Single Pass Spectral Sparsification in Dynamic Streams

2014.10.21

M. Kapralov, Y.T. Lee, C. Musco, C. Musco, A. Sidford
Massachusetts Institute of Technology

1-Pass Spectral Sparsification in Dynamic Streams

Overview

- In $\tilde{O}(n)$ space, maintain a graph compression from which we can always return a spectral sparsifier.

Main technique

- Use ℓ_2 heavy hitter sketches to sample by effective resistance in the streaming model.

1-Pass Spectral Sparsification in Dynamic Streams

Overview

- In $\tilde{O}(n)$ space, maintain a graph compression from which we can always return a spectral sparsifier.

Main technique

- Use ℓ_2 heavy hitter sketches to sample by effective resistance in the streaming model.

Outline

1. Graph Sparsification
2. Semi-Streaming Computational Model
3. Prior Work Review
4. Our Algorithm
   - Sampling in the Streaming Model
   - Recursive Sparsification [Li, Miller, Peng '12]

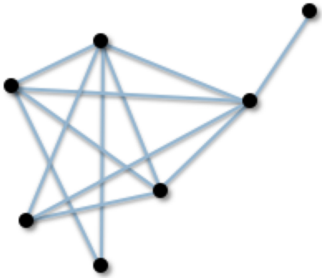
Overview

1. Graph Sparsification
2. Semi-Streaming Computational Model
3. Prior Work Review
4. Our Algorithm
   - Sampling in the Streaming Model
   - Recursive Sparsification [Li, Miller, Peng '12]

Graph Sparsification

General Idea

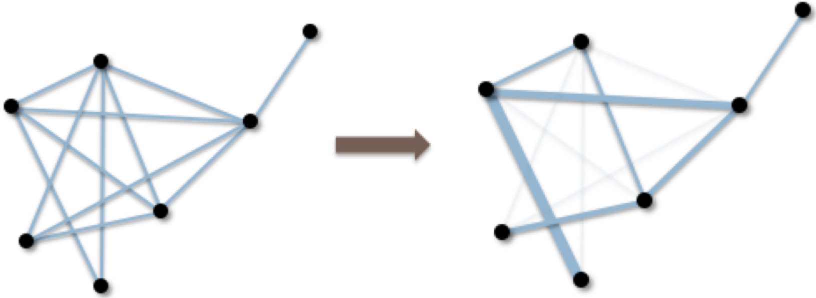
- Approximate a dense graph with a much sparser graph.
- Reduce $O(n^2)$ edges $\rightarrow$ $O(n \log n)$ edges



Graph Sparsification

General Idea

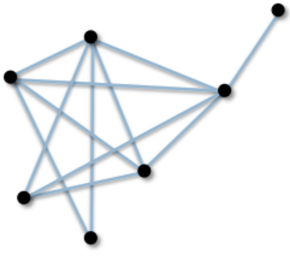
- Approximate a dense graph with a much sparser graph.
- Reduce $O(n^2)$ edges $\rightarrow$ $O(n \log n)$ edges



Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

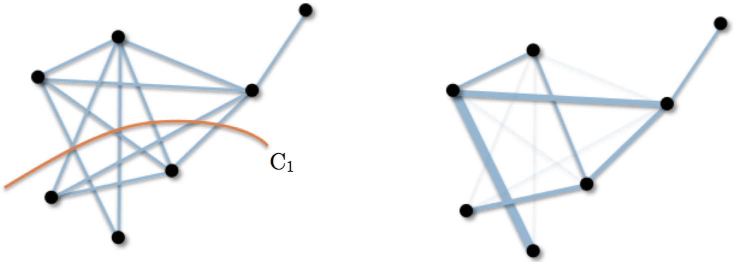


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

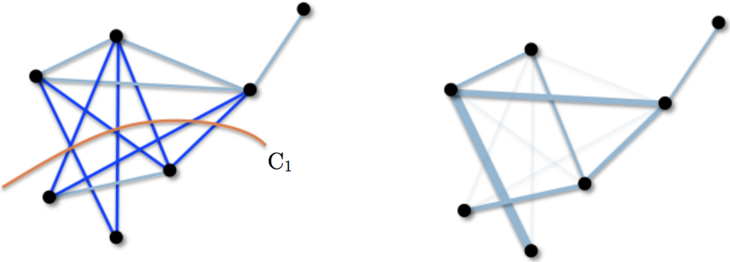


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

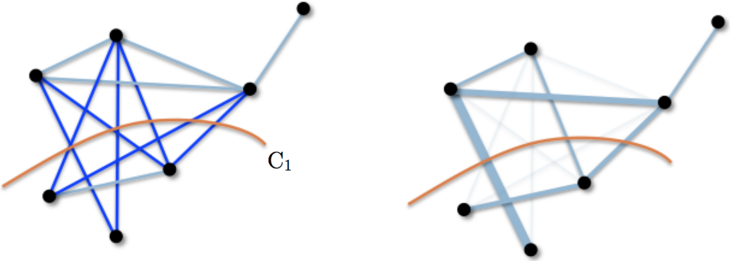


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

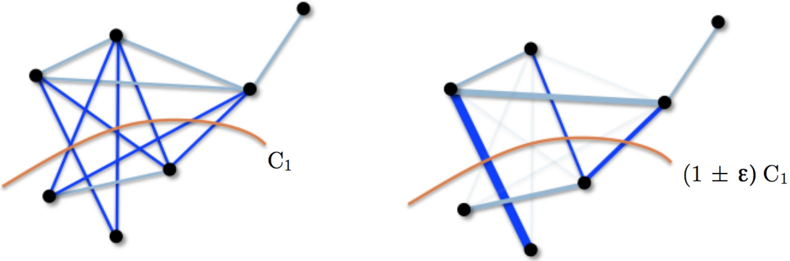


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

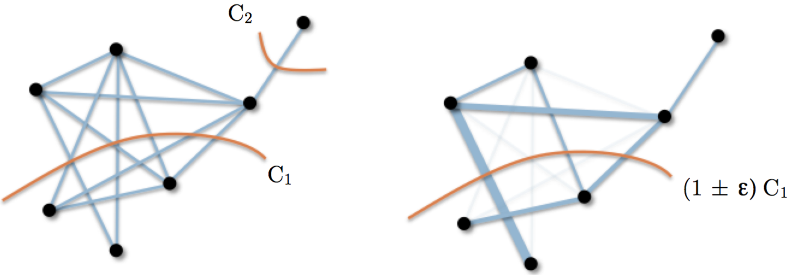


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

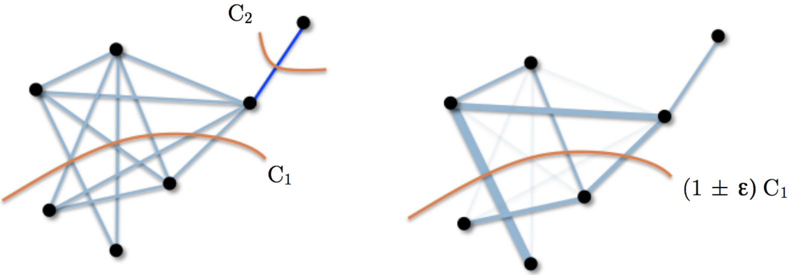


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

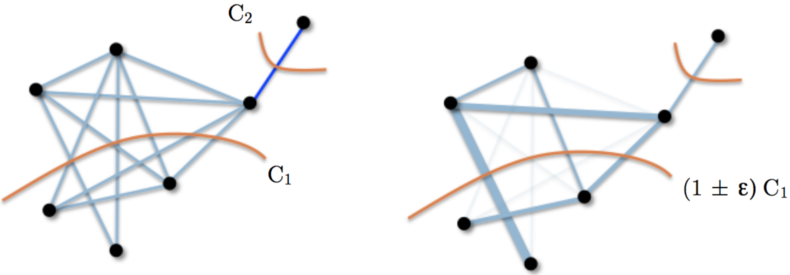


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

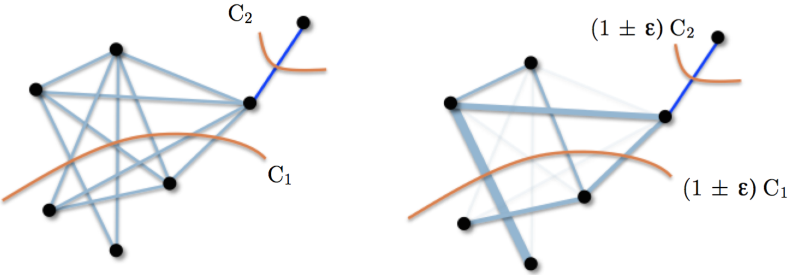


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Preserve *every* cut value to within $(1 \pm \varepsilon)$ factor

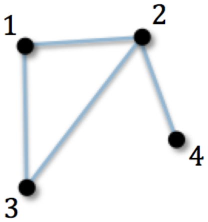


Applications: Minimum cut, sparsest cut, etc.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

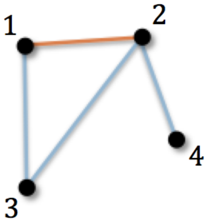


$$\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ \mathbf{B} \end{array}$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

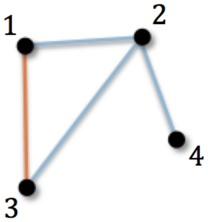


$$
\begin{array}{c}
 \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34}
\end{array}
\begin{array}{c}
\begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
\begin{bmatrix}
\mathbf{1} & \mathbf{-1} & \mathbf{0} & \mathbf{0} \\
1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 \\
0 & 1 & 0 & -1 \\
0 & 0 & 0 & 0
\end{bmatrix} \\
\mathbf{B}
\end{array}
$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

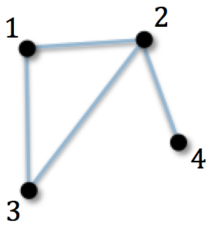


$$\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 1 & -1 & 0 & 0 \\ \mathbf{1} & \mathbf{0} & \mathbf{-1} & \mathbf{0} \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ \mathbf{B} \end{array}$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.



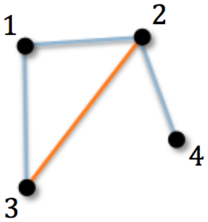
$$\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

B

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.



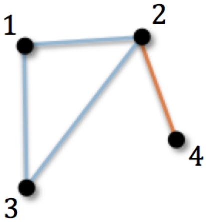
$$
\begin{array}{c c}
 & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
\begin{array}{c} e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} &
\begin{bmatrix}
1 & -1 & 0 & 0 \\
1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 \\
\mathbf{0} & \mathbf{1} & \mathbf{-1} & \mathbf{0} \\
0 & 1 & 0 & -1 \\
0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

$$\mathbf{B}$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.



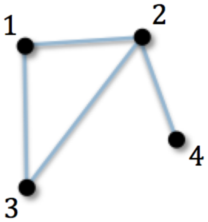
$$\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{-0} \\ 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

$\mathbf{B}$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.



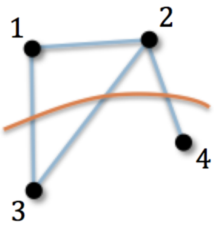
$$
\begin{array}{c c}
 & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
\begin{array}{c} e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} &
\begin{bmatrix}
1 & -1 & 0 & 0 \\
1 & 0 & -1 & 0 \\
0 & 0 & 0 & 0 \\
0 & 1 & -1 & 0 \\
0 & 1 & 0 & -1 \\
0 & 0 & 0 & 0
\end{bmatrix}
\end{array}
$$

$\mathbf{B}$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

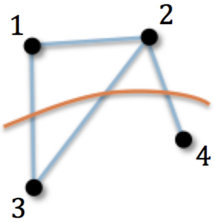


$$\begin{array}{c c} & \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \begin{array}{c} e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} & \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ & \mathbf{B} \end{array} \times \begin{array}{c} \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \\ \mathbf{x} \end{array}$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

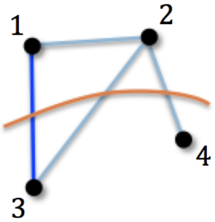


$$
\begin{array}{c}
\begin{array}{c|cccc}
 & v_1 & v_2 & v_3 & v_4 \\
\hline
e_{12} & 1 & -1 & 0 & 0 \\
e_{13} & 1 & 0 & -1 & 0 \\
e_{14} & 0 & 0 & 0 & 0 \\
e_{23} & 0 & 1 & -1 & 0 \\
e_{24} & 0 & 1 & 0 & -1 \\
e_{34} & 0 & 0 & 0 & 0
\end{array} \\
\mathbf{B}
\end{array}
\times
\underset{\mathbf{x}}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}}
=
\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}
$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

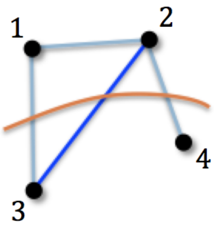


$$\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} \begin{array}{c} \begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\ \underbrace{\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}}_{\mathbf{B}} \end{array} \times \underbrace{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}}_{\mathbf{x}} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

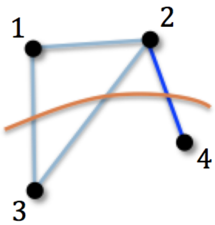


$$
\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array}
\begin{array}{c}
\begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
\mathbf{B}
\end{array}
\times
\underset{\mathbf{x}}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}}
=
\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}
$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.

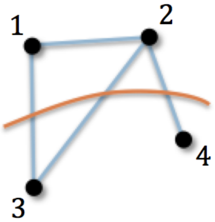


$$
\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array}
\begin{array}{c}
\begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array} \\
\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \\
\mathbf{B}
\end{array}
\times
\underset{\mathbf{x}}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}}
=
\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}
$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96)

- Let $\mathbf{B} \in \mathbb{R}^{\binom{n}{2} \times n}$ be the vertex-edge incidence matrix for a graph G.
- Let $\mathbf{x} \in \{0,1\}^n$ be an "indicator vector" for some cut.



$\|\mathbf{B}\mathbf{x}\|_2^2 = \text{cut value}$

$$\begin{array}{c} \\ e_{12} \\ e_{13} \\ e_{14} \\ e_{23} \\ e_{24} \\ e_{34} \end{array} \overset{\begin{array}{cccc} v_1 & v_2 & v_3 & v_4 \end{array}}{\underset{\mathbf{B}}{\begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 0 \end{bmatrix}}} \times \underset{\mathbf{x}}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96) So, $\|\mathbf{B}\mathbf{x}\|_2^2 =$ cut value.

Goal

Find some $\tilde{\mathbf{B}}$ such that, for all $\mathbf{x} \in \{0,1\}^n$,

$$(1-\varepsilon)\|\mathbf{B}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{B}}\mathbf{x}\|_2^2 \leq (1+\varepsilon)\|\mathbf{B}\mathbf{x}\|_2^2$$

- $\mathbf{x}^\top \tilde{\mathbf{B}}^\top \tilde{\mathbf{B}}\mathbf{x} \approx \mathbf{x}^\top \mathbf{B}^\top \mathbf{B}\mathbf{x}$.
- $\mathbf{L} = \mathbf{B}^\top \mathbf{B}$ is the *graph Laplacian*.

Graph Sparsification

Cut Sparsification (Benczúr, Karger '96) So, $\|\mathbf{Bx}\|_2^2 =$ cut value.

Goal

Find some $\tilde{\mathbf{B}}$ such that, for all $\mathbf{x} \in \{0,1\}^n$,

$$(1-\varepsilon)\|\mathbf{Bx}\|_2^2 \leq \|\tilde{\mathbf{B}}\mathbf{x}\|_2^2 \leq (1+\varepsilon)\|\mathbf{Bx}\|_2^2$$

- $\mathbf{x}^\top \tilde{\mathbf{B}}^\top \tilde{\mathbf{B}}\mathbf{x} \approx \mathbf{x}^\top \mathbf{B}^\top \mathbf{B}\mathbf{x}$.
- $\mathbf{L} = \mathbf{B}^\top \mathbf{B}$ is the *graph Laplacian*.

Graph Sparsification

Spectral Sparsification (Spielman, Teng '04)

Goal

Find some $\tilde{\mathbf{B}}$ such that, for all $\mathbf{x} \in \cancel{\{0,1\}^n}\ \mathbb{R}^n$,

$$(1-\varepsilon)\|\mathbf{Bx}\|_2^2 \leq \|\tilde{\mathbf{B}}\mathbf{x}\|_2^2 \leq (1+\varepsilon)\|\mathbf{Bx}\|_2^2$$

Applications: Anything cut sparsifiers can do, Laplacian system solves, computing effective resistances, spectral clustering, calculating random walk properties, etc.

Graph Sparsification

Spectral Sparsification (Spielman, Teng '04)

Goal

Find some $\tilde{\mathbf{B}}$ such that, for all $\mathbf{x} \in \cancel{\{0,1\}^n}\ \mathbb{R}^n$,

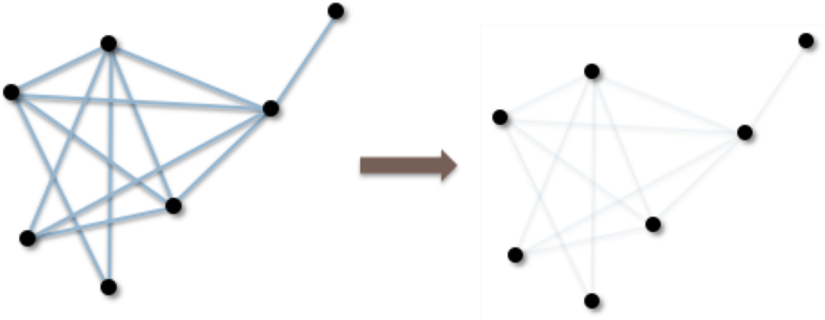
$$(1-\varepsilon)\|\mathbf{B}\mathbf{x}\|_2^2 \leq \|\tilde{\mathbf{B}}\mathbf{x}\|_2^2 \leq (1+\varepsilon)\|\mathbf{B}\mathbf{x}\|_2^2$$

Applications: Anything cut sparsifiers can do, Laplacian system solves, computing effective resistances, spectral clustering, calculating random walk properties, etc.

Graph Sparsification

How are sparsifiers constructed?

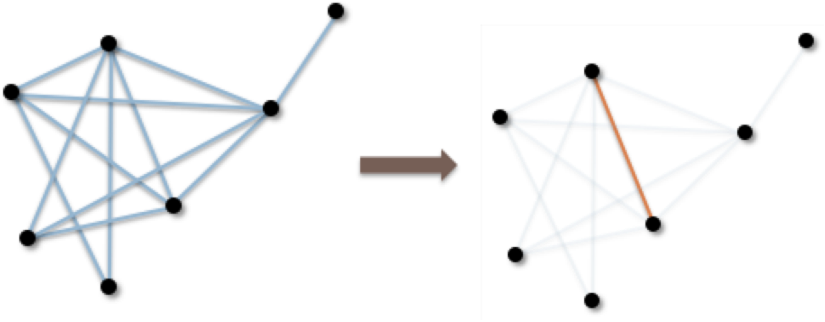
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

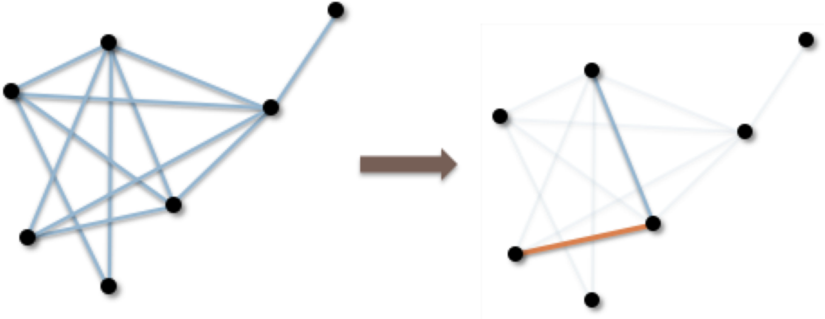
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

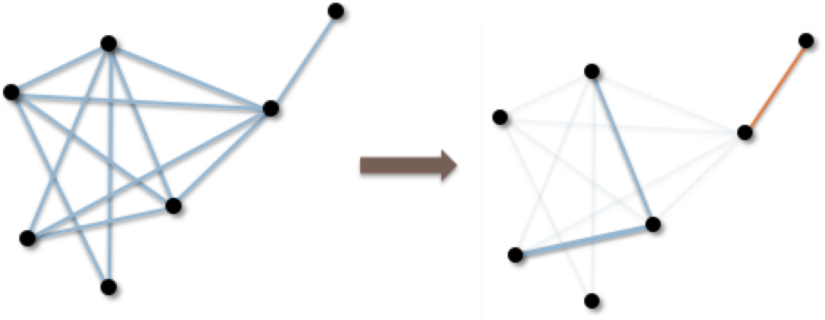
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

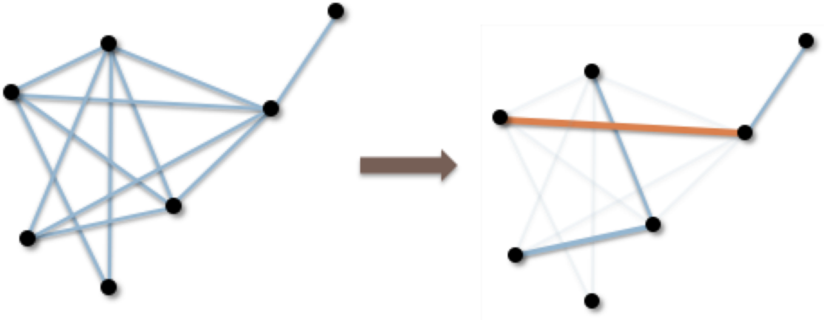
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

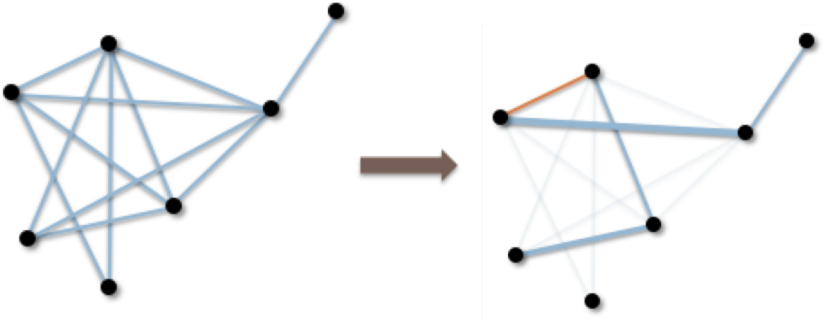
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

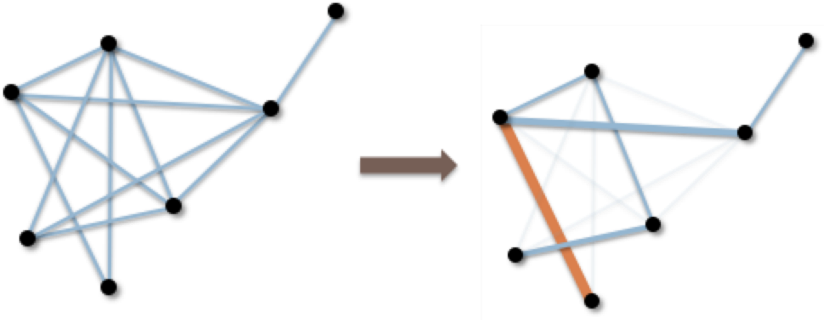
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

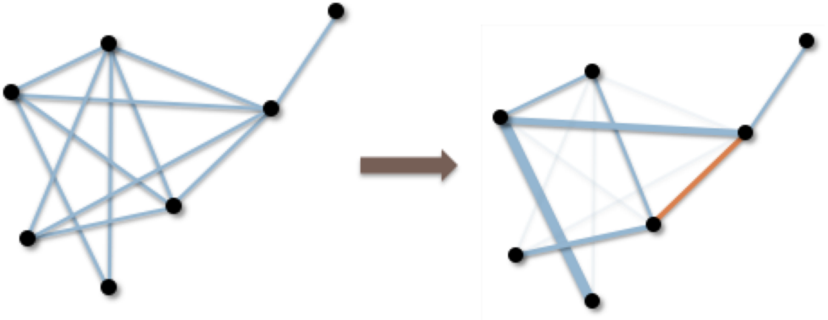
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

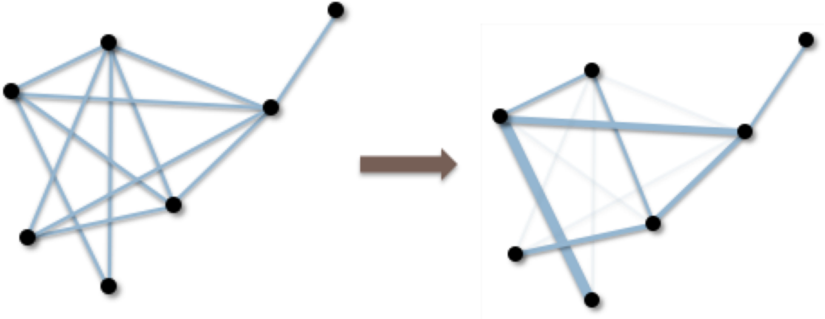
Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

Randomly sample edges (i.e. rows from $\mathbf{B}$):



Graph Sparsification

How are sparsifiers constructed?

Sampling probabilities:

- Connectivity for cut sparsifiers [Benczúr, Karger '96], [Fung, Hariharan, Harvey, Panigrahi '11].
- **Effective resistances** (i.e statistical leverage scores) for spectral sparsifiers [Spielman, Srivastava '08].

Actually oversample: by (effective resistance) $\times$ $O(\log n)$.
Gives sparsifiers with $O(n \log n)$ edges – reducing from $O(n^2)$.

Graph Sparsification

How are sparsifiers constructed?

Sampling probabilities:

- Connectivity for cut sparsifiers [Benczúr, Karger '96], [Fung, Hariharan, Harvey, Panigrahi '11].
- **Effective resistances** (i.e statistical leverage scores) for spectral sparsifiers [Spielman, Srivastava '08].

Actually oversample: by (effective resistance) $\times$ $O(\log n)$.
Gives sparsifiers with $O(n \log n)$ edges – reducing from $O(n^2)$.

Graph Sparsification

How are sparsifiers constructed?

Sampling probabilities:

- Connectivity for cut sparsifiers [Benczúr, Karger '96], [Fung, Hariharan, Harvey, Panigrahi '11].
- **Effective resistances** (i.e statistical leverage scores) for spectral sparsifiers [Spielman, Srivastava '08].

Actually oversample: by (effective resistance) $\times$ $O(\log n)$.
Gives sparsifiers with $O(n \log n)$ edges – reducing from $O(n^2)$.

Motivation

- Makes sense to compress a graph, but what if we cannot afford to store it in the first place?
- Is it possible to "sketch" a graph in small space by maintaining a sparsifier or some other representation?

Overview

Semi-Streaming Model

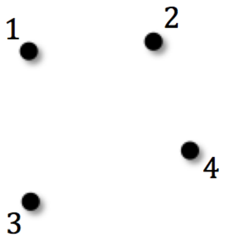
Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.

Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

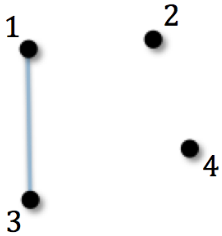
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

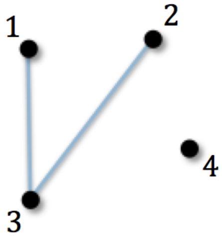
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

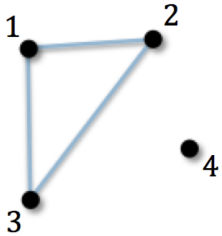
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

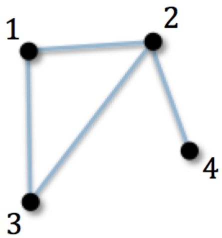
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

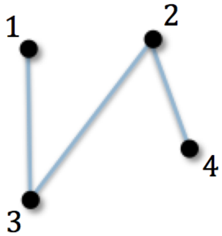
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

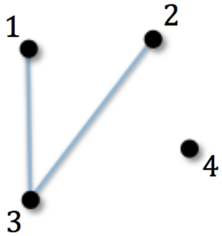
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

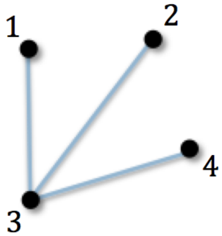
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

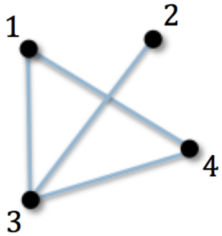
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

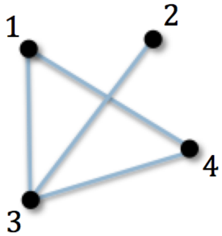
- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Semi-Streaming Model

Introduced by Feigenbaum, Kannan, McGregor, Suri, Zhang '05.

- Space allowance $n \log^c(n)$.
- Receive data via edge updates.
- Minimum spanning tree, maximal matching, graph connectivity, etc.



Overview

Prior Work

- **[Ahn, Guha '09], [Kelner, Levin '11]**: Cut and spectral sparsifiers in *insertion only* streams.
- **[Ahn, Guha, McGregor '12a]**: Introduced linear sketching for graphs. This breakthrough work is the first to handle edge deletions for graph problems. Connectivity, MST, multi-pass sparsifiers.
  - **[Ahn, Guha, McGregor '12b], [Goel, Kapralov, Post '12]**: Extend techniques to get single pass cut sparsifiers.
- **[Ahn, Guha, McGregor '13]**: Dynamic spectral sparsifiers, but $O(n^{5/3})$ space.
- **[Kapralov, Woodruff '14]**: Dynamic spectral sparsifiers, but multi-pass.

Our result: 1-Pass dynamic spectral sparsifiers in $\tilde{O}(n)$ space.

Prior Work

- **[Ahn, Guha '09], [Kelner, Levin '11]**: Cut and spectral sparsifiers in *insertion only* streams.
- **[Ahn, Guha, McGregor '12a]**: Introduced linear sketching for graphs. This breakthrough work is the first to handle edge deletions for graph problems. Connectivity, MST, multi-pass sparsifiers.
  - **[Ahn, Guha, McGregor '12b], [Goel, Kapralov, Post '12]**: Extend techniques to get single pass cut sparsifiers.
- **[Ahn, Guha, McGregor '13]**: Dynamic spectral sparsifiers, but $O(n^{5/3})$ space.
- **[Kapralov, Woodruff '14]**: Dynamic spectral sparsifiers, but multi-pass.

Our result: 1-Pass dynamic spectral sparsifiers in $\tilde{O}(n)$ space.

Prior Work

- **[Ahn, Guha '09], [Kelner, Levin '11]**: Cut and spectral sparsifiers in *insertion only* streams.
- **[Ahn, Guha, McGregor '12a]**: Introduced linear sketching for graphs. This breakthrough work is the first to handle edge deletions for graph problems. Connectivity, MST, multi-pass sparsifiers.
  - **[Ahn, Guha, McGregor '12b], [Goel, Kapralov, Post '12]**: Extend techniques to get single pass cut sparsifiers.
- **[Ahn, Guha, McGregor '13]**: Dynamic spectral sparsifiers, but $O(n^{5/3})$ space.
- **[Kapralov, Woodruff '14]**: Dynamic spectral sparsifiers, but multi-pass.

Our result: 1-Pass dynamic spectral sparsifiers in $\tilde{O}(n)$ space.

Prior Work

- **[Ahn, Guha '09], [Kelner, Levin '11]**: Cut and spectral sparsifiers in *insertion only* streams.
- **[Ahn, Guha, McGregor '12a]**: Introduced linear sketching for graphs. This breakthrough work is the first to handle edge deletions for graph problems. Connectivity, MST, multi-pass sparsifiers.
    - **[Ahn, Guha, McGregor '12b], [Goel, Kapralov, Post '12]**: Extend techniques to get single pass cut sparsifiers.
- **[Ahn, Guha, McGregor '13]**: Dynamic spectral sparsifiers, but $O(n^{5/3})$ space.
- **[Kapralov, Woodruff '14]**: Dynamic spectral sparsifiers, but multi-pass.

Our result: 1-Pass dynamic spectral sparsifiers in $\tilde{O}(n)$ space.

Prior Work

- **[Ahn, Guha '09], [Kelner, Levin '11]**: Cut and spectral sparsifiers in *insertion only* streams.
- **[Ahn, Guha, McGregor '12a]**: Introduced linear sketching for graphs. This breakthrough work is the first to handle edge deletions for graph problems. Connectivity, MST, multi-pass sparsifiers.
  - **[Ahn, Guha, McGregor '12b], [Goel, Kapralov, Post '12]**: Extend techniques to get single pass cut sparsifiers.
- **[Ahn, Guha, McGregor '13]**: Dynamic spectral sparsifiers, but $O(n^{5/3})$ space.
- **[Kapralov, Woodruff '14]**: Dynamic spectral sparsifiers, but multi-pass.

Our result: 1-Pass dynamic spectral sparsifiers in $\tilde{O}(n)$ space.

Prior Work

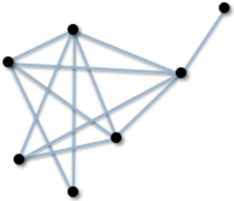
- **[Ahn, Guha '09], [Kelner, Levin '11]**: Cut and spectral sparsifiers in *insertion only* streams.
- **[Ahn, Guha, McGregor '12a]**: Introduced linear sketching for graphs. This breakthrough work is the first to handle edge deletions for graph problems. Connectivity, MST, multi-pass sparsifiers.
    - **[Ahn, Guha, McGregor '12b], [Goel, Kapralov, Post '12]**: Extend techniques to get single pass cut sparsifiers.
- **[Ahn, Guha, McGregor '13]**: Dynamic spectral sparsifiers, but $O(n^{5/3})$ space.
- **[Kapralov, Woodruff '14]**: Dynamic spectral sparsifiers, but multi-pass.

Our result: 1-Pass dynamic spectral sparsifiers in $\tilde{O}(n)$ space.

Overview

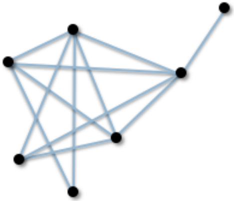
Why is the dynamic case hard?

Graph:

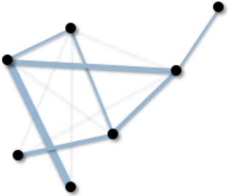


Why is the dynamic case hard?

Graph:

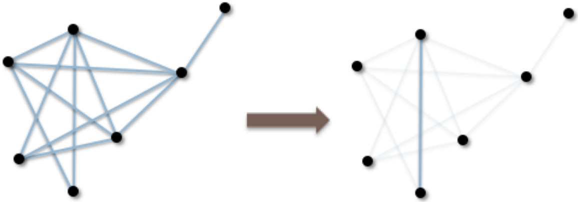


Sketch:

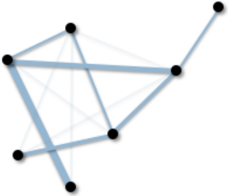


Why is the dynamic case hard?

Graph:

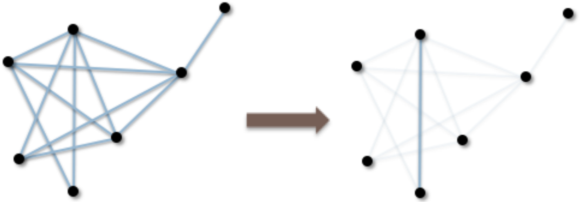


Sketch:

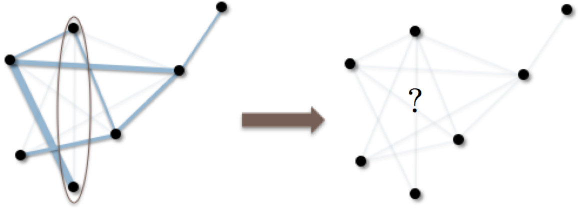


Why is the dynamic case hard?

Graph:



Sketch:



Why is the dynamic case hard?

How do we get around this issue?

Take a cue from standard streaming algorithms:

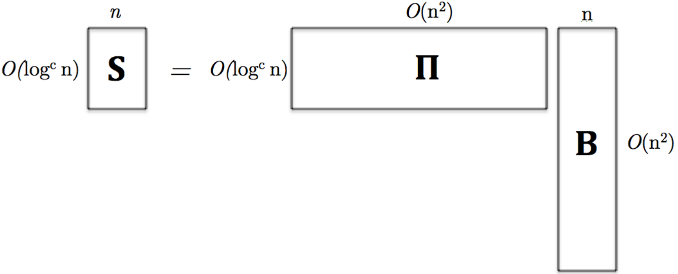
- Linear Sketching!
- Does *not* depend on insertion/deletion order.

Why is the dynamic case hard?

How do we get around this issue?

Take a cue from standard streaming algorithms:

- Linear Sketching!
- Does *not* depend on insertion/deletion order.



Overview

Sampling in the Streaming Model

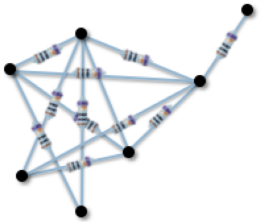
We are still going to sample by effective resistance.

- Treat graph as resistor network, each edge has resistance 1.
- Flow 1 unit of current from node i to j and measure voltage drop between the nodes.

Sampling in the Streaming Model

We are still going to sample by effective resistance.

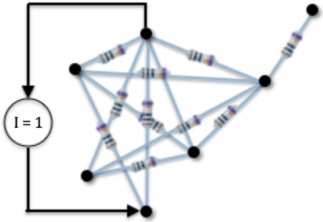
- Treat graph as resistor network, each edge has resistance 1.
- Flow 1 unit of current from node i to j and measure voltage drop between the nodes.



Sampling in the Streaming Model

We are still going to sample by effective resistance.

- Treat graph as resistor network, each edge has resistance 1.
- Flow 1 unit of current from node i to j and measure voltage drop between the nodes.



Sampling in the Streaming Model

Using standard $V = IR$ equations:

$$\boxed{\mathbf{L}} \times \boxed{\text{voltage}} = \boxed{\text{current}}$$

Sampling in the Streaming Model

Using standard $V = IR$ equations:

$$\mathbf{L} \times \text{voltage} = \text{current} \qquad \mathbf{L}^{-1} \times \text{current} = \text{voltage}$$

Sampling in the Streaming Model

Using standard $V = IR$ equations:

$$\mathbf{L} \times \text{voltage} = \text{current} \qquad \mathbf{L}^{-1} \times \text{current} = \text{voltage}$$

If $\mathbf{x}_e = \begin{bmatrix} 1 \\ 0 \\ 0 \\ -1 \\ 0 \end{bmatrix}$, e's effective resistance is $\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e$.

Sampling in the Streaming Model

Effective resistance of edge e is $\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e$.

Alternatively, τ_e is the e^{th} entry in the vector:

$$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$$

AND

$$\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e = \mathbf{x}_e^\top (\mathbf{L}^{-1})^\top \mathbf{B}^\top \mathbf{B} \mathbf{L}^{-1} \mathbf{x}_e = \|\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e\|_2^2$$

Sampling in the Streaming Model

Effective resistance of edge e is $\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e$.

Alternatively, τ_e is the e^{th} entry in the vector:

$$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$$

AND

$$\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e = \mathbf{x}_e^\top (\mathbf{L}^{-1})^\top \mathbf{B}^\top \mathbf{B} \mathbf{L}^{-1} \mathbf{x}_e = \|\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e\|_2^2$$

Sampling in the Streaming Model

Effective resistance of edge e is $\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e$.

Alternatively, τ_e is the e^{th} entry in the vector:

$$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$$

AND

$$\tau_e = \mathbf{x}_e^\top \mathbf{L}^{-1} \mathbf{x}_e = \mathbf{x}_e^\top (\mathbf{L}^{-1})^\top \mathbf{B}^\top \mathbf{B} \mathbf{L}^{-1} \mathbf{x}_e = \|\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e\|_2^2$$

Sampling in the Streaming Model

We just need two more ingredients: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

ℓ_2 **Heavy Hitters [GLPS10]:**

- Sketch vector *poly*(n) vector in *polylog*(n) space.
- Extract any element who's square is a $O(1/\log n)$ fraction of the vector's squared norm.

Coarse Sparsifier:

- $\tilde{\mathbf{L}}$ such that $\mathbf{x}^\top\tilde{\mathbf{L}}\mathbf{x} = (1 \pm \mathit{constant})\mathbf{x}^\top\mathbf{L}\mathbf{x}$

Sampling in the Streaming Model

We just need two more ingredients: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

ℓ_2 **Heavy Hitters [GLPS10]:**

- Sketch vector $poly(n)$ vector in $polylog(n)$ space.
- Extract any element who's square is a $O(1/\log n)$ fraction of the vector's squared norm.

Coarse Sparsifier:

- $\tilde{\mathbf{L}}$ such that $\mathbf{x}^\top \tilde{\mathbf{L}} \mathbf{x} = (1 \pm constant) \mathbf{x}^\top \mathbf{L} \mathbf{x}$

Sampling in the Streaming Model

We just need two more ingredients: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

ℓ_2 **Heavy Hitters [GLPS10]:**

- Sketch vector *poly*(n) vector in *polylog*(n) space.
- Extract any element who's square is a $O(1/\log n)$ fraction of the vector's squared norm.

Coarse Sparsifier:

- $\tilde{\mathbf{L}}$ such that $\mathbf{x}^\top \tilde{\mathbf{L}} \mathbf{x} = (1 \pm \mathit{constant}) \mathbf{x}^\top \mathbf{L} \mathbf{x}$

Sampling in the Streaming Model

Putting it all together: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

1. Sketch $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}$ in $n \log^c n$ space.
2. Compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-1}$.
3. For every possible edge e, compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e$
4. Extract heavy hitters from the vector, check if e^{th} entry is one.

$$\frac{\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e(e)^2}{\|\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e\|_2^2} \approx \frac{\tau_e^2}{\tau_e} = \tau_e$$

So, as long as $\tau_e > O(1/\log n)$, we will recover the edge!

Sampling in the Streaming Model

Putting it all together: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

1. Sketch $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}$ in $n \log^c n$ space.
2. Compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-1}$.
3. For every possible edge e, compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e$
4. Extract heavy hitters from the vector, check if e^{th} entry is one.

$$\frac{\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e(e)^2}{\|\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e\|_2^2} \approx \frac{\tau_e^2}{\tau_e} = \tau_e$$

So, as long as $\tau_e > O(1/\log n)$, we will recover the edge!

Sampling in the Streaming Model

Putting it all together: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

1. Sketch $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}$ in $n \log^c n$ space.
2. Compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-1}$.
3. For every possible edge e, compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e$
4. Extract heavy hitters from the vector, check if e^{th} entry is one.

$$\frac{\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e(e)^2}{\|\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e\|_2^2} \approx \frac{\tau_e^2}{\tau_e} = \tau_e$$

So, as long as $\tau_e > O(1/\log n)$, we will recover the edge!

Sampling in the Streaming Model

Putting it all together: $\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

1. Sketch $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}$ in $n \log^c n$ space.
2. Compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-\mathbf{1}}$.
3. For every possible edge e, compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-\mathbf{1}}\mathbf{x}_e$
4. Extract heavy hitters from the vector, check if e^{th} entry is one.

$$\frac{\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e(e)^2}{\|\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e\|_2^2} \approx \frac{\tau_e^2}{\tau_e} = \tau_e$$

So, as long as $\tau_e > O(1/\log n)$, we will recover the edge!

Sampling in the Streaming Model

Putting it all together: $\mathbf{BL}^{-1}\mathbf{x}_e$

1. Sketch $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}$ in $n \log^c n$ space.
2. Compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-\mathbf{1}}$.
3. For every possible edge e, compute $(\mathbf{\Pi}_{\text{heavy hitters}})\mathbf{B}\tilde{\mathbf{L}}^{-\mathbf{1}}\mathbf{x}_e$
4. Extract heavy hitters from the vector, check if e^{th} entry is one.

$$\frac{\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e(e)^2}{\|\mathbf{B}\tilde{\mathbf{L}}^{-1}\mathbf{x}_e\|_2^2} \approx \frac{\tau_e^2}{\tau_e} = \tau_e$$

So, as long as $\tau_e > O(1/\log n)$, we will recover the edge!

Sampling in the Streaming Model

How about edges with lower effective resistance? Sketch:

$\mathbf{BL}^{-1}\mathbf{x}_e$

Sampling in the Streaming Model

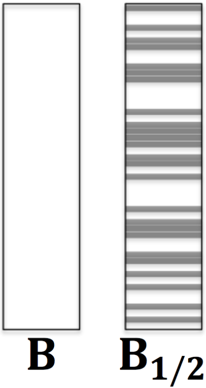
How about edges with lower effective resistance? Sketch:

$\mathbf{B}$

$\mathbf{BL^{-1}x}_e$

Sampling in the Streaming Model

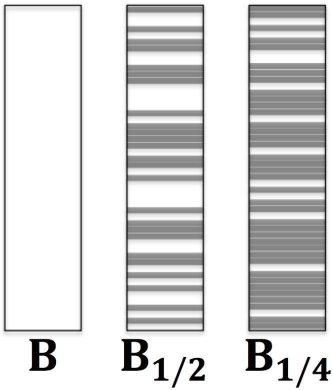
How about edges with lower effective resistance? Sketch:



$\mathbf{BL}^{-1}\mathbf{x}_e$

Sampling in the Streaming Model

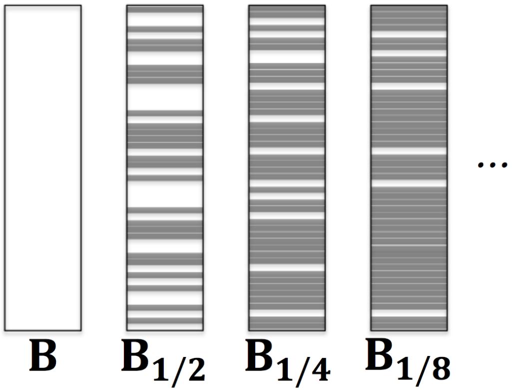
How about edges with lower effective resistance? Sketch:



$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

Sampling in the Streaming Model

How about edges with lower effective resistance? Sketch:



$\mathbf{BL}^{-1}\mathbf{x}_e$

Sampling in the Streaming Model

$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2\log n$ with probability 1/2.
- Third level: $\tau_e > 1/4\log n$ with probability 1/4.
- Forth level: $\tau_e > 1/8\log n$ with probability 1/8.
- ...

So, we can sample every edge by (effective resistance) $\times$ $O(\log n)$.

Sampling in the Streaming Model

$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2\log n$ with probability $1/2$.
- Third level: $\tau_e > 1/4\log n$ with probability $1/4$.
- Forth level: $\tau_e > 1/8\log n$ with probability $1/8$.
- ...

So, we can sample every edge by (effective resistance) $\times$ $O(\log n)$.

Sampling in the Streaming Model

$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2\log n$ with probability 1/2.
- Third level: $\tau_e > 1/4\log n$ with probability 1/4.
- Forth level: $\tau_e > 1/8\log n$ with probability 1/8.
- ...

So, we can sample every edge by (effective resistance) $\times$ $O(\log n)$.

Sampling in the Streaming Model

$\mathbf{BL}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2\log n$ with probability $1/2$.
- Third level: $\tau_e > 1/4\log n$ with probability $1/4$.
- Forth level: $\tau_e > 1/8\log n$ with probability $1/8$.
- ...

So, we can sample every edge by (effective resistance) $\times$ $O(\log n)$.

Sampling in the Streaming Model

$\mathbf{B}\mathbf{L}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2\log n$ with probability $1/2$.
- Third level: $\tau_e > 1/4\log n$ with probability $1/4$.
- Forth level: $\tau_e > 1/8\log n$ with probability $1/8$.
- ...

So, we can sample every edge by (effective resistance) $\times$ $O(\log n)$.

Sampling in the Streaming Model

$\mathbf{BL}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2\log n$ with probability $1/2$.
- Third level: $\tau_e > 1/4\log n$ with probability $1/4$.
- Forth level: $\tau_e > 1/8\log n$ with probability $1/8$.
- ...

So, we can sample every edge by (effective resistance) $\times$ $O(\log n)$.

Sampling in the Streaming Model

$\mathbf{BL}^{-1}\mathbf{x}_e$

How about edges with lower effective resistance?

- First level: $\tau_e > 1/\log n$ with probability 1.
- Second level: $\tau_e > 1/2 \log n$ with probability $1/2$.
- Third level: $\tau_e > 1/4 \log n$ with probability $1/4$.
- Forth level: $\tau_e > 1/8 \log n$ with probability $1/8$.
- ...

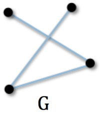
So, we can sample every edge by (effective resistance) $\times\ O(\log n)$.

Overview

Sparsifer Chain

Final Piece [Li, Miller, Peng '12]

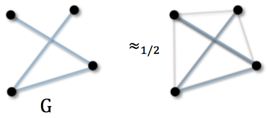
- We needed a constant error spectral sparsifier to get our $(1 \pm \epsilon)$ sparsifier.



Sparsifer Chain

Final Piece [Li, Miller, Peng '12]

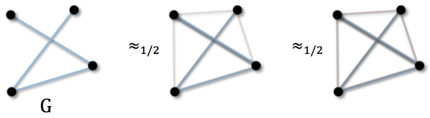
- We needed a constant error spectral sparsifier to get our $(1 \pm \epsilon)$ sparsifier.



Sparsifer Chain

Final Piece [Li, Miller, Peng '12]

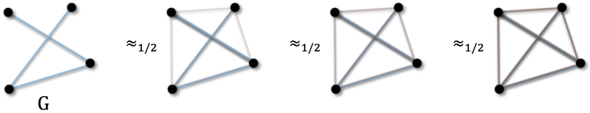
- We needed a constant error spectral sparsifier to get our $(1 \pm \epsilon)$ sparsifier.



Sparsifer Chain

Final Piece [Li, Miller, Peng '12]

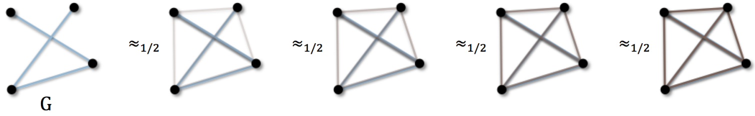
- We needed a constant error spectral sparsifier to get our $(1 \pm \epsilon)$ sparsifier.



Sparsifer Chain

Final Piece [Li, Miller, Peng '12]

- We needed a constant error spectral sparsifier to get our $(1 \pm \epsilon)$ sparsifier.



Conclusion

Final Thoughts:

- Note that everything we did extends unmodified to general matrices $\mathbf{B}$ and general quadratic forms $\mathbf{B}^\top\mathbf{B}$.
  - Just need to ensure that we have a row dictionary and can thus test every possible entry.
- Generically, storing a compression of $\mathbf{B}^\top\mathbf{B}$ takes $\Omega(n^2)$ space. Avoid lower bound simply when the row dictionary is $\mathrm{poly}(n)$ size.

Conclusion

Thank you!