for the TV-distance to become $\Omega(1)$.

By Proposition III.3, in order to prove Theorem I.5, it suffices to prove the following:

Theorem III.8. *For any fixed sketching matrix $\Phi \in \mathbb{R}^{k\times\binom{n}{2}}$ where $k \leq n^{1/20-\epsilon}$ for some constant $\epsilon > 0$, we have*

$$\mathbb{E}_\pi\left[d_{\text{TV}}\left((\Phi B^w)_{\pi,\text{yes}}, (\Phi B^w)_{\pi,\text{no}}\right)\right] \leq o(1).$$

By Proposition III.4, our goal is to bound the "effective resistance" $b_{\pi(1)\pi(n/2+1)}L_\pi^\dagger b_{\pi(1)\pi(n/2+1)}$ between vertices $\pi(1), \pi(n/2+1)$ in the matrix-weighted graph $\mathcal{H}_\pi$ consisting of the non-crossing edges, where edge (u,v) has matrix weight $n^{4/5}\log^{-1} n\,\phi_{uv}\phi_{uv}^T \in \mathbb{R}^{k\times k}$. We will do so by (significantly) generalizing our previous approach based on expander decomposition for ordinary graphs in Section III-C. Our approach for the $k = 1$ case essentially consists of two steps: (i) decomposing the graph H_ϕ into large expander subgraphs and (ii) proving that a random vertex induced subgraph of an expander is still an expander.

First note that there does not appear to be a combinatorial analog of conductance in matrix-weighted graphs, which suggests that we should define expanders in an algebraic way. Let us first recall the algebraic characterization of expanders for ordinary, unweighted graphs. The definition is based on eigenvalues of the *normalized Laplacian* of the graph, which is given by $N = D^{-1/2}LD^{-1/2}$, where D is a diagonal matrix with D_{uu} equal to the degree d_u of u.

Definition III.2 (Algebraic definition of ordinary, unweighted expanders)**.** An unweighted graph H is a ζ-expander for some $\zeta \in [0,1]$ if the smallest nonzero eigenvalue of its normalized Laplacian matrix N is at least ζ.

By Cheeger's inequality [3], for $\zeta \geq \tilde{\Omega}(1)$, this definition translates to that the graph H is a union of vertex-disjoint combinatorial expanders, each with conductance $\tilde{\Omega}(1)$. To come up with an analogous definition for matrix-weighted graphs, let us first define their associated matrices formally.

a) Matrices associated with matrix-weighted graphs.: We consider a $k \times k$ matrix-weighted graph $H = (V, E)$ with $|V| = n$. For each edge $(u, v) \in E$, there is a vector $\phi_{uv} \in \mathbb{R}^k$, indicating that (u, v) is weighted by the $k \times k$ rank-1 matrix $\phi_{uv}\phi_{uv}^T$.

Definition III.3 (Degree matrices)**.** For a vertex u, its generalized degree is given by

$$D_u = \sum_{u \sim v} \phi_{uv}\phi_{uv}^T \in \mathbb{R}^{k \times k}.$$

We then define the $nk \times nk$ degree matrix D as a block diagonal matrix (with block size $k \times k$), with the u^{th} block on the diagonal being $D_{uu} = D_u \in \mathbb{R}^{k \times k}$.

Definition III.4 (Laplacian matrices)**.** The Laplacian matrix is given by $L = \sum_{(u,v) \in E} b_{uv} b_{uv}^T$, where b_{uv}'s are defined in (1).

We will call b_{uv} the *incidence vector* of edge (u, v).

Definition III.5 (Normalized Laplacian matrices)**.** The normalized Laplacian matrix is given by $N \stackrel{\text{def}}{=} D^{\dagger/2} L D^{\dagger/2}$. Equivalently, we have $N = \sum_{(u,v) \in E} D^{\dagger/2} b_{uv} b_{uv}^T D^{\dagger/2}$.

We will call $D^{\dagger/2} b_{uv}$ the *normalized incidence vector* of edge (u, v).

The following proposition says that similar to scalar-weighted graphs, the eigenvalues of the normalized Laplacian of a matrix-weighted graph are also between $[0, 2]$. The proof this proposition appears in the full version.

Proposition III.9. *The eigenvalues of N are between $[0, 2]$.*

Now, a first attempt might be to define matrix-weighted expanders to also be graphs whose normalized Laplacians' nonzero eigenvalues are large, and then try to decompose any matrix-weighted graph into large expander subgraphs. However, we show that the latter goal may not be achievable in general, by presenting in the full version a hard instance, for which any large subgraph has a small nonzero eigenvalue.

b) Our approach.: In light of the hard instance, we loosen the requirement of being an expander by allowing small eigenvalues, but requiring instead that each edge, compared to the average, does not have too large "contribution" to the small eigenvectors. Formally, we want that every edge's normalized incidence vector has small (weighted) projection onto the bottom eigenspace. We will also need an analog of "almost regularity", which for ordinary, unweighted graphs says that the minimum degree is large. We give the formal definition of an almost regular matrix-weighted expander below.

Definition III.6 (Almost regular matrix-weighted expanders)**.** For a $k \times k$ matrix-weighted graph H, let $\lambda_1 \leq \ldots \leq \lambda_{nk}$ be the eigenvalues of its normalized Laplacian N, and let $f_1, \ldots, f_{nk} \in \mathbb{R}^{nk}$ be a set of corresponding orthonormal eigenvectors. We say H is a (γ, ζ, ψ)-almost regular expander if

1) (γ-almost regularity) For every vertex u and every incident edge $(u, v) \in E$, we have
$$\phi_{uv}^T D_u^\dagger \phi_{uv} \leq \frac{\gamma \cdot k}{n}. \tag{4}$$
2) ((ζ, ψ)-expander) For every edge $(u, v) \in E$ we have
$$\left(D^{\dagger/2} b_{uv}\right)^T \left(\sum_{i:\lambda_i \in (0,\zeta]} \frac{1}{\lambda_i} f_i f_i^T\right) D^{\dagger/2} b_{uv} \leq \frac{\psi \cdot k^2}{n^2}. \tag{5}$$

The LHS of (4) is the so-called *leverage score* of ϕ_{uv} w.r.t. D_u. It is known that the sum of leverage scores equals the rank of the matrix:

Proposition III.10. *For any fixed vertex u, $\sum_{(u,v) \in E} \phi_{uv}^T D_u^\dagger \phi_{uv} = \operatorname{rank}(D_u)$.*

Since D_u is a $k \times k$ matrix, we have $\operatorname{rank}(D_u) \leq k$. Therefore, in the case that u has $\Omega(n)$ incident edges, (4) is essentially saying that no incident edge's leverage score exceeds the average by too much.

To get intuition for condition (5), we need the following two results, whose proofs appear in the full version.

Theorem III.11. *Let H be a $k \times k$-matrix weighted graph that is γ-almost regular (in the sense of (4)). Then for any $\zeta \in (0, 1)$, the number of eigenvalues of its normalized Laplacian that are between $(0, \zeta]$ is at most $\frac{\gamma \cdot k^2}{(1-\zeta)^2}$.*

Proposition III.12. *Let ℓ be the number of λ_i's that are between $(0, \zeta]$. Then*

$$\sum_{(u,v) \in E} \left(D^{\dagger/2} b_{uv}\right)^T \left(\sum_{i:\lambda_i \in (0,\zeta]} \frac{1}{\lambda_i} f_i f_i^T\right) D^{\dagger/2} b_{uv} = \ell. \tag{6}$$

Therefore, in the case that $|E| = \Omega(n^2)$, (5) is essentially saying that the LHS for every edge (u, v) does not exceed the average by too much.

We then show that every dense enough matrix-weighted graph can indeed be made into an expander by downscaling a small number of edges. To this end, let us define, for a *scaling* $s : E \to [0, 1]$, the rescaled graph H^s, which is obtained from H by rescaling each edge (u, v)'s weight to $s_{uv}^2 \phi_{uv}\phi_{uv}^T$. The proof of the following theorem appears in the full version.

Theorem III.13. *There is an algorithm that, given any $k \times k$ matrix-weighted graph $H = (V, E)$ with $|E| \geq \Omega(n^2)$, outputs a scaling $s : E \to [0, 1]$ such that*

1) *The rescaled graph H^s is a (γ, ζ, ψ)-almost regular expander for*
$$\gamma = 8 \log n, \ \zeta = \frac{1}{\log n}, \ \psi = 16k^2 \log^3 n.$$
2) *The number of edges $(u, v) \in E$ with $s_{uv} < 1$ is $o(n^2)$.*

We next show that almost-regular expanders are preserved under vertex sampling. However, we will now use a different notion of "preservation". To state our specific result, let us

define some additional notations. For a vertex subset $C \subseteq V$, we write $L_{G[C]}$ to denote the Laplacian of the vertex-induced subgraph $G[C]$. We also let D_{CC} be the submatrix of D (the degree matrix of the original graph H) with rows and columns restricted to vertices in C, and let $(f_i)_C$ denote, for an eigenvector f_i, the vector f_i with indices restricted to C. We then have the following theorem, whose proof appears in the full version.

Theorem III.14. *There exists a $\theta = \theta(n) \leq n^{o(1)}$ with the following property. Let $H = (V, E)$ be a $k \times k$ matrix-weighted, (γ, ζ, ψ)-almost regular expander where $\zeta \leq 1/\log n$. For an $s \geq 2 \cdot 10^6 \gamma\psi\zeta^{-1}k^2\theta(n)$, let $C \subseteq V$ be a uniformly random vertex subset of size s. Then with probability at least $1 - 1/n^5$, we have that*

1) *The null space of $D_{CC}^{\dagger/2} L_{G[C]} D_{CC}^{\dagger/2}$ is exactly the linear span of $\{(f_i)_C : \lambda_i = 0\}$.*
2) *For all vectors $x \in \mathbb{R}^{|C|k}$ such that $x^T (f_i)_C = 0, \forall i : \lambda_i = 0$,*

$$x^T \left(D_{CC}^{\dagger/2} L_{G[C]} D_{CC}^{\dagger/2} \right)^{\dagger} x \leq n^{o(1)} \cdot x^T \left(\frac{n^2}{s^2} \sum_{i:\lambda_i \in (0,\zeta]} \frac{1}{\lambda_i} (f_i)_C (f_i)_C^T + \frac{n}{s} \cdot \frac{1}{\zeta} I \right) x. \tag{7}$$

We argue that (7) is roughly saying that the pseudoinverse of the subgraph $G[C]$ can be bounded by the pseudoinverse of the original graph that is (i) restricted to indices in C and (ii) rescaled in a certain way. For technical reasons, on the LHS of (7) we normalize the Laplacian of the vertex-induced subgraph using the degree matrix of the *original graph* H. As for the RHS, we can see it as a rescaled version of the pseudoinverse of N restricted to C, by noting that

$$\left(N^{\dagger}\right)_{CC} = \sum_{\lambda_i > 0} \frac{1}{\lambda_i} (f_i)_C (f_i)_C^T.$$

Thus, on the RHS of (7) we blow up the small eigenvalues quadratically in $1/(\text{sampling rate})$, but blow up the large eigenvalues linearly in $1/(\text{sampling rate})$.

With these tools, we are finally able to prove Theorem I.5.

1) Techniques for proving Theorems III.11, III.13, III.14: We now explain, at a very high level, the techniques that we use to prove these three key theorems, as well as their connections to previous works. More details can be found in the subsequent sections.

a) Proof of Theorem III.11.: We consider the "spectral embedding" induced by the bottom eigenvectors of the normalized Laplacian, which maps each vertex to a rectangular matrix. Such a spectral embedding may be seen as the matrix-weighted counterpart of the ones for scalar-weighted graphs, which map each vertex to a *vector*. The latter embeddings were previously used to prove higher-order Cheeger inequalities [24], [29]. We show that, for matrix-weighted graphs that are almost regular (in the sense of (4)), the spectral embedding has vertex-wise bounded spectral norm, and as a result the number of bottom eigenvectors must be small.

b) Proof of Theorem III.13.: Our proof consists of two steps: (i) decomposing the graph into an almost regular graph, and (ii) decomposing the graph into an almost regular expander. In achieving (ii), we actually invoke (i) repeatedly to maintain the almost regularity of the graph.

As noted above, the almost regularity condition (4) is essentially saying that no incident edge has leverage score too large comparing to the average. A similar task to (i) has in fact been investigated by a previous work [7], where the authors showed that given a set of vectors, one can, by downscaling a small number of them, make every vector have small leverage score comparing to the average. This result is achieved by an algorithm that iteratively downscales vectors with large leverage scores, while analyzing how each vector's leverage score changes in the process. While it is possible to directly invoke the result from [7] to get a large almost regular graph, its guarantee does not suffice for our purpose of smoothly incorporating (i) into (ii). In particular, since we will repeatedly invoke (i) in (ii), we need, in addition to that the number of rescaled edges is small, an extra bound on the number of completely deleted edges (i.e. those rescaled to 0) that is proportional to the rank change of the degree matrix D. As a result, we design a more involved algorithm for obtaining the scaling as well as carry out a more careful analysis of the algorithm.

Achieving (ii) turns out to be much more challenging. Although the LHS of (5) may be seen as a leverage score, there is the intrinsic difficulty that whenever the edge weights change, so do the eigenvalues and eigenvectors of the normalized Laplacian, as well as the degree matrix itself (hence also the normalized incidence vectors). Thus it is not clear how the LHS of (5) will change. As a result, when trying to obtain a desired scaling, we have to use some global measure of progress. This is in contrast to (i), where we can track the leverage score change of each edge *locally*. We resolve this issue by considering, as a potential function, the determinant of the normalized Laplacian restricted to the bottom eigenspace. In other words, our potential function is the product of the eigenvalues of N that are between $(0, \zeta]$[6]. We show that, by a delicate global analysis of such a potential function, we are able to make the graph an expander by only downscaling a small number of edges.

c) Proof of Theorem III.14.: Our proof is motivated by the *approximate Gaussian elimination* of the Laplacian matrices of scalar-weighted graphs, which was previously used as an algorithmic tool for solving graph structured linear systems [22], [23] and building data structures for dynamically maintaining effective resistances [9], [25], [27]. Our approach also relies on analyzing matrix-valued martingales which played a key role in [23]. which have played key roles in constructing vertex/subspace sparsifiers [11], [23], [26].

Let us first briefly review the Gaussian elimination of the Laplacian matrix of a scalar-weighted graph. Roughly

[6]Due to technical reasons, the actual potential function slightly differs from the one stated here.

speaking, by eliminating the row and column of L corresponding to a vertex u, we can obtain another Laplacian matrix L' supported on $V \setminus \{u\}$ whose pseudoinverse *equals* the pseudoinverse of the original L restricted to $V \setminus \{u\}$ (i.e. $(L')^\dagger = (L^\dagger)_{V\setminus\{u\}V\setminus\{u\}}$). Given a vertex subset $C \subseteq V$, one can also eliminate the vertices outside of C one by one and get a Laplacian matrix L'' supported on C with the same property that $(L'')^\dagger = (L^\dagger)_{CC}$. The matrix L'' is referred to as the *Schur complement* of L onto C. However, the graphs associated with L' and L'' could be dense, which are inefficient for algorithm design. Therefore [23] showed that one can perform an *approximate* Gaussian elimination by, upon each elimination, implicitly sub-sampling the edges in L'. They then showed that we eventually get a good approximation to L'' by analyzing a matrix-valued martingale induced by this process.

We now explain how to apply this idea to prove Theorem III.14. Since we are considering an induced subgraph $G[C]$ where C is a uniformly random subset of size s, we can also view the process for choosing C as deleting a sequence of $n - s$ vertices from V uniformly at random. Our goal is to compare the pseudoinverse of $G[C]$ with that of the original graph, therefore it suffices to compare it with the Schur complement of L onto C. We will in fact do such a comparison upon the elimination of every vertex. That is, if we let C_i be the set of remaining vertices at the i^{th} step, then we want to compare the Laplacian of $G[C_i]$ with the Schur complement of L onto C_i. At a high level, we do so by setting up a matrix-valued martingale, and show that it has good concentration when G is a matrix-weighted expander (in the sense of Definition III.6).

REFERENCES

[1] Arpit Agarwal, Sanjeev Khanna, Huan Li, and Prathamesh Patil. Sublinear algorithms for hierarchical clustering. 2022. To appear in Thirty-sixth Conference on Neural Information Processing Systems (NeurIPS 2022). Available at https://arxiv.org/abs/2206.07633.

[2] Kook Jin Ahn, Sudipto Guha, and Andrew McGregor. Graph sketches: sparsification, spanners, and subgraphs. In *Proceedings of the 31st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems, PODS 2012, Scottsdale, AZ, USA, May 20-24, 2012*, pages 5–14, 2012.

[3] Noga Alon and Vitali D Milman. $\lambda 1$, isoperimetric inequalities for graphs, and superconcentrators. *Journal of Combinatorial Theory, Series B*, 38(1):73–88, 1985.

[4] András A. Benczúr and David R. Karger. Randomized approximation schemes for cuts and flows in capacitated graphs. *SIAM J. Comput.*, 44(2):290–319, 2015.

[5] Jiecao Chen, He Sun, David P. Woodruff, and Qin Zhang. Communication-optimal distributed clustering. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3720–3728, 2016.

[6] Michael B. Cohen, Rasmus Kyng, Gary L. Miller, Jakub W. Pachocki, Richard Peng, Anup B. Rao, and Shen Chen Xu. Solving SDD linear systems in nearly $m\log^{1/2}n$ time. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 343–352, 2014.

[7] Michael B. Cohen, Yin Tat Lee, Cameron Musco, Christopher Musco, Richard Peng, and Aaron Sidford. Uniform sampling for matrix approximation. In *Proceedings of the 2015 Conference on Innovations in Theoretical Computer Science, ITCS 2015, Rehovot, Israel, January 11-13, 2015*, pages 181–190, 2015.

[8] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional gaussians. *arXiv preprint arXiv:1810.08693*, 2018.

[9] David Durfee, Rasmus Kyng, John Peebles, Anup B. Rao, and Sushant Sachdeva. Sampling random spanning trees faster than matrix multiplication. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 730–742. ACM, 2017.

[10] Arnold Filtser, Michael Kapralov, and Navid Nouri. Graph spanners by sketching in dynamic streams and the simultaneous communication model. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 1894–1913, 2021.

[11] Sebastian Forster, Gramoz Goranci, Yang P. Liu, Richard Peng, Xiaorui Sun, and Mingquan Ye. Minor sparsifiers and the distributed laplacian paradigm. In *62nd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2021, Denver, CO, USA, February 7-10, 2022*, pages 989–999. IEEE, 2021.

[12] Oded Goldreich and Dana Ron. A sublinear bipartiteness tester for bounded degree graphs. *Comb.*, 19(3):335–373, 1999.

[13] Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006.

[14] Arun Jambulapati and Aaron Sidford. Ultrasparse ultrasparsifiers and faster laplacian system solvers. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021, Virtual Conference, January 10 - 13, 2021*, pages 540–559, 2021.

[15] Ravi Kannan, Santosh S. Vempala, and Adrian Vetta. On clusterings: Good, bad and spectral. *J. ACM*, 51(3):497–515, 2004.

[16] Michael Kapralov, Robert Krauthgamer, Jakab Tardos, and Yuichi Yoshida. Towards tight bounds for spectral sparsification of hypergraphs. In Samir Khuller and Virginia Vassilevska Williams, editors, *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pages 598–611. ACM, 2021.

[17] Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single pass spectral sparsification in dynamic streams. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pages 561–570, 2014.

[18] Michael Kapralov, Yin Tat Lee, Cameron Musco, Christopher Musco, and Aaron Sidford. Single pass spectral sparsification in dynamic streams. *SIAM J. Comput.*, 46(1):456–477, 2017.

[19] Michael Kapralov, Aida Mousavifar, Cameron Musco, Christopher Musco, Navid Nouri, Aaron Sidford, and Jakab Tardos. Fast and space efficient spectral sparsification in dynamic streams. In *Proceedings of the 2020 ACM-SIAM Symposium on Discrete Algorithms, SODA 2020, Salt Lake City, UT, USA, January 5-8, 2020*, pages 1814–1833, 2020.

[20] Jonathan A. Kelner, Yin Tat Lee, Lorenzo Orecchia, and Aaron Sidford. An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 217–226, 2014.

[21] Ioannis Koutis. Simple parallel and distributed algorithms for spectral graph sparsification. In Guy E. Blelloch and Peter Sanders, editors, *26th ACM Symposium on Parallelism in Algorithms and Architectures, SPAA '14, Prague, Czech Republic - June 23 - 25, 2014*, pages 61–66. ACM, 2014.

[22] Rasmus Kyng, Yin Tat Lee, Richard Peng, Sushant Sachdeva, and Daniel A Spielman. Sparsified cholesky and multigrid solvers for connection laplacians. In *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, pages 842–850, 2016.

[23] Rasmus Kyng and Sushant Sachdeva. Approximate gaussian elimination for laplacians - fast, sparse, and simple. In Irit Dinur, editor, *IEEE 57th Annual Symposium on Foundations of Computer Science, FOCS 2016, 9-11 October 2016, Hyatt Regency, New Brunswick, New Jersey, USA*, pages 573–582. IEEE Computer Society, 2016.

[24] James R. Lee, Shayan Oveis Gharan, and Luca Trevisan. Multiway spectral partitioning and higher-order cheeger inequalities. *J. ACM*, 61(6):37:1–37:30, 2014.

[25] Huan Li, Stacy Patterson, Yuhao Yi, and Zhongzhi Zhang. Maximizing the number of spanning trees in a connected graph. *IEEE Trans. Inf. Theory*, 66(2):1248–1260, 2020.

[26] Huan Li and Aaron Schild. Spectral subspace sparsification. In Mikkel Thorup, editor, *59th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2018, Paris, France, October 7-9, 2018*, pages 385–396. IEEE Computer Society, 2018.

[27] Huan Li and Zhongzhi Zhang. Kirchhoff index as a measure of edge centrality in weighted networks: Nearly linear time algorithms. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 2377–2396. SIAM, 2018.

[28] Mu Li, Gary L. Miller, and Richard Peng. Iterative row sampling. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 127–136, 2013.

[29] Anand Louis, Prasad Raghavendra, Prasad Tetali, and Santosh S. Vempala. Many sparse cuts via higher eigenvalues. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 1131–1140, 2012.

[30] Noam Nisan. Pseudorandom generators for space-bounded computation. *Combinatorica*, 12(4):449–461, 1992.

[31] Richard Peng. Approximate undirected maximum flows in $O(m\text{polylog}(n))$ time. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 1862–1867, 2016.

[32] Jonah Sherman. Nearly maximum flows in nearly linear time. In *54th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2013, 26-29 October, 2013, Berkeley, CA, USA*, pages 263–269, 2013.

[33] Daniel A. Spielman and Shang-Hua Teng. Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems. In *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, Chicago, IL, USA, June 13-16, 2004*, pages 81–90, 2004.

[34] Daniel A. Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM J. Matrix Anal. Appl.*, 35(3):835–885, 2014.

[35] Andrew Chi-Chih Yao. Probabilistic computations: Toward a unified measure of complexity (extended abstract). In *18th Annual Symposium on Foundations of Computer Science, Providence, Rhode Island, USA, 31 October - 1 November 1977*, pages 222–227. IEEE Computer Society, 1977.