Algorithms, Graph Theory, and Linear Equations in Laplacians

Daniel A. Spielman
Yale University
Shang-Hua Teng
滕尚華
Serge Lang  
1927-2005

Gian-Carlo Rota  
1932-1999
Algorithms, Graph Theory, and Linear Equations in Laplacians

Daniel A. Spielman
Yale University
Solving Linear Equations $Ax = b$, Quickly

Goal: In time linear in the number of non-zeros entries of $A$

Special case: $A$ is the Laplacian Matrix of a Graph
Solving Linear Equations $Ax = b$, Quickly

Goal: In time linear in the number of non-zeros entries of $A$

Special case: $A$ is the Laplacian Matrix of a Graph

Cheeger’s Inequality
Random Walks
Random Matrices
Expanders
Approximations of Graphs
Solving Linear Equations $Ax = b$, Quickly

Goal: In time linear in the number of non-zeros entries of $A$

Special case: $A$ is the Laplacian Matrix of a Graph

1. Why
2. Classic Approaches
3. Recent Developments
4. Connections
Graphs

Set of vertices $V$. Set of edges $E$ of pairs $\{u,v\} \subseteq V$
Graphs

Set of vertices $V$. Set of edges $E$ of pairs $\{u,v\} \subseteq V$
Laplacian Quadratic Form of $G = (V, E)$

For $x : V \rightarrow \mathbb{R}$

$$x^T L_G x = \sum_{(u,v) \in E} (x(u) - x(v))^2$$
Laplacian Quadratic Form of $G = (V,E)$

For $\mathbf{x} : V \rightarrow \mathbb{R}$

$$\mathbf{x}^T L_G \mathbf{x} = \sum_{(u,v) \in E} (\mathbf{x}(u) - \mathbf{x}(v))^2$$
Laplacian Quadratic Form of $G = (V,E)$

For $x : V \rightarrow \mathbb{R}$

$$x^T L_G x = \sum_{(u,v) \in E} (x(u) - x(v))^2$$

$x : \begin{array}{ccc} -3 & -1 & 0 \\ 2^2 & 1^2 & \end{array}$

$$x^T L_G x = 15$$
Laplacian Quadratic Form for Weighted Graph

\[ G = (V, E, w) \]

\[ w : E \rightarrow \mathbb{R}^{+} \] assigns a positive weight to every edge

\[ \mathbf{x}^T L_G \mathbf{x} = \sum_{(u,v) \in E} w(u,v) (\mathbf{x}(u) - \mathbf{x}(v))^2 \]

Matrix \( L_G \) is positive semi-definite nullspace spanned by const vector, if connected
Laplacian Matrix of a Weighted Graph

\[ L_G(u, v) = \begin{cases} 
-w(u, v) & \text{if } (u, v) \in E \\
\sigma(u) & \text{if } u = v \\
0 & \text{otherwise}
\end{cases} \]

\[ d(u) = \sum_{(v, u) \in E} w(u, v) \]

*the weighted degree of* \( u \)
Laplacian Matrix of a Weighted Graph

\[ L_G(u, v) = \begin{cases} 
-w(u, v) & \text{if } (u, v) \in E \\
\delta(u) & \text{if } u = v \\
0 & \text{otherwise}
\end{cases} \]

\[ d(u) = \sum_{(v, u) \in E} w(u, v) \]

*the weighted degree of u*

combinatorial degree is # of attached edges

\[
\begin{bmatrix}
4 & -1 & 0 & -1 & -2 \\
-1 & 4 & -3 & 0 & 0 \\
0 & -3 & 4 & -1 & 0 \\
-1 & 0 & -1 & 2 & 0 \\
-2 & 0 & 0 & 0 & 2
\end{bmatrix}
\]
Networks of Resistors

Ohm’s laws gives \( i = \frac{v}{r} \)

In general, \( i = L_G v \) with \( w(u,v) = \frac{1}{r(u,v)} \)

Minimize dissipated energy \( v^T L_G v \)
Networks of Resistors

Ohm’s laws gives $i = \frac{v}{r}$

In general, $i = L_G v$ with $w_{(u,v)} = \frac{1}{r_{(u,v)}}$

Minimize dissipated energy $v^T L_G v$

By solving Laplacian
Learning on Graphs

Infer values of a function at all vertices from known values at a few vertices.

Minimize \( x^T L_G x = \sum_{(u,v) \in E} w_{(u,v)} (x(u) - x(v))^2 \)

Subject to known values

![Graph Diagram]
Learning on Graphs

Infer values of a function at all vertices from known values at a few vertices.

Minimize \[ x^T L_G x = \sum_{(u,v) \in E} w_{(u,v)} (x(u) - x(v))^2 \]

Subject to known values

By solving Laplacian
Spectral Graph Theory

Combinatorial properties of $G$ are revealed by eigenvalues and eigenvectors of $L_G$.

Compute the most important ones by solving equations in the Laplacian.
Solving Linear Programs in Optimization

Interior Point Methods for Linear Programming:
  network flow problems → Laplacian systems

Numerical solution of Elliptic PDEs

Finite Element Method
How to Solve Linear Equations Quickly

Fast when graph is simple,
   by elimination.

Fast when graph is complicated*,
   by Conjugate Gradient (Hestenes ‘51, Stiefel ‘52)
Cholesky Factorization of Laplacians

When eliminate a vertex, connect its neighbors.

Also known as Y-Δ
Cholesky Factorization of Laplacians

When eliminate a vertex, connect its neighbors.

Also known as Y-Δ
Cholesky Factorization of Laplacians

When eliminate a vertex, connect its neighbors.

Also known as Y-Δ
The order matters

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>-1</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>3</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>-1</td>
<td>2</td>
<td>-1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>-1</td>
<td>0</td>
<td>-1</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>0</th>
<th>0</th>
<th>0</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1.5</td>
<td>-1</td>
<td>-0.5</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>-1.0</td>
<td>2</td>
<td>-1.0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>-0.5</td>
<td>-1</td>
<td>1.5</td>
<td></td>
</tr>
</tbody>
</table>
Complexity of Cholesky Factorization

\[ \#\text{ops} \sim \sum_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree (connected, no cycles)

\[ \#\text{ops} \sim O(|V|) \]
Complexity of Cholesky Factorization

\[ \#ops \sim \Sigma_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree
(connected, no cycles)

\[ \#ops \sim O(|V|) \]
Complexity of Cholesky Factorization

\[ \#\text{ops} \sim \sum_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree

\[ \#\text{ops} \sim O(|V|) \]
Complexity of Cholesky Factorization

\[ \#\text{ops} \sim \sum_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree

\[ \#\text{ops} \sim O(|V|) \]
Complexity of Cholesky Factorization

\[ \#\text{ops} \sim \sum_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree

\[ \#\text{ops} \sim O(|V|) \]
Complexity of Cholesky Factorization

\[ \#\text{ops} \sim \Sigma_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree  

\[ \#\text{ops} \sim O(|V|) \]
Complexity of Cholesky Factorization

#ops \sim \Sigma_v (\text{degree of } v \text{ when eliminate})^2

Tree

\#ops \sim O(|V|)

Planar

\#ops \sim O(|V|^{3/2})

Lipton-Rose-Tarjan ‘79
Complexity of Cholesky Factorization

$\#\text{ops} \sim \Sigma_v \text{(degree of } v \text{ when eliminate)}^2$

Tree

$\#\text{ops} \sim O(|V|)$

Planar

$\#\text{ops} \sim O(|V|^{3/2})$

Lipton-Rose-Tarjan ‘79
Complexity of Cholesky Factorization

\[ \#\text{ops} \sim \sum_v (\text{degree of } v \text{ when eliminate})^2 \]

Tree

\[ \#\text{ops} \sim O(|V|) \]

Planar

\[ \#\text{ops} \sim O(|V|^{3/2}) \]

Lipton-Rose-Tarjan ‘79

Expander like random, but \( O(|V|) \) edges

\[ \#\text{ops} \gtrsim \Omega(|V|^3) \]

Lipton-Rose-Tarjan ‘79
Conductance and Cholesky Factorization

Cholesky slow when conductance high
Cholesky fast when low for \( G \) and all subgraphs

For \( S \subset V \)

\[
\Phi(S) = \frac{\text{# edges leaving } S}{\text{sum degrees on smaller side, } S \text{ or } V - S}
\]

\[
\Phi_G = \min_{S \subset V} \Phi(S)
\]
Conductance

\[ \Phi(S) \overset{\text{def}}{=} \frac{\# \text{ edges leaving } S}{\text{sum of degrees on smaller side}} \]
Conductance

\[ \Phi(S) \overset{\text{def}}{=} \frac{\# \text{ edges leaving } S}{\text{sum of degrees on smaller side}} \]

\[ \Phi(S) = \frac{3}{5} \]
Conductance

\[ \Phi(S) \overset{\text{def}}{=} \frac{\# \text{ edges leaving } S}{\text{sum of degrees on smaller side}} \]

\[ \Phi_G \overset{\text{def}}{=} \min_S \Phi(S) \]

\[ \Phi(S) = \frac{3}{\min(25, 23)} = \Phi_G \]
Cheeger’s Inequality and the Conjugate Gradient

Cheeger’s inequality (degree-\(d\) unwted case)

\[
\frac{1}{2} \frac{\lambda_2}{d} \leq \Phi_G \leq \sqrt{2 \frac{\lambda_2}{d}}
\]

\(\lambda_2 = \text{second-smallest eigenvalue of } L_G\)

~ \(d/\text{mixing time of random walk}\)
Cheeger’s Inequality and the Conjugate Gradient

Cheeger’s inequality (degree-\(d\) unwted case)

\[
\frac{1}{2} \frac{\lambda_2}{d} \leq \Phi_G \leq \sqrt{2} \frac{\lambda_2}{d}
\]

\(\lambda_2 = \) second-smallest eigenvalue of \(L_G\)
\(~ d/\text{mixing time of random walk}~\)

Conjugate Gradient finds \(\epsilon\)-approx solution to \(L_G x = b\)

in \(O(\sqrt{d/\lambda_2 \log \epsilon^{-1}})\) mults by \(L_G\)
in \(O(|E| \sqrt{d/\lambda_2 \log \epsilon^{-1}})\) ops
Fast solution of linear equations

CG fast when conductance high.

Elimination fast when low for G and all subgraphs.
Fast solution of linear equations

CG fast when conductance high.

Planar graphs

Elimination fast when low for G and all subgraphs.

Problems:

Want speed of extremes in the middle
Fast solution of linear equations

CG fast when conductance high.

Elimination fast when low for $G$ and all subgraphs.

Problems:

Want speed of extremes in the middle

Not all graphs fit into these categories!
Preconditioned Conjugate Gradient

Solve $L_G x = b$ by

Approximating $L_G$ by $L_H$ (the preconditioner)

In each iteration

solve a system in $L_H$

multiply a vector by $L_G$

$\epsilon$-approx solution after

$$O\left(\sqrt{\kappa(L_G, L_H)} \log \epsilon^{-1}\right)$$ iterations
Preconditioned Conjugate Gradient

Solve $L_G x = b$ by

Approximating $L_G$ by $L_H$ (the preconditioner)

In each iteration

solve a system in $L_H$

multiply a vector by $L_G$

$\epsilon$-approx solution after

$$O\left(\sqrt{\kappa(L_G, L_H)} \log \epsilon^{-1}\right)$$ iterations

relative condition number
Inequalities and Approximation

\[ L_H \preceq L_G \text{ if } L_G - L_H \text{ is positive semi-definite,} \]
i.e. for all \( x, \)

\[ x^T L_H x \preceq x^T L_G x \]

Example: if \( H \) is a subgraph of \( G \)

\[ x^T L_G x = \sum_{(u,v) \in E} w(u,v) (x(u) - x(v))^2 \]
Inequalities and Approximation

$L_H \preceq L_G$ if $L_G - L_H$ is positive semi-definite, i.e. for all $x$,

$$x^T L_H x \preceq x^T L_G x$$

$$\kappa(L_G, L_H) \leq t$$

if $L_H \preceq L_G \preceq tL_H$

iff $cL_H \preceq L_G \preceq ctL_H$ for some $c$
Inequalities and Approximation

$L_H \preceq L_G$ if $L_G - L_H$ is positive semi-definite, i.e. for all $x$,

$$x^T L_H x \preceq x^T L_G x$$

$$\kappa(L_G, L_H) \leq t$$

if

$$L_H \preceq L_G \preceq tL_H$$

iff

$$cL_H \preceq L_G \preceq ctL_H$$ for some $c$

Call $H$ a $t$-approx of $G$ if $\kappa(L_G, L_H) \leq t$
Other definitions of the condition number
(Goldstine, von Neumann ‘47)

\[ \kappa(L_G, L_H) = \left( \max_{x \in \text{Span}(L_H)} \frac{x^T L_G x}{x^T L_H x} \right) \left( \max_{x \in \text{Span}(L_G)} \frac{x^T L_H x}{x^T L_G x} \right) \]

\[ \kappa(L_G, L_H) = \frac{\lambda_{\text{max}}(L_G L_H^+) \lambda_{\text{min}}(L_G L_H^+)}{\lambda_{\text{min}}(L_G L_H^+)} \]

\(\text{pseudo-inverse}\)

\(\text{min non-zero eigenvalue}\)
Vaidya’s Subgraph Preconditioners

Precondition $G$ by a subgraph $H$

\[ L_H \preceq L_G \] so just need $t$ for which \[ L_G \preceq tL_H \]

Easy to bound $t$ if $H$ is a spanning tree

And, easy to solve equations in $L_H$ by elimination
The Stretch of Spanning Trees

Boman-Hendrickson ‘01: $L_G \leq \text{st}_G(T) L_T$

Where \( \text{st}_T(G) = \sum_{(u,v) \in E} \text{path-length}_T(u,v) \)
The Stretch of Spanning Trees

Boman-Hendrickson ‘01: \( L_G \leq \text{st}_G(T) L_T \)

Where \( \text{st}_T(G) = \sum_{(u,v) \in E} \text{path-length}_T(u, v) \)
The Stretch of Spanning Trees

Boman-Hendrickson ‘01: $L_G \preceq \text{st}_G(T)L_T$

Where $\text{st}_T(G) = \sum_{(u,v) \in E} \text{path-length}_T(u,v)$
The Stretch of Spanning Trees

Boman-Hendrickson ‘01: \( L_G \preceq \text{st}_G(T) L_T \)

Where \( \text{st}_T(G) = \sum_{(u,v) \in E} \text{path-length}_T(u,v) \)
The Stretch of Spanning Trees

Boman-Hendrickson ‘01: $L_G \preceq \text{st}_G(T)L_T$

Where $\text{st}_T(G) = \sum_{(u,v) \in E} \text{path-length}_T(u,v)$
The Stretch of Spanning Trees

Boman-Hendrickson ‘01: $L_G \preceq st_G(T) L_T$

Where $st_T(G) = \sum_{(u,v) \in E} \text{path-length}_T(u, v)$

In weighted case, measure resistances of paths
Low-Stretch Spanning Trees

For every $G$ there is a $T$ with

$$\text{st}_T(G) \leq m^{1+o(1)}$$

where $m = |E|$

(Alon-Karp-Peleg-West ’91)

$$\text{st}_T(G) \leq O(m \log m \log^2 \log m)$$

(Elkin-Emek-S-Teng ’04, Abraham-Bartal-Neiman ’08)

Solve linear systems in time $O(m^{3/2} \log m)$
Low-Stretch Spanning Trees

For every $G$ there is a $T$ with

$$\text{st}_T(G) \leq m^{1+o(1)}$$

where $m = |E|$

(Alon-Karp-Peleg-West ’91)

$$\text{st}_T(G) \leq O(m \log m \log^2 \log m)$$

(Elkin-Emek-S-Teng ’04, Abraham-Bartal-Neiman ’08)

If $G$ is an expander $\text{st}_T(G) \geq \Omega(m \log m)$
Expander Graphs

Infinite family of $d$-regular graphs (all degrees $d$) satisfying $\lambda_2 \geq \text{const} > 0$

Spectrally best are Ramanujan Graphs (Margulis ‘88, Lubotzky-Phillips-Sarnak ‘88) all eigenvalues inside $d \pm 2\sqrt{d - 1}$

Fundamental examples

Amazing properties
ExpANDERS APPROXIMATE COMPLETE GRAPHS

Let $G$ be the complete graph on $n$ vertices (having all possible edges)

All non-zero eigenvalues of $L_G$ are $n$

$$x^T L_G x = n \quad \text{for all} \quad x \perp 1, \|x\| = 1$$
Expanders Approximate Complete Graphs

Let $G$ be the complete graph on $n$ vertices

$$x^T L_G x = n \quad \text{for all} \quad x \perp 1, \|x\| = 1$$

Let $H$ be a $d$-regular Ramanujan Expander

$$(d - 2\sqrt{d - 1}) \leq x^T L_H x \leq (d + 2\sqrt{d - 1})$$

$$\kappa(L_G, L_H) \leq \frac{d + 2\sqrt{d - 1}}{d - 2\sqrt{d - 1}} \rightarrow 1$$
Sparsification

Goal: find sparse approximation for every $G$

S-Teng ‘04: For every $G$ is an $H$ with

$$O(n \log^7 n / \epsilon^2)$$ edges and $\kappa(L_G, L_H) \leq 1 + \epsilon$
Sparsification

S-Teng ‘04: For every $G$ is an $H$ with $O(n \log^{7} n / \epsilon^2)$ edges and $\kappa(L_G, L_H) \leq 1 + \epsilon$

Conductance high

- $\lambda_2$ high (Cheeger)
- random sample good (Füredi-Komlós ‘81)

Conductance not high

- can split graph while removing few edges
Fast Graph Decomposition by local graph clustering

Given vertex of interest find nearby cluster, small $\Phi(S)$, in time $O(|S|)$
Fast Graph Decomposition by local graph clustering

Given vertex of interest find nearby cluster, small $\Phi(S)$, in time $O(|S|)$

S-Teng ’04: Lovász-Simonovits
Andersen-Chung-Lang ‘06: PageRank
Andersen-Peres ‘09: evoloving set Markov chain
Sparsification

Goal: find sparse approximation for every $G$

S-Teng ‘04: For every $G$ is an $H$ with

$$O(n \log^7 n/\epsilon^2)$$

edges and $\kappa(L_G, L_H) \leq 1 + \epsilon$

S-Srivastava ‘08: with $O(n \log n/\epsilon^2)$ edges

proof by modern random matrix theory

Rudelson’s concentration for random sums
Sparsification

Goal: find sparse approximation for every $G$

S-Teng ‘04: For every $G$ is an $H$ with

$$O(n \log^7 n/\epsilon^2)$$

edges and $\kappa(L_G, L_H) \leq 1 + \epsilon$

S-Srivastava ‘08: with $O(n \log n/\epsilon^2)$ edges

Batson-S-Srivastava ‘09

$dn$ edges and $\kappa(L_G, L_H) \leq \frac{d + 1 + 2\sqrt{d}}{d + 1 - 2\sqrt{d}}$
Sparsification by Linear Algebra

edges $\rightarrow$ vectors

Given vectors $v_1, \ldots, v_m \in \mathbb{R}^n$ s.t. $\sum_e v_e v_e^T = I$

Find a small subset $S \subset \{1, \ldots, m\}$ and coefficients $c_e$ s.t.

$$\left\| \sum_{e \in S} c_e v_e v_e^T - I \right\| \leq \epsilon$$
Sparsification by Linear Algebra

Given vectors \( v_1, \ldots, v_m \in \mathbb{R}^n \) s.t. \( \sum_e v_e v_e^T = I \)

Find a small subset \( S \subset \{1, \ldots, m\} \)

and coefficients \( c_e \) s.t. \( \| \sum_{e \in S} c_e v_e v_e^T - I \| \leq \epsilon \)

Rudelson ‘99 says can find \( |S| \leq O(n \log n/\epsilon^2) \)
if choose \( S \) at random*

\[ * = \Pr[e] \sim 1/\|v_e\|^2 \]

In graphs, are effective resistances
Sparsification by Linear Algebra

Given vectors $v_1, \ldots, v_m \in \mathbb{R}^n$ s.t. $\sum_{e} v_{e} v_{e}^T = I$

Find a small subset $S \subset \{1, \ldots, m\}$

and coefficients $c_e$ s.t. $\left\| \sum_{e \in S} c_e v_{e} v_{e}^T - I \right\| \leq \epsilon$

Batson-S-Srivastava: can find $|S| \leq 4n/\epsilon^2$
by greedy algorithm
with Stieltjes potential function
Relation to Kadison-Singer, Paving Conjecture

Would be implied by the following strong version of Weaver’s conjecture KS'₂

Exists constant $\alpha$ s.t. for $v_1, ..., v_m \in \mathbb{R}^n$ s.t. for

$$\|v_e\|^2 = \frac{n}{m} \leq \alpha \quad \sum_{e} v_e v_e^T = I$$

Exists $S \subset \{1, ..., m\}, |S| = m/2$

$$\left\|\sum_{e \in S} v_e v_e^T - \frac{1}{2}I\right\| \leq \frac{1}{4}$$
Relation to Kadison-Singer, Paving Conjecture

Exists constant $\alpha$ s.t. for $v_1, \ldots, v_m \in \mathbb{R}^n$ s.t. for

$$\|v_e\|^2 = \frac{n}{m} \leq \alpha \quad \sum_e v_e v_e^T = I$$

Exists $S \subset \{1, \ldots, m\}, |S| = m/2 \quad \left\| \sum_{e \in S} v_e v_e^T - \frac{1}{2}I \right\| \leq \frac{1}{4}$

Rudelson ’99: $\alpha = \text{const}/(\log n)$

Batson-S-Srivastava ‘09: true, but with coefficients in sum
Relation to Kadison-Singer, Paving Conjecture

Exists constant $\alpha$ s.t. for $v_1, \ldots, v_m \in \mathbb{R}^n$ s.t. for

$$\|v_e\|^2 = \frac{n}{m} \leq \alpha \quad \sum_e v_e v_e^T = I$$

Exists $S \subset \{1, \ldots, m\}, |S| = m/2$

$$\left\| \sum_{e \in S} v_e v_e^T - \frac{1}{2} I \right\| \leq \frac{1}{4}$$

Rudelson ‘99: $\alpha = \text{const}/(\log n)$

Batson-S-Srivastava ‘09: true, but with coefficients in sum

S-Srivastava ‘10: Bourgain-Tzafriri Restricted Invertability
Sparsification and Solving Linear Equations

Can reduce any Laplacian to one with $O(|V|)$ non-zero entries/edges.

S-Teng ‘04: Combine Low-Stretch Trees with Sparsification to solve Laplacian systems in time

$$O(m \log^c n \log \epsilon^{-1})$$

$m = |E| \quad n = |V|$
Sparsification and Solving Linear Equations

S-Teng ‘04: Combine Low-Stretch Trees with Sparsification to solve Laplacian systems in time

\[ O(m \log^c n \log \epsilon^{-1}) \]

\[ m = |E| \quad n = |V| \]

Koutis-Miller-Peng ‘10: time \( O(m \log^2 n \log \epsilon^{-1}) \)

Kolla-Makarychev-Saberi-Teng ‘09:
\( O(m \log n \log \epsilon^{-1}) \) after preprocessing
What’s next

Other families of linear equations:
  from directed graphs
  from physical problems
  from optimization problems

Solving Linear equations as a primitive

Decompositions of the identity:
  Understanding the vectors we get from graphs
  the Ramanujan bound
  Kadison-Singer?
Conclusion

It is all connected