Spectral Subspace Sparsification

Huan Li
School of Computer Science
Fudan University
Shanghai, China
huanli16@fudan.edu.cn

Aaron Schild
Electrical Engineering and Computer Science
University of California, Berkeley
Berkeley, USA
aschild@berkeley.edu

***Abstract*—We introduce a new approach to spectral sparsification that approximates the quadratic form of the pseudoinverse of a graph Laplacian restricted to a subspace. We show that sparsifiers with a near-linear number of edges in the dimension of the subspace exist. Our setting generalizes that of Schur complement sparsifiers. Our approach produces sparsifiers by sampling a uniformly random spanning tree of the input graph and using that tree to guide an edge elimination procedure that contracts, deletes, and reweights edges. In the context of Schur complement sparsifiers, our approach has two benefits over prior work. First, it produces a sparsifier in almost-linear time with no runtime dependence on the desired error. We directly exploit this to compute approximate effective resistances for a small set of vertex pairs in faster time than prior work (Durfee-Kyng-Peebles-Rao-Sachdeva '17). Secondly, it yields sparsifiers that are reweighted minors of the input graph. As a result, we give a near-optimal answer to a variant of the Steiner point removal problem.**

A key ingredient of our algorithm is a subroutine of independent interest: a near-linear time algorithm that, given a chosen set of vertices, builds a data structure from which we can query a multiplicative approximation to the decrease in the effective resistance between two vertices after identifying all vertices in the chosen set to a single vertex with inverse polynomial additional additive error in near-constant time.

***Keywords*-Schur complements; random spanning trees; graph sparsification**

I. Introduction

Graph sparsification has had a number of applications throughout algorithms and theoretical computer science. In this work, we loosen the requirements of spectral sparsification and show that this loosening enables us to obtain sparsifiers with fewer edges. Specifically, instead of requiring that the Laplacian pseudoinverse quadratic form is approximated for every vector, we just require that the sparsifier approximates the Laplacian pseudoinverse quadratic form on a subspace:

Definition I.1 (Spectral subspace sparsifiers)**.** *Consider a weighted graph G, a vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$ that is orthogonal to $\mathbf{1}^{V(G)}$, and $\epsilon \in (0,1)$. For a minor H of G with contraction map $\phi : V(G) \to V(H)$, let $P \in \mathbb{R}^{V(H)\times V(G)}$ be a matrix with $P_{uv} = \mathbb{1}[u = \phi(v)]$ for all $u \in V(H), v \in V(G)$. A reweighted minor H of G is a called an* $(\mathcal{S},\epsilon)$-spectral subspace sparsifier *if for all vectors $x \in \mathcal{S}$,*

$$(1-\epsilon)x^T L_G^+ x \leq x_H^T L_H^+ x_H \leq (1+\epsilon)x^T L_G^+ x$$

where $x_H := Px$.

[KMST10] also considers a form of specific form of subspace sparsification related to controlling the k smallest eigenvalues of a spectral sparsifier for $\mathcal{S} = \mathbb{R}^{V(G)}$. When $\mathcal{S}$ is the dimension $|S|-1$ subspace of $\mathbb{R}^{|S|} \times \mathbf{0}^{n-|S|}$ that is orthogonal to $\mathbf{1}^{V(G)}$, a (S,ϵ)-spectral subspace sparsifier is a sparsifier for the Schur complement of G restricted to the set of vertices S. Schur complement sparsifiers are implicitly constructed in [KS16] and [KLP$^+$16] by an approximate Gaussian elimination procedure and have been used throughout spectral graph theory. For example, they are used in algorithms for random spanning tree generation [DKP$^+$17], [DPPR17], approximate maximum flow [MP13], and effective resistance computation [GHP18], [GHP17], [DKP$^+$17].

Unlike the existing construction of Schur complement sparsifiers [DKP$^+$17], our algorithm (a) produces a sparsifier with vertices outside of S and (b) produces a sparsifier that is a minor of the input graph. While (a) is a disadvantage to our approach, it is not a problem in applications, in which the number of edges in the sparsifier is the most relevant feature for performance, as illustrated by our almost-optimal algorithm for ϵ-approximate effective resistance computation. (b) is an additional benefit to our construction and connects to the well-studied class of Steiner point removal problems [CGH16], [EGK$^+$14].

In the Approximate Terminal Distance Preservation problem [CGH16], one is given a graph G and a set of k vertices S. One is asked find a reweighted minor H of G with size $\text{poly}(k)$ for which

$$d_G(u,v) \leq d_H(u,v) \leq \alpha d_G(u,v)$$

for all $u, v \in S$ and some small *distortion* $\alpha > 1$. The fact that H is a minor of G is particularly useful in the

Huan Li is with Shanghai Key Laboratory of Intelligent Information Processing, School of Computer Science, Fudan University, Shanghai 200433.

Aaron Schild is supported by NSF grant CCF-1553751

context of planar graphs. One can equivalently phrase this problem as a problem of finding a minor H in which the ℓ_1-norm of the ℓ_1-minimizing flow between any two vertices $s, t \in S$ is within an α-factor of the ℓ_1 norm of the ℓ_1-minimizing $s - t$ flow in G. The analogous problem for ℓ_∞ norms is the problem of constructing a *flow sparsifier* (with non-$s - t$ demands as well). Despite much work on flow sparsifiers [Moi09], [LM10], [CLLM10], [MM10], [EGK$^+$14], [Chu12], [AGK14], [RST14], it is still not known whether $\alpha = (1 + \epsilon)$-flow sparsifiers with size $\text{poly}(k, 1/\epsilon)$ exist, even when the sparsifier is not a minor of the original graph.

A. Our Results

Our main result is the following:

Theorem I.2. *Consider a weighted graph G, a d-dimensional vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$, and $\epsilon \in (0, 1)$. Then a $(\mathcal{S}, \epsilon)$-spectral subspace sparsifier for G with $O\left(\frac{d \log d}{\epsilon^2}\right)$ edges exists.*

When $\mathcal{S}$ is the maximal subspace of $\mathbb{R}^S \times \mathbf{0}^{V(G)\setminus S}$ orthogonal to $\mathbf{1}^{V(G)}$ for some set of vertices $S \subseteq V(G)$, $(\mathcal{S}, \epsilon)$-spectral subspace sparsifiers satisfy the same approximation guarantee as Schur complement sparsifiers. The approximation guarantee of a spectral subspace sparsifier H of G is equivalent to saying that for any demand vector $d \in \mathcal{S}$, the energy of the ℓ_2-minimizing flow for d in H is within a $(1 + \epsilon)$ factor of the energy for the ℓ_2-minimizing flow for d in G. This yields an near-optimal (up to a $\log d$ factor) answer to the $(1+\epsilon)$-approximate Steiner vertex removal problem for the ℓ_2 norm. The ℓ_2 version is substantially different from the ℓ_1 problem, in which there do not exist $o(k^2)$-size minors that 2-approximate all terminal distances [CGH16].

Unlike Schur complement sparsifiers, $(\mathbb{R}^S, \epsilon)$-spectral subspace sparsifiers may contain "Steiner nodes;" i.e. vertices outside of S. This is generally not relevant in applications, as illustrated by Corollary I.5. Allowing Steiner nodes allows us to obtain sparsifiers with fewer edges, which in turn allows us to obtain faster constructions. Specifically, we show the following result:

Theorem I.3. *Consider a weighted graph G, a set of vertices $S \subseteq V(G)$, and $\epsilon \in (0, 1)$. Let $\mathcal{T}_{rst}(G)$ denote the time it takes to generate a random spanning tree from a distribution with total variation distance at most $1/m^{10}$ from the uniform distribution. Then a $(\mathbb{R}^S \times \mathbf{0}^{V(G)\setminus S}, \epsilon)$-spectral subspace sparsifier for G with $\min(m, O\left(|S|\frac{\text{polylog}(n)}{\epsilon^2}\right))$ edges can be constructed in $\mathcal{T}_{rst}(G) + O(m\text{polylog}(n)) \leq m^{1+o(1)}$ time.*

This sparsifier has as many edges as the Schur complement sparsifier given in [DKP$^+$17]. Our runtime of $m^{1+o(1)}$ improves on their $\tilde{O}(m + n/\epsilon^2)$ runtime for subconstant values of ϵ. An important ingredient in our construction is a subroutine for multiplicatively approximating changes in effective resistances due to certain modifications of G. In this work, we call the following subroutine with $\delta_0 = \Theta(1)$ and $\delta_1 = 1/\text{poly}(n)$:

Lemma I.4. *Consider a weighted graph G, a set of vertices $S \subseteq V(G)$, and $\delta_0, \delta_1 \in (0, 1)$. There is an $O(m\text{polylog}(n)\log(m/\delta_1)/\delta_0^2)$-time algorithm $\texttt{DiffApx}(G, S, \delta_0, \delta_1)$ that outputs numbers ν_e for all $e \in E(G)$ with the guarantee that*

$$(1 - \delta_0)\nu_e - \delta_1 \leq \frac{b_e^T L_G^+ b_e}{r_e} - \frac{b_e^T L_{G/S}^+ b_e}{r_e} \leq (1 + \delta_0)\nu_e + \delta_1$$

We give a proof of Lemma I.4 in the full version. Finally, we replace the use of Theorem 6.1 in [DKP$^+$17] with our Theorem I.3 in their improvement to Johnson-Lindenstrauss to obtain a faster algorithm:

Corollary I.5. *Consider a weighted graph G, a set of pairs of vertices $P \subseteq V(G) \times V(G)$, and an $\epsilon \in (0, 1)$. There is an $m^{1+o(1)} + \tilde{O}(|P|/\epsilon^2)$-time algorithm $\texttt{ResApx}(G, P, \epsilon)$ that outputs $(1 + \epsilon)$-multiplicative approximations to the quantities*

$$b_{uv}^T L_G^+ b_{uv}$$

for all pairs $(u, v) \in P$.

This directly improves upon the algorithm in [DKP$^+$17], which takes $O((m + (n + |P|)/\epsilon^2)\text{polylog}(n))$-time. This result uses the same reduction as one given in [DKP$^+$17] from effective resistance computation to approximate Schur complements, so we defer the proof of Corollary I.5 to the full version.

B. Technical Overview

To construct Schur complement sparsifiers, [DKP$^+$17] eliminates vertices one-by-one and sparsifies the cliques resulting from those eliminations. This approach is fundamentally limited in that each clique sparsification takes $\Omega(1/\epsilon^2)$ time in general. Furthermore, in the $n + 1$ vertex star graph with n vertices $v_1, v_2, \ldots, v_n$ connected to a single vertex v_{n+1}, a $(1+\epsilon)$-approximate Schur complement sparsifier without Steiner vertices for the set $\{v_1, v_2, \ldots, v_n\}$ must contain $\Omega(n/\epsilon^2)$ edges. As a result, it seems difficult to obtain Schur complement sparsifiers in time less than $\tilde{O}(m + n/\epsilon^2)$ time using vertex elimination.

Instead, we eliminate edges from a graph by contracting or deleting them. Edge elimination has the attractive feature that, unlike vertex elimination, it always reduces the number of edges. Start by letting $H := G$. To eliminate an edge e from the current graph H, sample $X_e \sim \text{Ber}(p_e)$ for some probability p_e depending on e, contract e if $X_e = 1$, and delete e if $X_e = 0$.

To analyze the sparsifier produced by this procedure, we set up a matrix-valued martingale and reduce the problem to bounding the maximum and minimum eigenvalues of a random matrix with expectation equal to the identity matrix. The right value for p_e for preserving this matrix in expectation turns out to be the probability that a uniformly random spanning tree of H contains the edge e. To bound the variance of the martingale, one can use the Sherman-Morrison rank one update formula to bound the change in L_H^+ due to contracting or deleting the edge e. When doing this, one sees that the maximum change in eigenvalue is at most a constant times

$$\max_{x \in \mathcal{S}} \frac{(x^T L_H^+ b_e)^2}{r_e \min(\mathtt{lev}_H(e), 1 - \mathtt{lev}_H(e))(x^T L_G^+ x)}$$

where $\mathtt{lev}_H(e)$ is the probability that e is in a uniformly random spanning tree of H. This quantity is naturally viewed as the quotient of two quantities:

(a) The maximum fractional energy contribution of e to any demand vector in $\mathcal{S}$'s electrical flow.
(b) The minimum of the probabilities that e is in or is not in a uniformly random spanning tree of H.

We now make the edge elimination algorithm more specific to bound these two quantities. Quantity (a) is small on average over all edges in e (see Proposition III.9), so choosing the lowest-energy edge yields a good bound on the maximum change. To get a good enough bound on the stepwise martingale variance, it suffices to sample an edge uniformly at random from the half of edges with lowest energy. Quantity (b) is often not bounded away from 0, but can be made so by modifying the sampling procedure. Instead of contracting or deleting the edge e, start by *splitting* it into two parallel edges with double the resistance or two series edges with half the resistance, depending on whether or not $\mathtt{lev}_H(e) \leq 1/2$. Then, pick one of the halves e_0, contract it with probability p_{e_0}, or delete it otherwise. This produces a graph in which the edge e is either contracted, deleted, or reweighted. This procedure suffices for proving our main existence result (Theorem I.2). This technique is similar to the technique used to prove Lemma 1.4 of [Sch17].

While the above algorithm does take polynomial time, it does not take almost-linear time. We can accelerate it by batching edge eliminations together using what we call *steady oracles*. The contraction/deletion/reweight decisions for edges in H during each batch can be made by sampling just one $1/m^{10}$-approximate uniformly random spanning tree, which takes $m^{1+o(1)}$ time. The main remaining difficulty is finding a large set of edges for which quantity (a) does not change much over the course of many edge contractions/deletions. To show the existence of such a set, we exploit electrical flow localization [SRS17]. To find this set, we use matrix sketching and a new primitive for approximating the change in leverage score due to the identification of some set of vertices S (Lemma I.4), which may be of independent interest. The primitive for approximating the change works by writing the change in an Euclidean norm, reducing the dimension by Johnson-Lindenstrauss Lemma, and then computing the embedding by Fast Laplacian Solvers in near-linear time.

We conclude by briefly discussing why localization is relevant for showing that quantity (a) does not change over the course of many iterations. The square root of the energy contribution of an edge e to x's electrical flow after deleting an edge f is

$$\begin{aligned}
\left|\frac{x^T L_{H\setminus f}^+ b_e}{\sqrt{r_e}}\right| &= \left|\frac{x^T L_H^+ b_e}{\sqrt{r_e}} + \frac{(x^T L_H^+ b_f)(b_f^T L_H^+ b_e)}{(r_f - b_f^T L_H^+ b_f)\sqrt{r_e}}\right| \\
&= \left|\frac{x^T L_H^+ b_e}{\sqrt{r_e}} + \frac{1}{1 - \mathtt{lev}_H(f)} \frac{x^T L_H^+ b_f}{\sqrt{r_f}} \frac{b_f^T L_H^+ b_e}{\sqrt{r_f}\sqrt{r_e}}\right| \\
&\leq \left|\frac{x^T L_H^+ b_e}{\sqrt{r_e}}\right| + \frac{1}{1 - \mathtt{lev}_H(f)} \left|\frac{x^T L_H^+ b_f}{\sqrt{r_f}}\right| \left|\frac{b_f^T L_H^+ b_e}{\sqrt{r_f}\sqrt{r_e}}\right|
\end{aligned}$$

by Sherman-Morrison. In particular, the new energy on e is at most the old energy plus some multiple of the energy on the deleted edge f. By [SRS17], the average value of this multiplier over all edges e and f is $\tilde{O}(\frac{1}{|E(H)|})$, which means that the algorithm can do $\tilde{\Theta}(|E(H)|)$ edge deletions/contractions without seeing the maximum energy on edges e change by more than a factor of 2.

II. Preliminaries

A. Graphs and Laplacians

For a graph G and a subset of vertices S, let G/S denote the graph obtained by *identifying* S to a single vertex s. Specifically, for any edge $e = \{u, v\}$ in G, replace each endpoint $u, v \in S$ with s and do not change any endpoint not in S. Then, remove all self-loops to obtain G/S.

Let $G = (V(G), E(G))$ be a weighted undirected graph with n vertices, m edges, and edge weights $\{w_e\}_{e \in E(G)}$. The Laplacian of G is an $n \times n$ matrix given by:

$$(L_G)_{u,v} := \begin{cases} -w_{(u,v)} & \text{if } u \neq v \text{ and } (u,v) \in E(G), \\ \sum_{(u,w) \in E(G)} w_{(u,w)} & \text{if } u = v, \\ 0 & \text{otherwise.} \end{cases}$$

We define edge resistances $\{r_e\}_{e \in E(G)}$ by $r_e = 1/w_e$ for all $e \in E(G)$.

If we orient every edge $e \in E(G)$ arbitrarily, we can define the singed edge-vertex incidence matrix B_G by

$$(B_G)_{e,u} := \begin{cases} 1 & \text{if } u \text{ is } e\text{'s head}, \\ -1 & \text{if } u \text{ is } e\text{'s tail}, \\ 0 & \text{otherwise.} \end{cases}$$

Then we can write L_G as $L_G = B_G^T W_G B_G$, where W_G is a diagonal matrix with $(W_G)_{e,e} = w_e$.

For vertex sets $S, T \subseteq V$, $(L_G)_{S,T}$ denotes the submatrix of L_G with row indices in S and column indices in T.

L_G is always positive semidefinite, and only has one zero eigenvalue if G is connected. For a connected graph G, let $0 = \lambda_1(L_G) < \lambda_2(L_G) \leq \ldots \leq \lambda_n(L_G)$ be the eigenvalues of L_G. Let $u_1, u_2, \ldots, u_n$ be the corresponding set of orthonormal eigenvectors. Then, we can diagonalize L_G and write

$$L_G = \sum_{i=2}^{n} \lambda_i(L_G) u_i u_i^T.$$

The pseudoinverse of L_G is then given by

$$L_G^+ = \sum_{i=2}^{n} \frac{1}{\lambda_i(L_G)} u_i u_i^T.$$

In the rest of the paper, we will write $\lambda_{\min}(\cdot)$ to denote the smallest eigenvalue and $\lambda_{\max}(\cdot)$ to denote the largest eigenvalue. We will also write $\sigma_{\max}(\cdot)$ to denote the largest singular value, which is given by

$$\sigma_{\max}(A) = \sqrt{\lambda_{\max}(A^T A)}$$

for any matrix A.

B. Leverage Scores and Rank One Updates

For a graph G and an edge $e \in E(G)$, let $b_e \in \mathbb{R}^{V(G)}$ denote the signed indicator vector of the edge e; that is the vector with -1 on one endpoint, 1 on the other, and 0 everywhere else. Define the *leverage score* of e to be the quantity

$$\texttt{lev}_G(e) := \frac{b_e^T L_G^+ b_e}{r_e}$$

Let $d_1, d_2 \in \mathbb{R}^n$ be two vectors with $d_1, d_2 \perp \mathbf{1}^{V(G)}$. Then the following results hold by the Sherman-Morrison rank 1 update formula:

Proposition II.1. *For a graph G and an edge f, let $G \backslash f$ denote the graph with f deleted. Then*

$$d_1^T L_{G\backslash f}^+ d_2 = d_1^T L_G^+ d_2 + \frac{(d_1^T L_G^+ b_f)(b_f^T L_G^+ d_2)}{r_f - b_f^T L_G^+ b_f}$$

Proposition II.2. *For a graph G and an edge f, let G/f denote the graph with f contracted. Then*

$$d_1^T L_{G/f}^+ d_2 = d_1^T L_G^+ d_2 - \frac{(d_1^T L_G^+ b_f)(b_f^T L_G^+ d_2)}{b_f^T L_G^+ b_f}$$

C. Random Spanning Trees

We use the following result on uniform random spanning tree generation:

Theorem II.3 (Theorem 1.2 of [Sch17])**.** *Given a weighted graph G with m edges, a random spanning tree T of G can be sampled from a distribution with total variation distance at most $1/m^{10}$ from the uniform distribution in time $m^{1+o(1)}$.*

Let $T \sim G$ denote the uniform distribution over spanning trees of G. We also use the following classic result:

Theorem II.4 ([Kir47])**.** *For any edge $e \in E(G)$, $\Pr_{T\sim G}[e \in T] = \texttt{lev}_G(e)$.*

For an edge $e \in E(G)$, let $G[e]$ denote a random graph obtained by contracting e with probability $\texttt{lev}_G(e)$ and deleting e otherwise.

D. Some Useful Bounds and Tools

We now describe some useful bounds/tools we will need in our algorithms. In all the following bounds, we define the quantities $w_{\max}$ and $w_{\min}$ as follows:

$$w_{\max} := \max\left\{1, \max_{e\in E(G)} 1/r_e\right\},$$
$$w_{\min} := \min\left\{1, \min_{e\in E(G)} 1/r_e\right\}.$$

The following lemma bounds the range of eigenvalues for Laplacians and SDDM matrices:

Lemma II.5. *For any Laplacian L_G and $S \subset V(G)$,*

$$\lambda_2(L_G) \geq w_{\min}/n^2, \tag{1}$$
$$\lambda_{\min}\left((L_G)_{S,S}\right) \geq w_{\min}/n^2, \tag{2}$$
$$\lambda_{\max}\left((L_G)_{S,S}\right) \leq \lambda_{\max}(L_G) \leq n w_{\max}. \tag{3}$$

Proof: Defered to the full version. ∎

The lemma below gives upper bounds on the largest eigenvalues/singular values for some useful matrices:

Lemma II.6. *The following upper bounds on the largest singular values/eigenvalues hold:*

$$\sigma_{\max}(W_G^{1/2} B_G) \leq (n w_{\max})^{1/2}, \tag{4}$$
$$\lambda_{\max}(SC(L_G, S)) \leq n w_{\max}, \tag{5}$$
$$\sigma_{\max}((L_G)_{S,T}) = \sigma_{\max}((L_G)_{T,S}) \leq n w_{\max}, \tag{6}$$

where $T := V(G) \setminus S$.

Proof: Defered to the full version. ∎

We will need to invoke Fast Laplacian Solvers to apply the inverse of a Laplacian of an SDDM matrix. The following lemma characterizes the performance of Fast Laplacian Solvers:

Lemma II.7 (Fast Laplacian Solver [ST14], [CKM$^+$14])**.** *There is an algorithm $\tilde{x} = \texttt{LaplSolve}(M, b, \epsilon)$ which takes a matrix $M_{n\times n}$ either a Laplacian or an SDDM matrix*

with m nonzero entries, a vector $b \in \mathbb{R}^n$, and an error parameter $\epsilon > 0$, and returns a vector $\tilde{x} \in \mathbb{R}^n$ such that

$$\|x - \tilde{x}\|_M \leq \epsilon \|x\|_M$$

holds with high probability, where $\|x\|_M := \sqrt{x^T M x}$, $x := M^{-1}b$, and M^{-1} denotes the pseudoinverse of M when M is a Laplacian. The algorithm runs in time $O(m\text{polylog}(n)\log(1/\epsilon))$.

The following lemmas show how to bound the errors of Fast Laplacian Solvers in terms of ℓ_2 norms, which follows directly from the bounds on Laplacian eigenvalues in Lemma II.5:

Lemma II.8. *For any Laplacian L_G, vectors $x, \tilde{x} \in \mathbb{R}^n$ both orthogal to 1, and real number $\epsilon > 0$ satifiying*

$$\|x - \tilde{x}\|_{L_G} \leq \epsilon \|x\|_{L_G},$$

the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \|x\|.$$

Proof: Defered to the full version. ∎

Lemma II.9. *For any Laplacian L_G, $S \subset V$, vectors $x, \tilde{x} \in \mathbb{R}^{|S|}$, and real number $\epsilon > 0$ satifiying*

$$\|x - \tilde{x}\|_M \leq \epsilon \|x\|_M,$$

where $M := (L_G)_{S,S}$, the following statement holds:

$$\|x - \tilde{x}\| \leq \epsilon n^{1.5} \left(\frac{w_{\max}}{w_{\min}}\right)^{1/2} \|x\|.$$

Proof: Defered to the full version. ∎

When computing the changes in effective resistances due to the identification of a given vertex set (i.e. merging vertices in that set and deleting any self loops formed), we will need to use Johnson-Lindenstrauss lemma to reduce dimensions:

Lemma II.10 (Johnson-Lindenstrauss Lemma [JL84], [Ach01])**.** *Let $v_1, v_2, \ldots, v_n \in \mathbb{R}^d$ be fixed vectors and $0 < \epsilon < 1$ be a real number. Let k be a positive integer such that $k \geq 24 \log n/\epsilon^2$ and $Q_{k\times d}$ be a random ± 1 matrix. With high probability, the following statement holds for any $1 \leq i, j \leq n$:*

$$(1-\epsilon)\|v_i - v_j\|^2 \leq \|Qv_i - Qv_j\|^2 \leq (1+\epsilon)\|v_i - v_j\|^2.$$

We will also need to use Schur complements to explicitly write the changes in effective resistances when identifying a vertex set. We give the following definition of Schur complements:

Definition II.11 (Schur Complements)**.** *The Schur complement of a graph G onto a subset of vertices $S \subset V(G)$, denoted by $SC(G, S)$ or $SC(L_G, S)$, is defined as*

$$SC(L_G, S) = (L_G)_{S,S} - (L_G)_{S,T}(L_G)^{-1}_{T,T}(L_G)_{T,S},$$

where $T := V(G) \setminus S$.

The fact below relates Schur complements to the inverse of graph Laplacian:

Fact II.12 (see, e.g., Fact 5.4 in [DKP+17])**.** *For any graph G and $S \subset V(G)$,*

$$\left(I - \frac{1}{|S|}J\right)\left(L_G^+\right)_{S,S}\left(I - \frac{1}{|S|}J\right) = (SC(L_G, S))^+,$$

where I denotes the identity matrix, and J denotes the matrix whose entries are all 1.

III. Existence of sparsifiers

In this section, we reduce the construction of spectral subspace sparsifiers to an oracle that outputs edges that have low energy with respect to every demand vector in the chosen subspace $\mathcal{S}$. We prove it by splitting and conditioning on edges being present in a uniformly random spanning tree one-by-one until $\tilde{O}(d/\epsilon^2)$ edges are left. This construction is a high-dimensional generalization of the construction given in Section 10.1 of [Sch17]. We use the following matrix concentration inequality:

Theorem III.1 (Matrix Freedman Inequality applied to symmetric matrices [Tro11])**.** *Consider a matrix martingale $(Y_k)_{k\geq 0}$ whose values are symmetric matrices with dimension s, and let $(X_k)_{k\geq 1}$ be the difference sequence $X_k := Y_k - Y_{k-1}$. Assume that the difference sequence is uniformly bounded in the sense that*

$$\lambda_{\max}(X_k) \leq R$$

almost surely for $k \geq 1$. Define the predictable quadratic variation process of the martingale:

$$W_k := \sum_{j=1}^{k} \boldsymbol{E}[X_j^2 | Y_{j-1}]$$

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

$$\Pr[\exists k \geq 0 : \lambda_{\max}(Y_k - Y_0) \geq t \text{ and } \lambda_{\max}(W_k) \leq \sigma^2] \leq s \exp\left(\frac{-t^2/2}{\sigma^2 + Rt/3}\right)$$

Now, we give an algorithm `SubspaceSparsifier`$(G, \mathcal{S}, \epsilon)$ that proves Theorem I.2. The algorithm simply splits and conditions on the edge that minimizes the martingale difference repeatedly until there are too few edges left. For efficiency purposes, `SubspaceSparsifier`$(G, \mathcal{S}, \epsilon)$ receives martingale-difference-minimizing edges from a steady oracle $\mathcal{O}$ with the additional guarantee that differences remain small after many edge updates. This oracle is similar to the stable oracles given in Section 10 of [Sch17].

Definition III.2 (Steady oracles)**.** *A $(\rho, K(z))$-steady oracle is a function $Z \leftarrow \mathcal{O}(I, \mathcal{S})$ that takes in a graph I and a subspace $\mathcal{S} \subseteq \mathbb{R}^{V(I)}$ that satisfy the following condition:*

- *(Leverage scores) For all $e \in E(I)$, $\texttt{lev}_I(e) \in [3/16, 13/16]$.*

and outputs a set $Z \subseteq E(I)$. Let $I_0 = I$ and for each $i > 0$, obtain I_i by picking a uniformly random edge $f_{i-1} \in Z$, arbitrarily letting $I_i \leftarrow I_{i-1} \setminus f_{i-1}$ or $I_i \leftarrow I_{i-1}/f_{i-1}$, and letting $Z \leftarrow Z \setminus \{f_{i-1}\}$. $\mathcal{O}$ satisfies the following guarantees with high probability for all $i < K(|E(I)|)$:

- *(Size of Z) $|Z| \geq |E(I)|/\rho$*
- *(Leverage score stability) $\texttt{lev}_{I_i}(f_i) \in [1/8, 7/8]$*
- *(Martingale change stability) $\max_{x \in \mathcal{S}} \frac{(x_{I_i}^T L_{I_i}^+ b_{f_i})^2}{r_{f_i}(x^T L_I^+ x)} \leq \frac{\rho dim(\mathcal{S})}{|E(I)|}$*

We now state the main result of this section:

Lemma III.3. *Consider a weighted graph G, a d-dimensional vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$, and $\epsilon \in (0, 1)$. There is an algorithm $\texttt{SubspaceSparsifier}(G, \mathcal{S}, \epsilon)$ that, given access to a $(\rho, K(z))$-steady-oracle $\mathcal{O}$, computes a $(\mathcal{S}, \epsilon)$-spectral subspace sparsifier for G with*

$$O\left(\frac{\rho^2 d \log d}{\epsilon^2}\right)$$

edges in time

$$O\left((\log n)(\max_{z \leq |E(G)|} z/K(z))(\mathcal{T}_{rst} + \mathcal{T}_{\mathcal{O}} + m)\right)$$
$$\leq O\left((\log n)(\max_{z \leq |E(G)|} z/K(z))(m^{1+o(1)} + \mathcal{T}_{\mathcal{O}})\right)$$

where $\mathcal{T}_{rst}$ is the time required to generate a spanning tree of G from a distribution with total variation distance $\leq n^{-10}$ from uniform and $\mathcal{T}_{\mathcal{O}}$ is the runtime of the oracle.

The algorithm will use two simple subroutines that modify the graph by splitting edges. `Split` replaces each edge with approximate leverage score less than 1/2 with a two-edge path and each edge with approximate leverage score greater than 1/2 with two parallel edges. `Unsplit` reverses this split for all pairs that remain in the graph. We prove the following two results about this subroutines in the appendix:

Proposition III.4. *There is a linear-time algorithm $(I, \mathcal{P}) \leftarrow \texttt{Split}(H)$ that, given a graph H, produces a graph I with $V(H) \subseteq V(I)$ and a set of pairs of edges $\mathcal{P}$ with the following additional guarantees:*

- *(Electrical equivalence) For all $x \in \mathbb{R}^{V(I)}$ that are supported on $V(H)$, $x^T L_I^+ x = x_H^T L_H^+ x_H$.*
- *(Bounded leverage scores) For all $e \in E(I)$, $\texttt{lev}_I(e) \in [3/16, 13/16]$*
- *($\mathcal{P}$ description) Every edge in I is in exactly one pair in $\mathcal{P}$. Furthermore, there is a bijection between pairs $(e_0, e_1) \in \mathcal{P}$ and edges $e \in E(H)$ for which either (a) e_0, e_1 and e have the same endpoint pair or (b) $e_0 = \{u, w\}$, $e_1 = \{w, v\}$, and $e = \{u, w\}$ for some degree 2 vertex w.*

Proposition III.5. *There is a linear-time algorithm $H \leftarrow \texttt{Unsplit}(I, \mathcal{P})$ that, given a graph I and a set of pairs $\mathcal{P}$ of edges in I, produces a minor H with $V(H) \subseteq V(I)$ and the following additional guarantees:*

- *(Electrical equivalence) For all $x \in \mathbb{R}^{V(I)}$ that are supported on $V(H)$, $x^T L_I^+ x = x_H^T L_H^+ x_H$.*
- *(Edges of H) There is a surjective map $\phi : E(I) \to E(H)$ from non-self-loop,non-leaf edges of I such that for any pair $(e_0, e_1) \in \mathcal{P}$, $\phi(e_0) = \phi(e_1)$. Furthermore, for each $e \in E(H)$, either (a) $\phi^{-1}(e) = e$, (b) $\phi^{-1}(e) = \{e_0, e_1\}$, with $(e_0, e_1) \in \mathcal{P}$ and e_0, e_1 having the same endpoints as e or (c) $\phi^{-1}(e) = \{e_0, e_1\}$, with $(e_0, e_1) \in \mathcal{P}$ and $e_0 = \{u, w\}, e_1 = \{w, v\}$, and $e = \{u, v\}$ for a degree 2 vertex w.*

function SUBSPACESPARSIFIER($G, \mathcal{S}, \epsilon$)
  Input: A weighted graph G, a vector space $\mathcal{S} \subseteq \mathbb{R}^{V(G)}$, $\epsilon \in (0, 1)$, and (implicitly) a $(\rho, K(z))$-steady oracle $\mathcal{O}$
  Output: A $(\mathcal{S}, \epsilon)$-spectral subspace sparsifier for G
  $H \leftarrow G$
  while $|E(H)| \geq 10000\rho^2(\dim(\mathcal{S}) \log(\dim(\mathcal{S})))/\epsilon^2$ **do**
    $(I, \mathcal{P}) \leftarrow \texttt{Split}(H)$
    $Z \leftarrow \mathcal{O}(I, \mathcal{S})$
    $Z' \leftarrow$ a uniformly random subset of Z with size $K(|E(I)|)$
    $T \leftarrow$ a spanning tree of I drawn from a distribution with TV distance $\leq \kappa_0 := 1/n^{10}$ from uniform
    $I' \leftarrow$ the graph with all edges in $E(T) \cap Z'$ contracted and all edges in $Z' \setminus E(T)$ deleted
    $H \leftarrow \texttt{Unsplit}(I', \mathcal{P})$
  end while
  Return H
end function

We analyze the approximation guarantees of H by setting up two families of matrix-valued martingales. In all of the proof besides the final "Proof of Theorem III.3," we sample T from the uniform distribution rather than from a distribution with total variation distance κ_0 from uniform. We bound the error incurred from doing this in the final "Proof of Lemma III.3."

We start by defining the first family, which just consists of one martingale. Let $H_0 := G$ and let H_k be the graph H between iterations k and $k+1$ of the while loop of `SubspaceSparsifier`. Let $d = \dim(\mathcal{S})$. Since $\mathcal{S}$ is orthogonal to $\mathbf{1}^{V(G)}$, $\dim((L_G^+)^{1/2}\mathcal{S}) = \dim(\mathcal{S}) = d$, which means that $\mathcal{S}$ has a basis $\{y_i\}_{i=1}^d$ for which $y_i^T L_G^+ y_j = 0$

for all $i \neq j \in [d]$ and $y_i^T L_G^+ y_i = 1$ for all $i \in [d]$. Let Y_k be the $|V(H_k)| \times d$ matrix with ith column $(y_i)_{H_k}$ and let $Y := Y_0$. Let $M_k := Y_k^T L_{H_k}^+ Y_k - Y^T L_G^+ Y$. Since the y_is form a basis of $\mathcal{S}$, there is a vector a_x for which $x = Y a_x$ for any $x \in \mathcal{S}$. Furthermore, $x_{H_k} = Y_k a_x$ for any $k \geq 0$. In particular,

$$\left| \frac{x_{H_k}^T L_{H_k}^+ x_{H_k}}{x^T L_G^+ x} - 1 \right| = \left| \frac{a_x^T M_k a_x}{||a_x||_2^2} \right|$$

so it suffices to show that $\lambda_{\max}(M_k) \leq \epsilon$ for all $k \leq k_{\text{final}}$, where k_{final} is the number of while loop iterations.

In order to bound the change between M_k and M_{k+1}, we introduce a second family of martingales consisting of one martingale for each while loop iteration. Let $I_{k,0} := I$ during the kth iteration of the while loop in `SubspaceSparsifier`. Generate Z' in Z during iteration k of the while loop by sampling a sequence of edges $f_{k,0}, f_{k,1}, \ldots, f_{k,K(|E(I)|)-1}$ without replacement from Z. Let $I_{k,t} = I_{k,t-1}[f_{k,t-1}]$ for all $t > 0$. For a vector $v \in \mathbb{R}^{V(G)}$, let $v_{I_{k,0}} \in \mathbb{R}^{V(I_{k,0})}$ be the vector with $v_{I_{k,0}}(p) = v_{H_k}(p)$ for $p \in V(H_k)$ and $v_{I_{k,0}}(p) = 0$ for $p \in V(I_{k,0}) \setminus V(H_k)$. For $t > 0$ and $v \in \mathbb{R}^{V(G)}$, let $v_{I_{k,t}} := (v_{I_{k,0}})_{I_{k,t}}$. Let $Y_{k,t}$ be the $|V(I_{k,t})| \times d$ matrix with ith column $(y_i)_{I_{k,t}}$. Let $N_{k,t} := Y_{k,t}^T L_{I_{k,t}}^+ Y_{k,t} - Y^T L_G^+ Y$. For any $x \in \mathcal{S}$, $t \geq 0$, and $k \geq 0$, $x_{I_{k,t}} = Y_{k,t} a_x$. In particular,

$$\left| \frac{x_{I_{k,t}}^T L_{I_{k,t}}^+ x_{I_{k,t}}}{x^T L_G^+ x} - 1 \right| = \left| \frac{a_x^T N_{k,t} a_x}{||a_x||_2^2} \right|$$

Next, we write an equivalent formulation for the steady oracle "Martingale change stability" guarantee that is easier to analyze:

Proposition III.6.

$$\max_{x \in \mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f (x^T L_G^+ x)} = \frac{b_f^T L_{I_{k,t}}^+ Y_{k,t} Y_{k,t}^T L_{I_{k,t}}^+ b_f}{r_f}$$

Proof: Notice that

$$\begin{aligned}
\max_{x \in \mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f (x^T L_G^+ x)} &= \max_{x \in \mathcal{S}} \frac{(a_x^T Y_{k,t}^T L_{I_{k,t}}^+ b_f)(b_f^T L_{I_{k,t}}^+ Y_{k,t} a_x)}{r_f ||a_x||_2^2} \\
&= \max_{a \in \mathbb{R}^d} \frac{a^T Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t} a}{r_f ||a||_2^2} \\
&= \lambda_{\max} \left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{r_f} \right) \\
&= \frac{b_f^T L_{I_{k,t}}^+ Y_{k,t} Y_{k,t}^T L_{I_{k,t}}^+ b_f}{r_f}
\end{aligned}$$

as desired. ∎

Now, we analyze the inner family of matrices $N_{k,t}$. Let $Z_{k,t}$ denote the set Z during iteration k of the while loop after sampling t edges without replacement.

Proposition III.7. *$Y_t := N_{k,t}$ for fixed $k \geq 0$ and varying $t \geq 0$ is a matrix martingale. Furthermore, if*

$$\frac{x_{I_{k,s}}^T L_{I_{k,s}}^+ x_{I_{k,s}}}{x^T L_G^+ x} \leq 10$$

for all $x \in \mathcal{S}$, $k \geq 0$, and $s \leq t$ for some $t \geq 0$, $\lambda_{\max}(X_{t+1}) \leq \frac{90d}{|E(I_{k,t})|}$ and $\lambda_{\max}(\boldsymbol{E}[X_{t+1}^2 | Y_t]) \leq \frac{25600\rho^2 d}{|E(I_{k,0})|^2}$, where X_{t+1} is defined based on the Y_ss as described in Theorem III.1.

Proof: We compute the conditional expectation of $X_{t+1} = Y_{t+1} - Y_t$ given Y_t using Sherman-Morrison:

$$\begin{aligned}
\mathbf{E}[X_{t+1} | Y_t] &= \mathbf{E}[N_{k,t+1} - N_{k,t} | N_{k,t}] \\
&= \frac{1}{|Z_{k,t}|} \sum_{f \in Z_{k,t}} -\frac{b_f^T L_{I_{k,t}}^+ b_f}{r_f} \left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{b_f^T L_{I_{k,t}}^+ b_f} \right) \\
&+ \frac{1}{|Z_{k,t}|} \sum_{f \in Z_{k,t}} \left(1 - \frac{b_f^T L_{I_{k,t}}^+ b_f}{r_f} \right) \left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{r_f - b_f^T L_{I_{k,t}}^+ b_f} \right) \\
&= 0
\end{aligned}$$

Therefore, $(Y_t)_{t \geq 0}$ is a martingale. Since $I_{k,0}$ is the output of `Split`, all edges in $I_{k,0}$ have leverage score between $3/16$ and $13/16$ by Proposition III.4. In particular, the input condition to $\mathcal{O}$ is satisfied. Furthermore,

$$\begin{aligned}
\lambda_{\max}(X_{t+1}) &\leq \lambda_{\max}(N_{k,t+1} - N_{k,t}) \\
&\leq \frac{1}{|Z_{k,t}|} \sum_{f \in Z_{k,t}} \lambda_{\max} \left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{\min(b_f^T L_{I_{k,t}}^+ b_f, r_f - b_f^T L_{I_{k,t}}^+ b_f)} \right) \\
&\leq 8 \max_{f \in Z_{k,t}} \lambda_{\max} \left(\frac{Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}}{r_f} \right) \\
&= 8 \max_{f \in Z_{k,t}} \max_{x \in \mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f (x^T L_G^+ x)} \\
&\leq 80 \max_{f \in Z_{k,t}} \max_{x \in \mathcal{S}} \frac{(x_{I_{k,t}}^T L_{I_{k,t}}^+ b_f)^2}{r_f (x_{I_{k,0}}^T L_{I_{k,0}}^+ x_{I_{k,0}})} \\
&\leq \frac{90 \rho d}{|E(I_{k,0})|}
\end{aligned}$$

where the third inequality follows from "Leverage score stability," the equality follows from Proposition III.6, the fourth inequality follows from the input condition, and the last inequality follows from "Martingale change stability." Also,

$$
\begin{aligned}
&\lambda_{\max}(\mathbf{E}[X_{t+1}^2|Y_t]) \\
&= \lambda_{\max}\left(\mathbf{E}[(N_{k,t+1}-N_{k,t})^2|Y_t]\right) \\
&\le \frac{256}{|Z_{k,t}|}\lambda_{\max}\Big(\sum_{f\in Z_{k,t}} \\
&\frac{1}{r_f^2}Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t} Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}\Big) \\
&\le \frac{256d}{|Z_{k,t}|}\left(\max_{f\in Z_{k,t}}\frac{1}{r_f}b_f^T L_{I_{k,t}}^+ Y_{k,t}Y_{k,t}^T L_{I_{k,t}}^+ b_f\right) \\
&\lambda_{\max}\left(\sum_{f\in Z_{k,t}}\frac{1}{r_f}Y_{k,t}^T L_{I_{k,t}}^+ b_f b_f^T L_{I_{k,t}}^+ Y_{k,t}\right) \\
&\le \frac{2560\rho d}{|Z_{k,t}||E(I_{k,0})|}\lambda_{\max}\left(Y_{k,t}^T L_{I_{k,t}}^+ Y_{k,t}\right) \\
&= \frac{2560\rho d}{|Z_{k,t}||E(I_{k,0})|}\max_{x\in\mathcal{S}}\frac{x_{I_{k,t}}^T L_{I_{k,t}}^+ x_{I_{k,t}}}{x^T L_G^+ x} \\
&\le \frac{25600\rho^2 d}{|E(I_{k,0})|^2}
\end{aligned}
$$

where the second inequality follows from Sherman-Morrison and "Leverage score stability," the fourth follows from "Martingale change stability," and the last follows from "Size of Z" and the input condition. ∎

Now, consider the sequence of matrices $((N_{k,t})_{t=0}^{K(|E(I_{k,t})|)})_{k\ge 0}$ obtained by concatenating the $(N_{k,t})_t$ martingales for each k. We now analyze this sequence:

Proposition III.8. *The sequence of matrices $(Y_{kt})_{k,t}$ ordered lexicographically by (k,t) pairs defined by $Y_{kt} := N_{k,t}$ is a matrix martingale. Furthermore, if for any $k \ge 0, t \ge 0$, any pairs (l,s) lexicographically smaller than (k,t), and any $x \in \mathcal{S}$,*

$$\frac{x_{I_{l,s}}^T L_{I_{l,s}}^+ x_{I_{l,s}}}{x^T L_G^+ x} \le 10$$

then

$$\lambda_{\max}(X_{k't'}) \le \frac{90d}{|E(I_{k',t'})|}$$

$\lambda_{\max}(\boldsymbol{E}[X_{k't'}^2|Y_{kt}]) \le \frac{25600\rho^2 d}{|E(I_{k',t'})|^2}$, *and*

$$\lambda_{\max}(W_{kt}) \le \sum_{(l,s)\le(k,t)}\frac{25600\rho^2 d}{|E(I_{l,s})|^2}$$

where (k',t') is the lexicographic successor to (k,t) and $X_{k't'} = Y_{k't'} - Y_{kt}$ as described in Theorem III.1.

Proof: Consider a pair (k',t') with lexicographic predecessor (k,t). If $k = k'$, then $t' = t+1$, which means that

$$\mathbf{E}[Y_{k't'}|Y_{kt}] = \mathbf{E}[N_{k,t+1}|N_{k,t}] = N_{k,t} = Y_{kt}$$

, $\lambda_{\max}(X_{k't'}) \le \frac{90d}{|E(I_{k',t'})|}$, and $\lambda_{\max}(\mathbf{E}[X_{k't'}^2|Y_{kt}]) \le \frac{25600\rho^2 d}{|E(I_{k',t'})|^2}$ by Proposition III.7. If $k = k'-1$, then $t' = 0$. As a result, $Y_{kt} = N_{k,t} = M_k$ by the "Electrical equivalence" guarantee of Proposition III.5 and $M_k = N_{k',t'} = Y_{k't'}$ by the "Electrical equivalence" guarantee of Proposition III.4. In particular, $X_{k't'} = 0$ and satisfies the desired eigenvalue bounds. The bound on $\lambda_{\max}(W_{kt})$ follows directly from the stepwise bound $\lambda_{\max}(\mathbf{E}[X_{k't'}^2|Y_{kt}])$ and the definition of W_{kt}. ∎

Now, we are ready to prove Lemma III.3.

Proof of Lemma III.3:

H a minor of G. It suffices to show that for every $k \ge 0$, H_{k+1} is a minor of H_k. I' is a minor of I, as I is only modified by deletion and contraction. Now, we show that the unweighted version of H_{k+1} can be obtained from H_k by contracting each edge $e \in E(H_k)$ with an I'-self-loop in its pair $(e_0, e_1) \in P$ and deleting each edge $e \in E(H_k)$ with an I'-leaf edge in its pair. Let H'_{k+1} be the result of this procedure.

We show that $H'_{k+1} = H_{k+1}$ without weights. We start by showing that $V(H_{k+1}) = V(H'_{k+1})$. Each vertex $v \in V(H_{k+1})$ corresponds to a set of vertices in $V(H_k)$ that were identified, as the "Edges of H" requirement ensures that H_{k+1} contains no vertices that were added to H_k by `Split`. Since T is a tree, each vertex $v \in V(H_{k+1})$ corresponds to a subtree of identified vertices in H_k. Since Z only contains one edge for each pair in $\mathcal{P}$, the self-loop edges in I' match the edges contracted to form the subtree for v, which means that $V(H_{k+1}) = V(H'_{k+1})$. $E(H_{k+1}) \subseteq E(H'_{k+1})$ because for every $e \in E(H_{k+1})$, $\phi^{-1}(e)$ does not contain an I' self-loop or leaf by the "Edges of H" and "$\mathcal{P}$ description" guarantees. $E(H'_{k+1}) \subseteq E(H_{k+1})$ because each $e \in E(H'_{k+1})$ does not map to a self-loop or leaf in I', which means that $\phi^{-1}(e)$ exists by surjectivity of ϕ. Therefore, $H'_{k+1} = H_{k+1}$. Since H'_{k+1} is a minor of H_k, H_{k+1} is also a minor of H_k, as desired.

Number of edges. This follows immediately from the while loop termination condition.

Approximation bound. Let (k_τ, t_τ) be the final martingale index pair that the while loop encounters before termination. We start by obtaining a high-probability bound on $W_{k_\tau t_\tau}$ given that T is drawn from the exact uniform distribution on spanning trees of I. By Proposition III.8,

$$W_{k_\tau t_\tau} \le \sum_{(k,t)\le(k_\tau,t_\tau)}\frac{25600\rho^2 d}{|E(I_{k,t})|^2}$$

The process of generating $I_{k,t+1}$ from $I_{k,t}$ does not increase the number of edges and decreases the number of edges by 1 with probability at least 1/8, by "Leverage score stability." Therefore, by Azuma's Inequality, $|E(I_{k,t})| \le 2|E(G)| - c_{k,t}/8 + 10\sqrt{\log n}\sqrt{c_{k,t}}$ with probability at least $1 - 1/n^5$, where $c_{k,t}$ is the number of pairs that are lexicographically less than (k,t). Therefore, as long as $|E(G)| > 20\log n$,

which is true when $n > 10000000 = \Theta(1)$,

$$|E(I_{k,t})| \le 2|E(G)| - c_{k,t}/16$$

with probability at least $1 - 1/n^3$ for all pairs (k,t). This means that

$$c_{k_\tau,t_\tau} \le 32000000|E(G)|$$

and that

$$W_{k_\tau,t_\tau} \le \frac{32000000000\rho^2 d}{|E(I_{k_\tau,t_\tau})|} \le \epsilon^2/(10\log d)$$

with probability at least $1 - 1/d^3$.

Now, we apply the Matrix Freedman Inequality (Theorem III.1). Apply it to the martingale $(Y_{kt})_{k,t}$ to bound $\lambda_{\max}(Y_{k_\tau t_\tau} - Y_{00})$. By Proposition III.8 and the termination condition for the while loop, we may set $R \leftarrow \epsilon/(10\log d) \ge \frac{90d}{|E(I_{k_\tau,t_\tau})|}$. By Theorem III.1,

$$\begin{aligned}&\Pr\left[\lambda_{\max}(Y_{k_\tau t_\tau}) \ge \epsilon \text{ and } \lambda_{\max}(W_{k_\tau t_\tau}) \le \epsilon^2/(10\log d)\right]\\ &\le d\exp\left(\frac{-\epsilon^2/2}{\epsilon^2/(10\log d) + \epsilon^2/(30\log d)}\right)\\ &\le 1/d^2\end{aligned}$$

Therefore,

$$\Pr_{T \text{ uniform}}[\lambda_{\max}(Y_{k_\tau t_\tau}) \ge \epsilon] \le 1/d^2 + 1/d^3 \le 2/d^2$$

Now, switch uniform spanning tree sampling to κ_0-approximate random spanning tree sampling. The total number of iterations is at most m, so the total TV distance of the joint distribution sampled throughout all iterations is at most $m\kappa_0$. Therefore,

$$\Pr_{T\ \kappa_0\text{-uniform}}[\lambda_{\max}(Y_{k_\tau t_\tau})] \le 2/d^2 + m\kappa_0 \le 3/d^2$$

In particular, with probability at least $1 - 3/d^2$,

$$\left|\frac{x_{H_{k_\tau}} L^+_{H_{k_\tau}} x_{H_{k_\tau}}}{x^T L_G^+ x} - 1\right| \le \lambda_{\max}(M_{k_\tau}) = \lambda_{\max}(Y_{k_\tau t_\tau}) \le \epsilon$$

for all $x \in \mathcal{S}$, as desired.

Runtime. Each while loop iteration eliminates at least $K(|E(H_k)|)/8$ edges from H_k in expectation. By Azuma's Inequality, this means that there are at most

$$O((\log n) \max_{z \le |E(G)|} z/K(z))$$

iterations with high probability. Each iteration samples one spanning tree, calls the oracle once, and does a linear amount of additional work, yielding the desired runtime. ∎

function SLOWORACLE$(I, \mathcal{S})$
Input: A graph I and a subspace $\mathcal{S} \subseteq \mathbb{R}^{V(I)}$
Output: A set Z of edges satisfying the steady oracle definition
Return all edges $e \in E(I)$ with $\max_{x\in\mathcal{S}} \frac{(x^T L_I^+ b_e)^2}{r_e(x^T L_I^+ x)} \le \frac{2\dim(\mathcal{S})}{|E(I)|}$
end function

A. Slow Oracle and Proof of Existence

In this section, we prove Theorem I.2 by exhibiting a $(2,1)$-steady oracle SlowOracle$(I, \mathcal{S})$. The oracle just returns all edges in the bottom half by maximum energy fraction:

To lower bound the number of edges added to Z, we use the following result and Markov's Inequality:

Proposition III.9.

$$\sum_{f\in E(I)} \max_{x\in\mathcal{S}} \frac{(x^T L_I^+ b_f)^2}{r_f(x^T L_I^+ x)} = \textit{dim}(\mathcal{S})$$

Proof:

Let Y_I be a $V(I) \times \dim(\mathcal{S})$-matrix consisting of a basis $(y_i)_{i=1}^d$ for $\mathcal{S}$ with $y_i^T L_I^+ y_j = 0$ for all $i \ne j \in [\dim(\mathcal{S})]$ and $y_i^T L_I^+ y_i = 1$ for all $i \in [\dim(\mathcal{S})]$. By Proposition III.6,

$$\begin{aligned}\sum_{f\in E(I)} \max_{x\in\mathcal{S}} \frac{(x^T L_I^+ b_f)^2}{r_f(x^T L_I^+ x)} &= \sum_{f\in E(I)} \frac{b_f^T L_I^+ Y_I Y_I^T L_I^+ b_f}{r_f}\\ &= \sum_{f\in E(I)} \text{trace}\left(\frac{b_f^T L_I^+ Y Y^T L_I^+ b_f}{r_f}\right)\\ &= \sum_{f\in E(I)} \text{trace}\left(\frac{L_I^+ Y_I Y_I^T L_I^+ b_f b_f^T}{r_f}\right)\\ &= \text{trace}(L_I^+ Y_I Y_I^T)\\ &= \sum_{i=1}^d y_i^T L_I^+ y_i\\ &= \dim(\mathcal{S})\end{aligned}$$

as desired. ∎

Now, we prove Theorem I.2:

Proof of Theorem I.2:

By Lemma III.3, it suffices to show that SlowOracle is a $(2,1)$-steady oracle.

Size of Z. By Markov's Inequality and Proposition III.9, $|Z| \ge |E(I)|/2$.

Leverage score stability. We are only interested in $i = 0$, for which the "Leverage score" input condition immediately implies the "Leverage score stability" guarantee.

Martingale change stability. We are only interested in $i = 0$. The return statement specifies the "Martingale change stability" guarantee for $\rho = 2$. ∎

IV. Fast Oracle

In this section, we give a $(O(\log^3 n), \Omega(z/\log^3 n))$-steady oracle `FastOracle` that proves Theorem I.3 when plugged into `SubspaceSparsifier`. To do this, we use localization [SRS17] to find a set of edges whose leverage scores and martingale changes do not change much over time. We use sketching and Lemma I.4 to find these edges efficiently. This section can be described using the *flexible function* framework given in [Sch17], but we give a self-contained treatment here.

A. Efficient Identification of Low-Change Edges

`FastOracle` needs to find a large collection of edges whose electrical energies do not change over the course of many iterations. This collection exists by the following result:

Theorem IV.1 (Theorem 1 of [SRS17])**.** *Let I be a graph. Then for any vector $w \in \mathbb{R}^{E(I)}$,*

$$\sum_{e,f \in E(I)} w_e w_f \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq c_{local}(\log^2 n)||w||_2^2$$

for some constant c_{local}.

Plugging in $w \leftarrow \mathbf{1}^{E(I)}$ shows that at least half of the edges $e \in E(I)$,

$$\sum_{f \in E(I)} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 2c_{\text{local}} \log^2 n$$

We decrease this bound by subsampling the edges in I to obtain Z. To identify the edges with low sum, we use matrix sketching (for example, [Ind06]).

1) Approximation of Column Norms: Consider a graph I and a set $W \subseteq E(I)$. We can obtain multiplicative approximations the quantities $\sum_{f \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ for all $e \in W$ in near-linear time using ℓ_1-sketching [Ind06]. However, we actually need to multiplicatively approximate the quantities $\sum_{f \in W, f \neq e} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$. In particular, we need to estimate the ℓ_1 norm of the rows of the matrix M with $M_{ef} := \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ with the diagonal left out. To do this, we tile the matrix as described in Section 11.3.2 of [Sch17]:

- Do $\Theta(\log n)$ times:
  - Pick a random balanced partition (W_0, W_1) of W
  - For each $e \in W_0$, approximate $a_e \leftarrow \sum_{f \in Z_1} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$ using sketching
- For each $e \in W$, average the a_es together and scale up the average by a factor of 4 to obtain an estimate for $\sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}}$

The expected contribution of each off-diagonal entry is 1, while no diagonal entry can contribute. After $\Theta(\log n)$ trials, the averages concentrate by Chernoff bounds and a union bound. We use this construction to prove the following result:

Proposition IV.2. *There is a near-linear time algorithm $(a_e)_{e \in W} \leftarrow$ `ColumnApx`(I, W) that takes a graph I and a set of edges $W \subseteq E(I)$ and returns estimates a_e for which*

$$a_e/2 \leq \sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq 3a_e/2$$

for all $e \in W$.

For a full proof of this proposition, see the full version.

2) Construction of Concentrated Edges: Now, we subsample localized sets:

Proposition IV.3. *Given a graph I and $\gamma \in (0, 1)$, there is a set of edges $W \subseteq E(I)$ with two properties:*

- *(Size) $|W| \geq (\gamma/4)|E(I)|$*
- *(Value) For all $e \in W$, $\sum_{f \neq e \in W} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq \psi$ for all $e \in W$, where $\psi := 100 c_{local} \gamma (\log^2 n)$*

Furthermore, there is an $\tilde{O}(|E(I)|/\gamma)$-expected time algorithm `Subsample`(I, γ) that produces W.

```
1: function SUBSAMPLE(I, γ)
2:     while W does not satisfy Proposition IV.3 do
3:         W_0 ← random subset of E(I), with each edge
   of e ∈ E(I) included i.i.d. with probability 2γ
4:         (a_e)_{e∈W_0} ← ColumnApx(I, W_0)
5:         W ← set of edges e ∈ W_0 with a_e ≤ ψ/2
6:     end while
7:     Return W
8: end function
```

To show that this algorithm works, we just need to show that this while loop condition is met with probability at least $\Omega(\gamma)$ during each iteration. The crux of the proof is the birthday paradox. Since edges are selected for inclusion in W_0 independently and the sum that a_e estimates is over edges f not including e, the expectation of a_e for W_0 is a 2γ factor smaller than the expectation of a_e for W, which is at most $O(\log^2 n)$ for a large fraction of edges in W by Theorem IV.1. For a full proof of Proposition IV.3, see the full version.

B. `FastOracle`

We now implement the $(\Theta(\log^3 n), \Theta(z/\log^3 n))$-steady oracle `FastOracle`. It starts by finding a set W guaranteed by Proposition IV.3 with $\gamma = \Theta(1/\log^3 n)$. It then further restricts W down to the set of edges satisfying "Martingale change stability" for I_0 and returns that set. By Sherman-Morrison, the "Value" guarantee of Proposition IV.3 ensures

that these edges continue to satisfy the "Martingale change stability" guarantee even after conditioning on edges in Z.

function FASTORACLE($I, \mathcal{S}$)
  Input: a graph I with leverage scores in $[3/16, 13/16]$ and a subspace $\mathcal{S} \subseteq V(I)$ with $\mathcal{S} := \mathbb{R}^S \times \mathbf{0}^{V(I)\setminus S}$ for some $S \subseteq V(I)$
  Output: a set $Z \subseteq E(I)$ satisfying the steady oracle guarantees
  $W \leftarrow$ Subsample(I, γ), where $\gamma := 1/(10000 c_{\text{local}}(\log^3 n))$
  $\{\nu_e\}_{e \in E(I)} \leftarrow$ DiffApx($I, S, 1/4, 1/m^5$)
  return all $e \in W$ for which $\nu_e \leq \frac{4|S|}{|W|}$
end function

To ensure that DiffApx is applicable, note the following equivalence to what its approximating and the quantity in the "Martingale change stability" guarantee:

Proposition IV.4.

$$\max_{x \in \mathbb{R}^S} \frac{(x^T L_H^+ b_f)^2}{r_f (x^T L_H^+ x)} = \frac{b_f^T L_H^+ b_f}{r_f} - \frac{b_f^T L_{H/S}^+ b_f}{r_f}$$

Proof: See full version. ∎

To analyze FastOracle, we start by showing that any set of localized edges remains localized under random edge modifications. The main tool behind the proof is the Sherman-Morrision rank 1 update formula. For a full proof, see the full version:

Proposition IV.5. *Consider a graph I and a set of edges $Z \subseteq E(I)$ that satisfy the following two initial conditions:*

- *(Initial leverage scores)* $\texttt{lev}_I(e) \in [3/16, 13/16]$ *for all* $e \in Z$.
- *(Initial localization)* $\sum_{f \in Z} \frac{|b_e^T L_I^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq \tau$ *for all* $e \in Z$, *where* $\tau = \frac{1}{10000 \log n}$.

Sample a sequence of minors $\{I_k\}_{k \geq 0}$ of I and sets $Z_k \subseteq E(I_k)$ by letting $I_0 := I$ and for each $k \geq 0$, sampling a uniformly random edge $e_k \in Z_k$, letting $I_{k+1} \leftarrow I_k \backslash e_k$ or $I_{k+1} \leftarrow I_k / e_k$ arbitrarily, and letting $Z_{k+1} \leftarrow Z_k \backslash e_k$. Then with probability at least $1 - 1/n^2$, the following occurs for all i:

- *(All leverage scores)* $\texttt{lev}_{I_k}(e) \in [1/8, 7/8]$ *for all* $e \in Z_k$.
- *(All localization)* $\sum_{f \in Z_k, f \neq e} \frac{|b_e^T L_{I_k}^+ b_f|}{\sqrt{r_e}\sqrt{r_f}} \leq \tau'$ *for all* $e \in Z_k$, *where* $\tau' = 2\tau$.

Now, we prove Theorem I.3. By Lemma III.3, it suffices to show that FastOracle is a $(O(\log^3 n), \Omega(z/\log^3 n)))$-steady oracle with runtime $\tilde{O}(|E(I)|)$. The "Size of Z" guarantee follows from Proposition III.9 and Markov's Inequality. Proposition IV.5 applies because of the "Leverage scores" condition for steady oracles and the "Value" guarantee applied to W. The "Leverage score stability" guarantee of steady oracles follows from the output conditions of Proposition IV.5, while the "Martingale change stability" guarantee follows from applications of Sherman-Morrison that exploit the "All localization" guarantee. See the full version for a full proof of Theorem I.3.

ACKNOLEDGEMENTS

We thank Gramoz Goranci, Jason Li, and Richard Peng for helpful discussions.

REFERENCES

[Ach01] Dimitris Achlioptas. Database-friendly random projections. In *Proceedings of the 20th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS)*, 2001.

[AGK14] Alexandr Andoni, Anupam Gupta, and Robert Krauthgamer. Towards $(1 + \epsilon)$-approximate flow sparsifiers. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 279–293, 2014.

[CGH16] Yun Kuen Cheung, Gramoz Goranci, and Monika Henzinger. Graph minors for preserving terminal distances approximately - lower and upper bounds. In *43rd International Colloquium on Automata, Languages, and Programming, ICALP 2016, July 11-15, 2016, Rome, Italy*, pages 131:1–131:14, 2016.

[Chu12] Julia Chuzhoy. On vertex sparsifiers with steiner nodes. In *Proceedings of the 44th Symposium on Theory of Computing Conference, STOC 2012, New York, NY, USA, May 19 - 22, 2012*, pages 673–688, 2012.

[CKM+14] 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 mlog1/2n time. In *Proceedings of the Forty-sixth Annual ACM Symposium on Theory of Computing*, STOC '14, pages 343–352, New York, NY, USA, 2014. ACM.

[CL06] Fan Chung and Linyuan Lu. Concentration inequalities and martingale inequalities: a survey. *Internet Math.*, 3(1):79–127, 2006.

[CLLM10] Moses Charikar, Tom Leighton, Shi Li, and Ankur Moitra. Vertex sparsifiers and abstract rounding algorithms. *CoRR*, abs/1006.4536, 2010.

[DKP+17] David Durfee, Rasmus Kyng, John Peebles, Anup B. Rao, and Sushant Sachdeva. Sampling random spanning trees faster than matrix multiplication. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 730–742, New York, NY, USA, 2017. ACM.

[DPPR17] David Durfee, John Peebles, Richard Peng, and Anup B. Rao. Determinant-preserving sparsification of SDDM matrices with applications to counting and sampling spanning trees. *CoRR*, abs/1705.00985, 2017.

[EGK+14] Matthias Englert, Anupam Gupta, Robert Krauthgamer, Harald Räcke, Inbal Talgam-Cohen, and Kunal Talwar. Vertex sparsifiers: New results from old techniques. *SIAM J. Comput.*, 43(4):1239–1262, 2014.

[GHP17] Gramoz Goranci, Monika Henzinger, and Pan Peng. Improved Guarantees for Vertex Sparsification in Planar Graphs. In Kirk Pruhs and Christian Sohler, editors, *25th Annual European Symposium on Algorithms (ESA 2017)*, volume 87 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 44:1–44:14, Dagstuhl, Germany, 2017. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[GHP18] Gramoz Goranci, Monika Henzinger, and Pan Peng. Dynamic effective resistances and approximate schur complement on separable graphs. *CoRR*, abs/1802.09111, 2018.

[Ind06] Piotr Indyk. Stable distributions, pseudorandom generators, embeddings, and data stream computation. *J. ACM*, 53(3):307–323, 2006.

[JL84] William B Johnson and Joram Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. *Contemporary Mathematics*, 26(189-206):1, 1984.

[Kir47] G. Kirchhoff. Ueber die Auflösung der Gleichungen, auf welche man bei der Untersuchung der linearen Vertheilung galvanischer Ströme geführt wird. *Annalen der Physik*, 148:497–508, 1847.

[KLP+16] 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 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 842–850, 2016.

[KMST10] Alexandra Kolla, Yury Makarychev, Amin Saberi, and Shang-Hua Teng. Subgraph sparsification and nearly optimal ultrasparsifiers. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 57–66, 2010.

[KS16] R. Kyng and S. Sachdeva. Approximate gaussian elimination for laplacians - fast, sparse, and simple. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 573–582, Oct 2016.

[LM10] Frank Thomson Leighton and Ankur Moitra. Extensions and limits to vertex sparsification. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pages 47–56, 2010.

[MM10] Konstantin Makarychev and Yury Makarychev. Metric extension operators, vertex sparsifiers and lipschitz extendability. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pages 255–264, 2010.

[Moi09] Ankur Moitra. Approximation algorithms for multicommodity-type problems with guarantees independent of the graph size. In *50th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2009, October 25-27, 2009, Atlanta, Georgia, USA*, pages 3–12, 2009.

[MP13] Gary L. Miller and Richard Peng. Approximate maximum flow on separable undirected graphs. In *Proceedings of the Twenty-fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '13, pages 1151–1170, Philadelphia, PA, USA, 2013. Society for Industrial and Applied Mathematics.

[RST14] Harald Räcke, Chintan Shah, and Hanjo Täubig. Computing cut-based hierarchical decompositions in almost linear time. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2014, Portland, Oregon, USA, January 5-7, 2014*, pages 227–238, 2014.

[Sch17] Aaron Schild. An almost-linear time algorithm for uniform random spanning tree generation. *CoRR*, abs/1711.06455, 2017.

[SRS17] Aaron Schild, Satish Rao, and Nikhil Srivastava. Localization of electrical flows. *CoRR*, abs/1708.01632, 2017.

[SS08] Daniel A. Spielman and Nikhil Srivastava. Graph sparsification by effective resistances. In *Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, British Columbia, Canada, May 17-20, 2008*, pages 563–568, 2008.

[ST14] Daniel A. Spielman and Shang-Hua Teng. Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems. *SIAM J. Matrix Analysis Applications*, 35(3):835–885, 2014.

[Tro11] Joel Tropp. Freedman's inequality for matrix martingales. *Electron. Commun. Probab.*, 16:262–270, 2011.