Abstract

We show that every graph is spectrally similar to the union of a constant number of forests. Moreover, we show that Spielman-Srivastava sparsifiers are the union of $O(\log n)$ forests. This result can be used to estimate boundaries of small subsets of vertices in nearly optimal query time.

1 Introduction

A sparsifier of a graph $G = (V, E, c)$, is a sparse graph $H$ with similar properties. In this paper, we consider spectral sparsifiers, defined by Spielman and Teng in [ST04]. A graph $H$ is said to be a $(1 + \epsilon)$-spectral sparsifier of a graph $G$ if, for all vectors $x \in \mathbb{R}^{|V|}$,

$$\left(\frac{1}{1 + \epsilon}\right)x^T L_G x \leq x^T L_H x \leq (1 + \epsilon)x^T L_G x. \quad (1)$$

Here, $L_G$ and $L_H$ are the Laplacians of $G$ and $H$ respectively.

Spielman and Srivastava proved in [SS08] that every graph has a spectral sparsifier with $O\left(\frac{n \log n}{\epsilon^2}\right)$ edges using an edge sampling routine. Batson, Spielman, and Srivastava proved in [BSS09] that there exist $(1 + \epsilon)$-spectral sparsifiers of graphs with $O(n/\epsilon^2)$ edges. Furthermore, the Marcus-Spielman-Srivastava proof of the Kadison-Singer conjecture in [MSS13] can be used to show that the edge sampling routine of [SS08] gives an $(1 + \epsilon)$-spectral sparsifier with $O(n/\epsilon^2)$ edges, with non-zero probability. [S13]

Our primary result is to show that Spielman-Srivastava sparsifiers can be written as the union of $O\left(\frac{\log n}{\epsilon^2}\right)$ forests, and that Marcus-Spielman-Srivastava sparsifiers can be
written as the union of \( O(1/\epsilon^2) \) forests. This is as tight a bound as we can hope for up to \( \epsilon \) factors.

Our result can be applied to approximating cut queries efficiently. As shown by Andoni, Krauthgamer, and Woodruff in [AKW14], any sketch of a graph that w.h.p. preserves all cuts in an \( n \)-vertex graph must be of size \( \Omega(n/\epsilon^2) \) bits. We show that the Spielman-Srivastava sparsifiers, in addition to achieving nearly optimal construction time and storage space, can also be made to achieve the nearly optimal query time \( O(|S| \frac{\log n}{\epsilon^2}) \) when estimating the boundary of \( S \subseteq V \), compared to the trivial query time of \( O(n \frac{\log n}{\epsilon^2}) \).

## 2 Preliminaries

### 2.1 Electrical Flows and Effective Resistance

Let graph \( G = (V, E, c) \) have edge weights \( c_e \), where \( c_e \) is the conductance of each edge. Define the resistance \( r_e \) on each edge to be \( \frac{1}{c_e} \).

Let \( L_G \) be the Laplacian of graph \( G \). Let \( \vec{v} \in \mathbb{R}^{|V|} \) be the vector of voltages on the vertices of \( V \).

Let the vector \( \vec{\chi} \) denote the vector of excess demand on each vertex. It’s well known in the theory of electrical networks that

\[
L_G \vec{v} = \vec{\chi},
\]

(2)

or equivalently,

\[
\vec{v} = L_G^+ \vec{\chi}
\]

(3)

where \( L_G^+ \) is the Moore-Penrose pseudo-inverse of \( L_G \). For edge \( e = (i, j) \) with \( i, j \in V \), the effective resistance \( R_e(G) \) is defined as

\[
R_e(G) = \frac{\vec{\chi}^T L_G^+ \vec{\chi}}{2}
\]

(4)

where

\[
\vec{\chi} := \begin{cases} 
1 & x = i \\
-1 & x = j \\
0 & \text{otherwise}
\end{cases}
\]

(5)

The effective resistance of edge \( e \) can be interpreted as the voltage drop across that edge given an flow of 1 unit of current from \( i \) to \( j \).

When the underlying choice of graph \( G \) is clear, \( R_e(G) \) will be shortened to \( R_e \).

**Lemma 2.1.** For all graphs \( H \) that spectrally sparsify \( G \),

\[
\left( \frac{1}{1 + \epsilon} \right) R_e(H) < R_e(G) < (1 + \epsilon) R_e(H)
\]

(6)
Proof. This follows immediately from Equation 1 and substituting

\[ R_e(G) = \frac{\chi^T L_G^+ \chi}{2} \]

and

\[ R_e(H) = \frac{\chi^T L_H^+ \chi}{2}. \]

where \( R_e(H) \) denotes the effective resistance of edge \( e \) in \( H \) and \( R_e(G) \) denotes the effective resistance of \( e \) in \( G \).

### 2.2 The Spielman-Srivastava Sparsifier

Spielman and Srivastava showed in [SS08] that any graph can be sparsified with high probability using the following routine, for a large enough constant \( C \):

- For each edge, assign it a probability \( p_e := \frac{R_e c_e}{(n-1)} \), where \( R_e \) is the effective resistance of that edge and \( c_e \) is the conductance (the inverse of the actual resistance) of that edge. Create a distribution on edges where each edge occurs with probability equal to \( p_e \).

- Weight each edge to have conductance \( \frac{c_e \epsilon^2}{(Cn \log n) p_e} \), and sample \( Cn \log n / \epsilon^2 \) edges from this distribution.

We call such a scheme a Spielman-Srivastava sparsifying routine. Note that this scheme allows for multiple edges between any two vertices.

**Remark 2.2.** Sampling by approximate effective resistances (as Spielman and Srivastava did in their original paper [SS08]) will work in place of using exact values for effective resistances. The results in our paper will still go through; an approximation will still ensure that every edge has a relatively large weighting, which is what the result in our paper depends on.

### 2.3 The Marcus-Spielman-Srivastava Sparsifier

The following scheme from [SL13] produces a sparsifier with non-zero probability, for sufficiently large constants \( C \):

- For each edge, assign it a probability \( p_e := \frac{R_e c_e}{(n-1)} \), where \( R_e \) is the effective resistance of that edge and \( c_e \) is the conductance (inverse of actual resistance) of that edge. Create a distribution on edges where each edge occurs with probability equal to \( p_e \).

- Weight each edge to have conductance \( \frac{c_e \epsilon^2}{p_e (Cn)} \), and sample \( Cn / \epsilon^2 \) edges from this distribution.
We call such a scheme a Marcus-Spielman-Srivastava sparsifying routine. Note that this scheme allows for multiple edges between any two vertices.

Note that the probability this routine returns a sparsifier may be exponentially small, and there is no known efficient algorithm to actually find such a sparsifier, making the [SS08] result more algorithmically relevant.

2.4 Uniform Sparsity and Low Arboricity

Definition 2.3. The arboricity of a graph $G$ is the equal to the minimum number of forests its edges can be decomposed into.

Definition 2.4. A graph $G = (V, E, c)$ is said to be $c$-uniformly sparse if, for all subsets $V' \subset V$, the subgraph induced on $G$ by $V'$ contains no more than $c \cdot |V'|$ edges.

Lemma 2.5. Uniform Sparsity implies Low Arboricity. That is, if $G$ is $c$-uniformly sparse, then the arboricity of $G$ is no greater than $2c$.

This statement is proven in 4.2.

3 The Main Result

First we establish some preliminary lemmas.

Lemma 3.1. (Foster’s Resistance Theorem) Let $G = (V, E, c)$ be any graph. Then

$$\sum_{e \in E} R_e c_e = n - 1.$$ \hfill (7)

where $n := |V|$. \[F61\]

Lemma 3.2. (Effective Resistances of edges in a subgraph are higher than in the original graph) Let $H$ be a subgraph of $G = (V, E, c)$, where $L_H$ is treated as a linear operator from $\mathbb{R}^{|V|}$ to $\mathbb{R}^{|V|}$. Then

$$x^T L_H^+ x \geq x^T L_G^+ x$$ \hfill (8)

for all $x \in \mathbb{R}^{|V|}$ orthogonal to the nullspace of $L_H$.

Proof. Let $x = \frac{4}{3} y$ and $y = (L_G^+) \frac{4}{3} x$. Now Equation \[S\] is equivalent to the equation

$$y^T \left( L_G^+ L_H^+ L_G^+ \right) y \geq y^T y$$ \hfill (9)

holding true for all vectors $y \in \mathbb{R}^{|V|}$.

Since $x$ is assumed to be orthogonal to the nullspace of $L_H$ (which equals the nullspace of $L_H^+$), it follows that $y$ is orthogonal to the nullspace of $L_G^+ L_H^+ L_G^+$. Therefore it suffices to
show that all the non-zero eigenvalues of $L_G^1 L_H^1 L_G^1$ are greater than 1. This is equivalent to showing that all the non-zero eigenvalues of $(L_G^1)^{1/2} L_H (L_G^1)^{1/2}$ are less than 1, a fact which follows immediately from Rayleigh monotonicity.

Lemma 3.3. Let $V' \subset V$. Let $G'$ be the subgraph of $G$ induced by $V'$, and let $E'$ be the edges of the induced subgraph. Then

$$\sum_{e \in E'} R_e(G)c_e \leq |V'|-1 \quad (10)$$

Proof. Note that $R_e = \frac{1}{2} \chi L_G \chi$, where $\chi$ is defined as in Equation 5. Since $e$ is an edge of subgraph $H$, it follows that $\chi$ is orthogonal to the nullspace of $L_H$. Thus we can apply Lemma 3.2 and Lemma 3.1 to show that

$$\sum_{e \in E'} R_e(G')c_e = \sum_{e \in E'} R_e(G')c_e \leq |V'|-1 \quad (11)$$

as desired. \qed

Theorem 3.4. Marcus-Spielman-Srivastava sparsifiers are $O(1/\epsilon^2)$-uniformly sparse.

Proof. For each edge included in the graph by the Marcus-Spielman-Srivastava sampling scheme, they’re included in the graph with weight $c_e^2/\epsilon^2$. Therefore, by Lemma 2.1, the value of $R_e(H)c_e(H)$ on edge $e$ is within a $(1 + \epsilon)$ multiple of

$$R_e(G)c_e(H) = R_e \cdot \frac{c_e^2}{(Cn)p_e} = R_e \cdot \frac{c_e^2}{Cn} \cdot \frac{(n-1)}{Re_c} \geq \frac{e^2}{2C}. \quad (12)$$

Here, $R_e(H)$ and $c_e(H)$ denote the effective resistance and conductance of edge $e$ in graph $H$ respectively, and $R_e$ and $c_e$ denote the effective resistance and conductance of edge $e$ in graph $G$ respectively.

Using Lemma 3.3 it follows that the subgraph induced by $V'$ has no more than $2C(|V'|-1)/\epsilon^2$ edges. This implies that any subgraph of a Marcus-Spielman-Srivastava sparsifier is sparse. \qed

Corollary 3.5. Marcus-Spielman-Srivastava sparsifiers have $O(1/\epsilon^2)$ arboricity.

Theorem 3.6. Spielman-Srivastava sparsifiers are $O\left(\frac{\log n}{\epsilon^2}\right)$-uniformly sparse.

Proof. The proof is identical to the the proof of Theorem 3.4 with $Cn$ replaced with $Cn \log n$. \qed

Corollary 3.7. Spielman-Srivastava sparsifiers have arboricity $O\left(\frac{\log n}{\epsilon^2}\right)$. 


4 Applications to Approximating Cut Queries

Definition 4.1. We say that a total ordering \( \prec \) of the vertices of a graph is \( c \)-treelike if every vertex \( u \) has at most \( c \) neighbors \( v \) such that \( u \prec v \).

Lemma 4.2. Every \( c \)-uniformly sparse graph has a \( 2c \)-treelike ordering. Moreover, this ordering can be computed in linear time.

Proof. Let \( G \) be a \( c \)-uniformly sparse graph. Let \( v \) be the minimum degree vertex of \( G \). Note that \( d(v) \leq 2c \). We set \( v \) to be the smallest in the ordering \( \prec \) and then recursively construct the remainder of the ordering on \( G' = G \setminus \{v\} \).

Lemma 4.3. Let \( G = (V, E) \) be \( c \)-uniformly sparse. After preprocessing in linear time and space, we can answer queries about the boundaries of subsets of vertices of \( G \) in \( O(ck) \) time, where \( k \) is the size of the queried subset.

Proof. Using Lemma 4.2 we first compute a \( 2c \)-treelike ordering \( \prec \) of \( V \). For every vertex \( u \in V \), we store a list of edges \( (u, v) \in E \) such that \( u \prec v \). We also compute and store the weighted degree \( wd(v) \) for every vertex of \( G \).

Assume we are given \( S \subseteq V \). We first compute the total weight \( s_{\text{internal}} \) of edges internal to \( S \). To this end, for every vertex \( u \in S \) we go through its neighbors that are larger in the ordering \( \prec \) and sum up the weights of edges that lead to \( S \). Note that every edge in \( S \times S \) will be encountered exactly once. The boundary of \( S \) can be computed as

\[
s_{\text{cut}} := \left( \sum_{v \in S} wd(v) \right) - 2s_{\text{internal}}.
\]

Corollary 4.4. Given a graph with \( n \) vertices, there exists a data structure that:

- achieves the construction time, storage space, and cut approximation guarantees of Spielman-Srivastava sparsifiers, and

- can compute approximate weights of cuts in \( O(k \frac{\log n}{\epsilon^2}) \) time, where \( k \) is the size of the smaller side of the cut.

5 Final Note

Similar techniques to those presented can show that if the vertices of graph \( G \) are ordered by the sum of \( R_{e \in e} \) on edges that have an endpoint of that vertex, any graph has a sparsifier that can be written as the union of \( O(1/\epsilon^2) \) trees with that topological ordering on their vertices. The proof uses the same machinery as that presented above (with a slightly different use of the Marcus-Spielman-Srivastava sampling scheme), and we omit the full proof here.
References

[AKW14] Alexandr Andoni, Robert Krauthgamer, David Woodruff. The Sketching Complexity of Graph Cuts. *CoRR*, abs/1403.7058, 2014.

[BSS09] Joshua Batson, Daniel A. Spielman, and Nikhil Srivastava. Twice-Ramanujan Sparsifiers. *SIAM Review, Vol. 56, Issue 2, pp 315 – 334. 2014.*

[F61] R.M. Foster. An Extension of a Network Theorem IRE Trans. Circuit Theory, 8 (1961), pp.75-76.

[MSS13] Adam Marcus, Daniel A. Spielman, and Nikhil Srivastava. Interlacing Families II: Mixed Characteristic Polynomials and the Kadison-Singer Problem. To appear in Annals of Mathematics.

[SS08] Daniel A. Spielman and Nikhil Srivastava. Graph Sparsification by Effective Resistances. Conference version appeared in STOC ’08.

[ST04] Daniel A. Spielman and Shan-Hua Teng. Nearly-Linear Time Algorithms for Graph Partitioning, Graph Sparsification, and Solving Linear Systems *Proceedings of the 36th Annual ACM Symposium on Theory of Computing, pp. 81 – 90. 2004*

[S13] Nikhil Srivastava. Discrepancy, Graphs, and the Kadison-Singer Problem.  http://windowsontheory.org/2013/07/11/discrepancy-graphs-and-the-kadison-singer-conjecture-2/