

“Expander” Graphs: aside

$$G = (V, E)$$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

$$\text{Best Sparsity: } 1/d.$$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S |E(S, V - S)| / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$ $d/2$ edges to vertices with more 1's.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$ $d/2$ edges to vertices with more 1's.

Degree d expander graph.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$ $d/2$ edges to vertices with more 1's.

Degree d expander graph.

Sparsity: $\Omega(1)$.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$ $d/2$ edges to vertices with more 1's.

Degree d expander graph.

Sparsity: $\Omega(1)$. Vertex expansion: $\Omega(1)$.

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$ $d/2$ edges to vertices with more 1's.

Degree d expander graph.

Sparsity: $\Omega(1)$. Vertex expansion: $\Omega(1)$. Small sets: $\Omega(1/d)$.

Construction:

“Expander” Graphs: aside

$$G = (V, E)$$

Conductance: $\min_S |E(S, V - S)| / \min(|E(S)|, |E(V - S)|)$.

Edge Expansion: $\min_S E(S, V - S) / \min(|S|, |V - S|)$.

Vertex Expansion: $\min_{S, |S| \leq n/2} |N(S)| / |S|$.

Sparsity (up to scaling): $\min_S |E(S, V - S)| / (|E(S)| \times |E(V - S)|)$

Complete Graph: (uniform degree)

$$\text{Sparsity: } \frac{n/2 \times n/2}{\binom{n}{2}} \approx 1/2.$$

Hypercube:

Best Sparsity: $1/d$.

Dimension cut.

(Vertex expansion: 1, each vertex has partner in other subcube..)

Vertex Expansion: $\Theta(1/\sqrt{d})$. $\binom{d}{d/2} \propto \frac{2^d}{\sqrt{d}}$

Majority cut.

Sparsity: $\Theta(1/\sqrt{d})$ $d/2$ edges to vertices with more 1's.

Degree d expander graph.

Sparsity: $\Omega(1)$. Vertex expansion: $\Omega(1)$. Small sets: $\Omega(1/d)$.

Construction: randomized, or explicit e.g., group theory, zig-zag,...

Gaussian Elimination

$$Ax = b.$$

Gaussian Elimination

$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

Gaussian Elimination

$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

Gaussian Elimination

$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Gaussian Elimination

$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

Gaussian Elimination

$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

$$\vdots$$

Gaussian Elimination

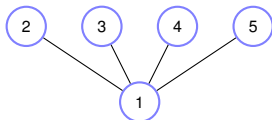
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

Gaussian Elimination

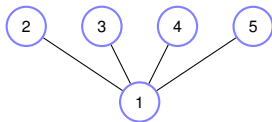
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

Gaussian Elimination

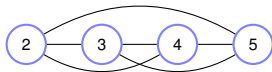
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Gaussian Elimination

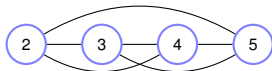
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices,

Gaussian Elimination

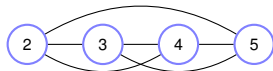
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Gaussian Elimination

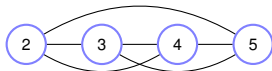
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Really complex.

Gaussian Elimination

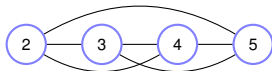
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Really complex.

Path:

Gaussian Elimination

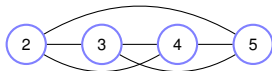
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Really complex.

Path:



Gaussian Elimination

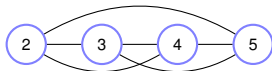
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Really complex.

Path:



Gaussian Elimination

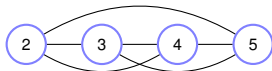
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Really complex.

Path:



Gaussian Elimination

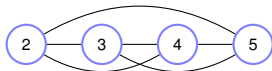
$$Ax = b.$$

$$x_1 = b_1 - a_1 \cdot x$$

$$a'_{2,1} = 0$$

$$\forall j \geq 1, a'_{2,j} = a_{2,j} - \frac{a_{2,1}}{a_{1,1}} a_{1,j} \quad b'_2 = b_2 - \frac{a_{2,1}}{a_{1,1}} b_1$$

⋮



Eliminate variable: graph view.

$$\forall j \geq 1, a'_{2,j} = 0 - \frac{a_{2,1}}{a_{1,1}} a_{1,j}$$

Expander: eliminate any half of vertices, (almost) get complete graph!

Really complex.

Path:



Algorithms.

Iterative good on fast communication systems: lots of short paths.

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

maybe combine iterative $1/\sqrt{n}$ -expanders

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

may combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

may combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Preconditioning:

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

may combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Preconditioning:

Solve $B^{-1}Ax = B^{-1}b$, where B is easy to invert.

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

maybe combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Preconditioning:

Solve $B^{-1}Ax = B^{-1}b$, where B is easy to invert.

And $B^{-1}A$ has good condition number, i.e., λ_1/λ_n .

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

maybe combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Preconditioning:

Solve $B^{-1}Ax = B^{-1}b$, where B is easy to invert.

And $B^{-1}A$ has good condition number, i.e., λ_1/λ_n .

if $B = A$, $B^{-1}A = I$, where condition number is 1.

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

maybe combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Preconditioning:

Solve $B^{-1}Ax = B^{-1}b$, where B is easy to invert.

And $B^{-1}A$ has good condition number, i.e., λ_1/λ_n .

if $B = A$, $B^{-1}A = I$, where condition number is 1.

Vaidya: if B max weight spanning tree, condition number $O(m)$

Algorithms.

Iterative good on fast communication systems: lots of short paths.

Elimination good on slow communication,
not very connected systems.

Nested Dissection:

Planar graphs (elimination) in $O(n^{3/2})$ time:
complete graph only on separators.

Tradeoff in general:

maybe combine iterative $1/\sqrt{n}$ -expanders
and eliminate between them (iteratively)?

Preconditioning:

Solve $B^{-1}Ax = B^{-1}b$, where B is easy to invert.

And $B^{-1}A$ has good condition number, i.e., λ_1/λ_n .

if $B = A$, $B^{-1}A = I$, where condition number is 1.

Vaidya: if B max weight spanning tree, condition number $O(m)$
regardless of weights on edges!

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

- Form B by finding subclusters that are expanders,
- ultra-sparsify using low stretch trees,

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate,

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify,

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify, ...

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify, ...

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify, ...

Kind of a big mess.

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify, ...

Kind of a big mess. Tons of ideas.

Spielman-Teng $\rightarrow \dots \rightarrow$ Kyung-Sachdeva

Solve $B^{-1}Ax = b$, where B is easy to invert.

Spielman-Teng:

Form B by finding subclusters that are expanders,
ultra-sparsify using low stretch trees, eliminate, sparsify, ...

Kind of a big mess. Tons of ideas.

Kyung-Sachdeva

Elimination/sparsify.

Kyung-Sachdeva

Elimination/sparsify.

Iteratively: $Ax_n = b$.

Kyung-Sachdeva

Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Kyung-Sachdeva

Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Kyung-Sachdeva

Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!

Kyung-Sachdeva

Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!

Kyung-Sachdeva

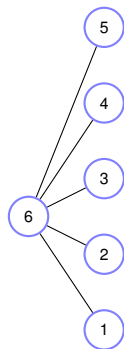
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Kyung-Sachdeva

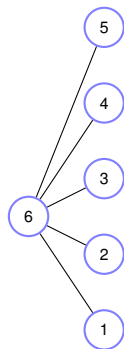
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

Kyung-Sachdeva

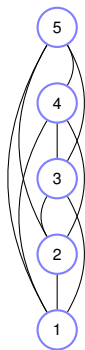
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.
The neighbors would form a complete graph.

Kyung-Sachdeva

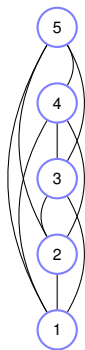
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.
The neighbors would form a complete graph.
Implicitly Sparsify:

Kyung-Sachdeva

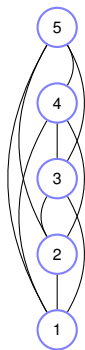
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

Kyung-Sachdeva

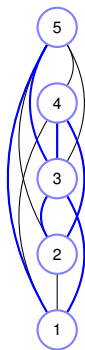
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.
every edge unchanged in expectation.

Kyung-Sachdeva

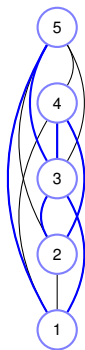
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

Kyung-Sachdeva

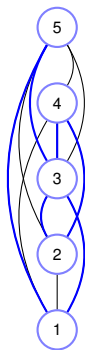
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

(Approximate Cholesky factorization.)

Kyung-Sachdeva

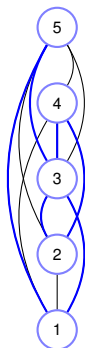
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

(Approximate Cholesky factorization.)

Didn't really solve $x_{n+1} = (A^{-1})(b - Ax_n)$.

Kyung-Sachdeva

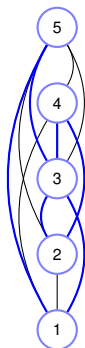
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

(Approximate Cholesky factorization.)

Didn't really solve $x_{n+1} = (A^{-1})(b - Ax_n)$.

Argued elimination process approximates eigenvalues of A^{-1}

Kyung-Sachdeva

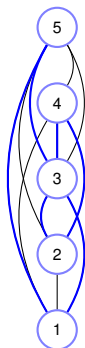
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

(Approximate Cholesky factorization.)

Didn't really solve $x_{n+1} = (A^{-1})(b - Ax_n)$.

Argued elimination process approximates eigenvalues of A^{-1}
Good condition number for $\tilde{A}^{-1}A$.

Kyung-Sachdeva

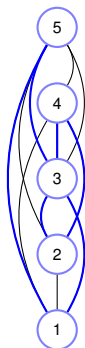
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

(Approximate Cholesky factorization.)

Didn't really solve $x_{n+1} = (A^{-1})(b - Ax_n)$.

Argued elimination process approximates eigenvalues of A^{-1}

Good condition number for $\tilde{A}^{-1}A$.

Note: the \tilde{A}^{-1} is fake notation representing elimination.

Kyung-Sachdeva

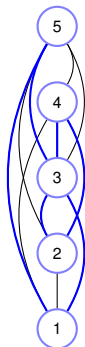
Elimination/sparsify.

Iteratively: $Ax_n = b$.

Find $x_{n+1} = A^{-1}(b - Ax_n)$

Would be done in one iteration!

Circular!



Eliminate a vertex.

The neighbors would form a complete graph.

Implicitly Sparsify:

Select edge with probability p , multiply by $1/p$.

every edge unchanged in expectation.

fewer edges though.

(Approximate Cholesky factorization.)

Didn't really solve $x_{n+1} = (A^{-1})(b - Ax_n)$.

Argued elimination process approximates eigenvalues of A^{-1}

Good condition number for $\tilde{A}^{-1}A$.

Note: the \tilde{A}^{-1} is fake notation representing elimination. No matrix \tilde{A} .

Low stretch trees.

Another cool approach:

Low stretch trees.

Another cool approach:

Low stretch spanning tree.

Low stretch trees.

Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Low stretch trees.

Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander.

Low stretch trees.

Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree.

Low stretch trees.

Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Low stretch trees.

Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.

Low stretch trees.

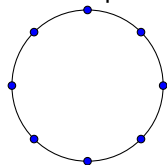
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Low stretch trees.

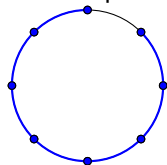
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Low stretch trees.

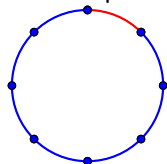
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

Low stretch trees.

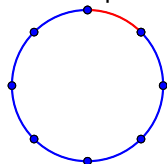
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

Low stretch trees.

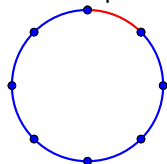
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge

Low stretch trees.

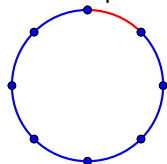
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge get a tree.

Low stretch trees.

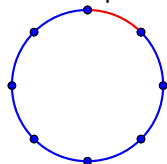
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1$

Low stretch trees.

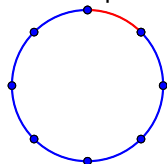
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1)$

Low stretch trees.

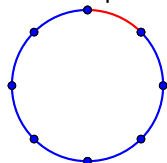
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1) \approx$

Low stretch trees.

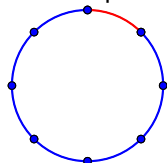
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n-1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1) \approx 2$

Low stretch trees.

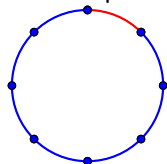
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1) \approx 2$

In general: $\tilde{O}(m)$

Low stretch trees.

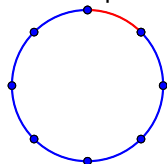
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n - 1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1) \approx 2$

In general: $\tilde{O}(m) = O(m \log n \log \log n)$.

Low stretch trees.

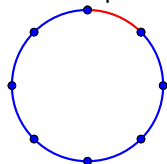
Another cool approach:

Low stretch spanning tree.

Spanning tree T with small $\ell_T(u, v)$ for average edge $e = (u, v)$.

Example: expander. Any short tree. Diameter is $O(\log n)$.

Example: cycle.



Distance 1 goes to $n-1$!

For cycle, remove a random edge get a tree.

Average Stretch of edge: $\frac{n-1}{n} \times 1 + \frac{1}{n} \times (n-1) \approx 2$

In general: $\tilde{O}(m) = O(m \log n \log \log n)$.

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

KOSZ

Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$

KOSZ

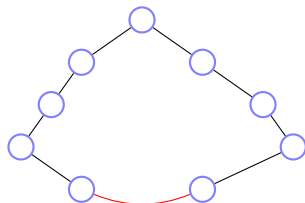
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

KOSZ

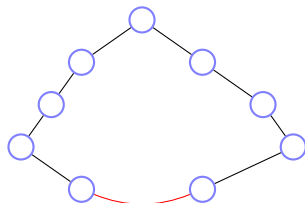
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u, v)} (f(u, v) - \phi(u, v))^2$

KOSZ

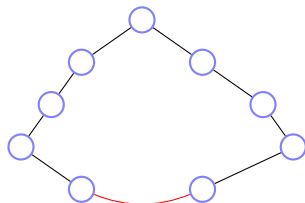
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u, v)} (f(u, v) - \phi(u, v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges)

KOSZ

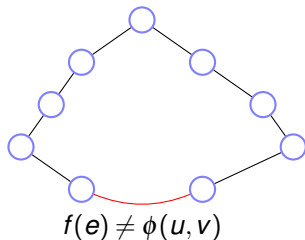
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u, v)} (f(u, v) - \phi(u, v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges
reduces a lot on e since large violation

KOSZ

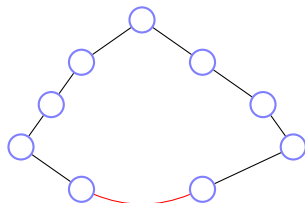
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u, v)} (f(u, v) - \phi(u, v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges
reduces a lot on e since large violation

Decrease on **one edge** dominate increase on **many edges**?

KOSZ

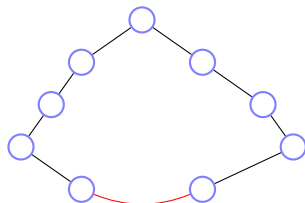
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u, v)} (f(u, v) - \phi(u, v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges
reduces a lot on e since large violation

Decrease on **one edge** dominate increase on **many edges**?

No violation on many, large violation on one.

KOSZ

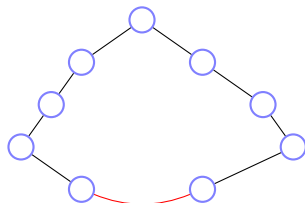
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u,v) - \phi(u,v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u,v)} (f(u,v) - \phi(u,v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges
reduces a lot on e since large violation

Decrease on **one edge** dominate increase on **many edges**?

No violation on many, large violation on one. Quadratic.

KOSZ

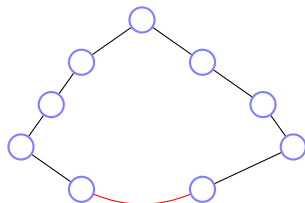
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u,v) - \phi(u,v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u,v)} (f(u,v) - \phi(u,v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges
reduces a lot on e since large violation

Decrease on **one edge** dominate increase on **many edges**?

No violation on many, large violation on one. Quadratic.

$$\text{Increase: } \ell \times (\epsilon^2), \text{ decrease } 1^2 - (1 - \epsilon)^2 = -2\epsilon + \epsilon^2.$$

$$\text{Change: } -2\epsilon + (\ell + 1)\epsilon^2.$$

KOSZ

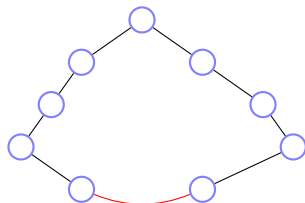
Laplacian systems \equiv to routing electrical flow with sources and sinks.

Solve $\phi(u)$ and currents, $f(\cdot)$, on spanning tree.

$$f(e) = (\phi(u) - \phi(v))/r(e) = \phi(u) - \phi(v) = \phi(u, v).$$

(Assume resistances $r(e) = 1$.)

Pick non-tree edge with probability $\propto \ell_T(e) + 1$



$$f(e) \neq \phi(u, v)$$

Change flow by $\frac{f(u, v) - \phi(u, v)}{\ell_T(e) + 1}$.

Reduces $\sum_{e=(u, v)} (f(u, v) - \phi(u, v))^2$
Increases a bit on $\ell_T(e)$ (no violation) edges
reduces a lot on e since large violation

Decrease on **one edge** dominate increase on **many edges**?

No violation on many, large violation on one. Quadratic.

$$\text{Increase: } \ell \times (\varepsilon^2), \text{ decrease } 1^2 - (1 - \varepsilon)^2 = -2\varepsilon + \varepsilon^2.$$

$$\text{Change: } -2\varepsilon + (\ell + 1)\varepsilon^2. \quad \varepsilon = \frac{1}{\ell + 1} \implies -\varepsilon \text{ decrease.}$$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by $\approx \frac{1}{\ell}(f(e) - \phi(u, v))^2$.**

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell)$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

\implies expected reduction $\propto \sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u, v))^2.$$

The constant of proportionality:

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u, v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1)$.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u, v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u, v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

Probabilities:

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$** .

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

Again: used exact solve on tree to speed up iterative method.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

Again: used exact solve on tree to speed up iterative method.

Shout out to “Algebra”.

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u,v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u,v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u,v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u,v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u,v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u,v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

Again: used exact solve on tree to speed up iterative method.

Shout out to “Algebra”.

Update efficiently?

Analysis Idea

Total violation: $\sum_{e=(u,v)} (f(e) - \phi(u, v))^2$.

Edge w/stretch ℓ selected: **reduce violation by** $\approx \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Non-tree edge reduces $\propto \alpha (f(e) - \phi(u, v))^2$

tree edges increase $\propto \ell (\alpha^2 (f(e) - \phi(u, v))^2)$

$\alpha = \Theta(1/\ell) \implies$ reduction $\propto \frac{1}{\ell} (f(e) - \phi(u, v))^2$.

Selected $\propto \ell$.

$$(f(e) - \phi(u, v))^2 \times \frac{1}{\ell} \times \ell$$

$$\implies \text{expected reduction} \propto \sum_{e=(u,v)} (f(e) - \phi(u, v))^2.$$

The constant of proportionality: $\sum_e (\ell_T(e) + 1) = \tilde{O}(m)$

$$\text{Probabilities: } \approx \frac{\ell_e + 1}{\sum_e (\ell_T(e) + 1)}$$

$(1 - \tilde{\Omega}(1/m))$ expected reduction.

$\tilde{O}(m)$ iterations halves error.

Again: used exact solve on tree to speed up iterative method.

Shout out to “Algebra”.

Update efficiently? Update on paths, path decomposition of trees.

Electrical Flow and Laplacian Systems.

A graph $G = (V, E)$.

Electrical Flow and Laplacian Systems.

A graph $G = (V, E)$.

Circuit: nodes V , resistors E , value 1 (for today.)

Electrical Flow and Laplacian Systems.

A graph $G = (V, E)$.

Circuit: nodes V , resistors E , value 1 (for today.)

Given $\chi : V \rightarrow \mathfrak{R}$

Find flow that routes χ and minimizes

$$\sum_e f(e)^2.$$

Electrical Flow and Laplacian Systems.

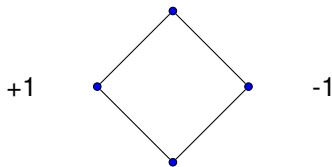
A graph $G = (V, E)$.

Circuit: nodes V , resistors E , value 1 (for today.)

Given $\chi : V \rightarrow \mathfrak{R}$

Find flow that routes χ and minimizes

$$\sum_e f(e)^2.$$



Electrical Flow and Laplacian Systems.

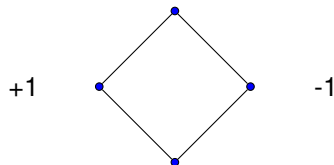
A graph $G = (V, E)$.

Circuit: nodes V , resistors E , value 1 (for today.)

Given $\chi : V \rightarrow \mathfrak{R}$

Find flow that routes χ and minimizes

$$\sum_e f(e)^2.$$



Claim: Minimizer is electrical flow.

Electrical Flow and Laplacian Systems.

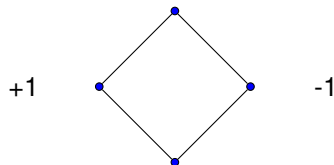
A graph $G = (V, E)$.

Circuit: nodes V , resistors E , value 1 (for today.)

Given $\chi : V \rightarrow \mathfrak{R}$

Find flow that routes χ and minimizes

$$\sum_e f(e)^2.$$



Claim: Minimizer is electrical flow.

Flow corresponds to flow induced by a set of potentials.

Some Matrices.

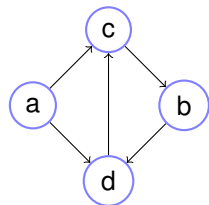
Given $G = (V, E)$, arbitrarily orient edges.

Some Matrices.

Given $G = (V, E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e = (u, v) \\ 1 & e = (v, u) \\ 0 & \text{otherwise} \end{cases}$$

$$L_{u,v} = \begin{cases} d & u = v \\ -1 & (u, v) \in E \\ 0 & \text{otherwise} \end{cases}$$



B

	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

L

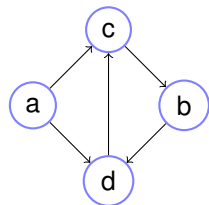
	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

Some Matrices.

Given $G = (V, E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e = (u, v) \\ 1 & e = (v, u) \\ 0 & \text{otherwise} \end{cases}$$

$$L_{u,v} = \begin{cases} d & u = v \\ -1 & (u, v) \in E \\ 0 & \text{otherwise} \end{cases}$$



B

	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

L

	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

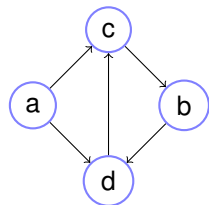
Fun facts: $\mathbf{f} \in \mathfrak{R}^{|E|}$

Some Matrices.

Given $G = (V, E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e = (u, v) \\ 1 & e = (v, u) \\ 0 & \text{otherwise} \end{cases}$$

$$L_{u,v} = \begin{cases} d & u = v \\ -1 & (u, v) \in E \\ 0 & \text{otherwise} \end{cases}$$



B

	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

L

	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

Fun facts: $\mathbf{f} \in \mathfrak{R}^{|E|}$

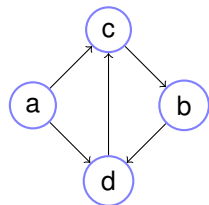
$$[B^T \mathbf{f}]_u = \sum_{e=(u,v)} f_e - \sum_{e=(v,u)} f_e$$

Some Matrices.

Given $G = (V, E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e = (u, v) \\ 1 & e = (v, u) \\ 0 & \text{otherwise} \end{cases}$$

$$L_{u,v} = \begin{cases} d & u = v \\ -1 & (u, v) \in E \\ 0 & \text{otherwise} \end{cases}$$



	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

Fun facts: $\mathbf{f} \in \mathfrak{R}^{|E|}$

$$[B^T \mathbf{f}]_u = \sum_{e=(u,v)} f_e - \sum_{e=(v,u)} f_e$$

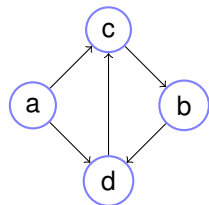
$$B^T B = L$$

Some Matrices.

Given $G = (V, E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e = (u, v) \\ 1 & e = (v, u) \\ 0 & \text{otherwise} \end{cases}$$

$$L_{u,v} = \begin{cases} d & u = v \\ -1 & (u, v) \in E \\ 0 & \text{otherwise} \end{cases}$$



B

	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

L

	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

Fun facts: $\mathbf{f} \in \mathfrak{R}^{|E|}$

$$[B^T \mathbf{f}]_u = \sum_{e=(u,v)} f_e - \sum_{e=(v,u)} f_e$$

$$B^T B = L$$

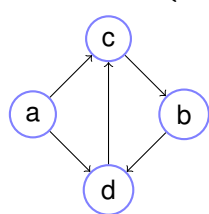
$$[B\mathbf{x}]_{e=(u,v)} = x_u - x_v$$

Some Matrices.

Given $G = (V, E)$, arbitrarily orient edges.

$$B_{v,e} = \begin{cases} -1 & e = (u, v) \\ 1 & e = (v, u) \\ 0 & \text{otherwise} \end{cases}$$

$$L_{u,v} = \begin{cases} d & u = v \\ -1 & (u, v) \in E \\ 0 & \text{otherwise} \end{cases}$$



B

	a	b	c	d
(a,b)	1	-1	0	0
(a,c)	1	0	-1	0
(c,d)	0	0	1	-1
(d,b)	0	-1	0	1
(b,c)	0	1	-1	1

L

	a	b	c	d
a	2	-1	-1	0
b	-1	2	0	-1
c	-1	-1	3	-1
d	-1	-1	-1	3

Fun facts: $\mathbf{f} \in \mathfrak{R}^{|E|}$

$$[B^T \mathbf{f}]_u = \sum_{e=(u,v)} f_e - \sum_{e=(v,u)} f_e$$

$$B^T B = L$$

$$[B\mathbf{x}]_{e=(u,v)} = x_u - x_v$$

$$\mathbf{x}^T L \mathbf{x} = \sum_{e=(u,v)} (x_u - x_v)^2$$

Duality..

Given $G, \chi, \chi \perp 1$

Duality..

Given $G, \chi, \chi \perp 1$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian:

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual:

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes**

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$?

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$$2f(e) + 2(\phi_v - \phi_u) = 0$$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem:

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem: Find ϕ that maximizes ...

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem: Find ϕ that maximizes ...

$\max L(\phi, B\phi) = \phi^T B^T B\phi + 2\phi^T(\chi - B^T B\phi)$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem: Find ϕ that maximizes ...

$$\max L(\phi, B\phi) = \phi^T B^T B\phi + 2\phi^T(\chi - B^T B\phi) \implies \max_{\phi} 2\phi^T \chi - \phi^T L\phi$$

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that **maximizes** $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem: Find ϕ that maximizes ...

$\max L(\phi, B\phi) = \phi^T B^T B\phi + 2\phi^T(\chi - B^T B\phi) \implies \max_{\phi} 2\phi^T \chi - \phi^T L\phi$

Note: want $\phi^T L\phi = \sum_e (\phi_u - \phi_v)^2$ to be small.

Duality..

Given $G, \chi, \chi \perp 1$

Minimize $|f|^2$ subject to $B^T f = \chi$.

Lagrangian: $L(\phi, f) = \sum_e f(e)^2 + 2\phi^T(\chi - B^T f)$

Lagrangian Dual: Find ϕ that maximizes $\min_f L(\phi, f)$.

Given ϕ , minimize $L(\phi, f)$? Calculus.

For $e = (u, v)$

$2f(e) + 2(\phi_v - \phi_u) = 0$ (Minimum when partial derivatives = 0.)

$\rightarrow f(e) = (\phi_u - \phi_v)$ Potential differences!!!

Matrix Form: $f = B\phi$ Again, flows should be potential differences.

Dual problem: Find ϕ that maximizes ...

$\max_{\phi} L(\phi, B\phi) = \phi^T B^T B\phi + 2\phi^T(\chi - B^T B\phi) \implies \max_{\phi} 2\phi^T \chi - \phi^T L\phi$

Note: want $\phi^T L\phi = \sum_e (\phi_u - \phi_v)^2$ to be small.

Minimize Squared Potential differences!

Why did we take dual?

Dual problem:

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Also useful for convergence.

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Also useful for convergence.

Algorithm maintains feasible ϕ, f ,

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Also useful for convergence.

Algorithm maintains feasible ϕ, f ,

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi^T L\phi$

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Also useful for convergence.

Algorithm maintains feasible ϕ, f ,

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi^T L\phi$

Duality gap is “distance” from optimal!

Why did we take dual?

Dual problem:

Find ϕ that maximizes ...

$$\max_{\phi} 2\phi\chi - \phi L\phi$$

Take the derivative:

$$L\phi - \chi$$

$L\phi = \chi$ at optimal point!

Optimal potential is solution to a Laplacian linear system.

Also useful for convergence.

Algorithm maintains feasible ϕ, f ,

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi^T L\phi$

Duality gap is “distance” from optimal!

Algorithm: Work on flow and potentials.

To drive gap to 0.

Alg.

Given: χ, G

Alg.

Given: χ, G

Take a spanning tree T of G .

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$!

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$!!

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$!!!

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$! ! ! !

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$! ! ! ! !

Alg.

Given: χ, G

Take a spanning tree T of G . (Which tree?)

Route flow, f , to satisfy χ through T

Compute, ϕ , using tree ; $\phi_s = 0$, add f_e through T

Repeat:

Choose non-tree edge $e = (u, v)$ (Which non-tree edge?)

$$f(e) = (\phi_u - \phi_v) / (\ell_T(u, v) + 1)$$

($\ell_T(u, v)$ path length in T)

Route excess on path through tree.

Which Tree?

Claim: Linear time algorithm for T w/ stretch $O(m \log n \log \log n)$!

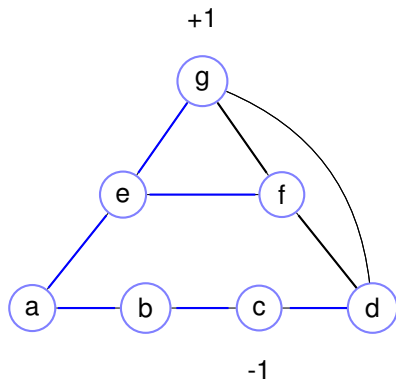
$$\text{Stretch: } \sum_{e=(u,v)} \ell_T(u, v)$$

Which non-tree edge?

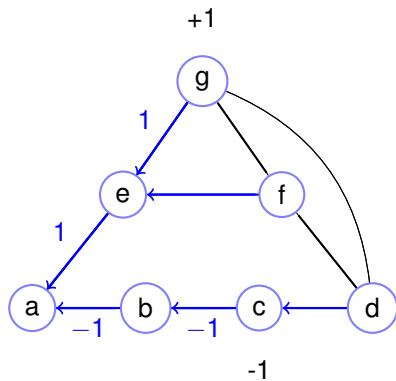
Choose an edge w/prob. proportional to $\ell_T(e)$.

Finds $(1 + \varepsilon)$ approximation in $O(m \log n \log \log n \log(\frac{n}{\varepsilon}))$! ! ! ! !

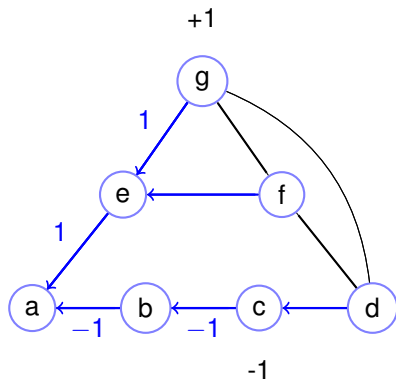
Animation



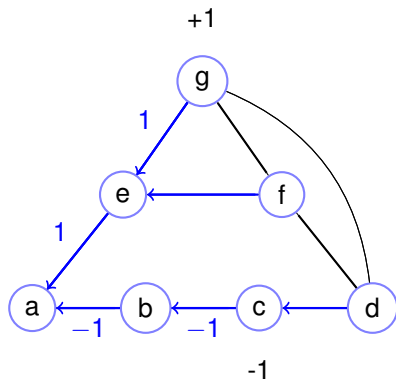
Animation



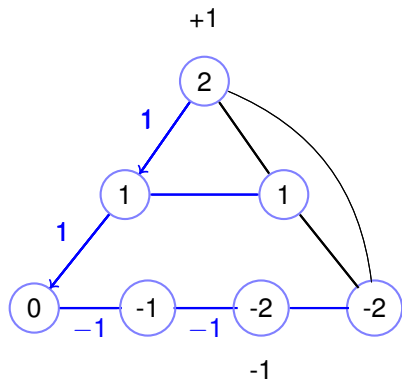
Animation



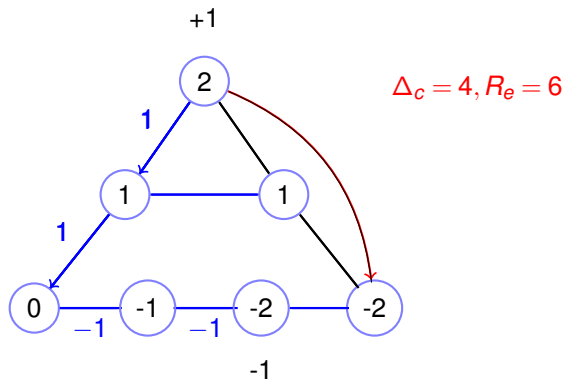
Animation



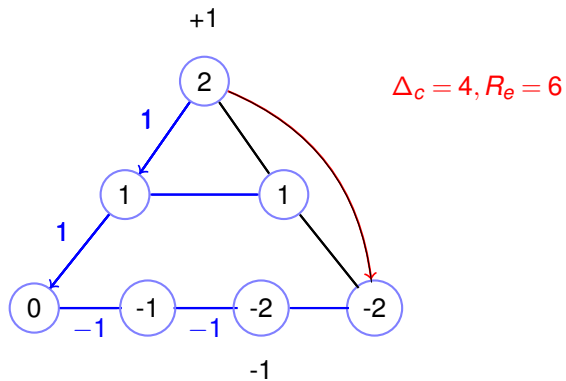
Animation



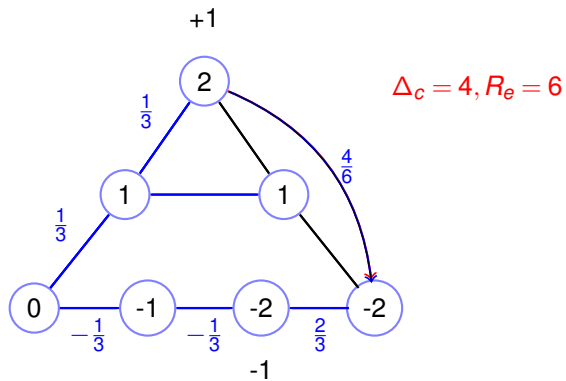
Animation



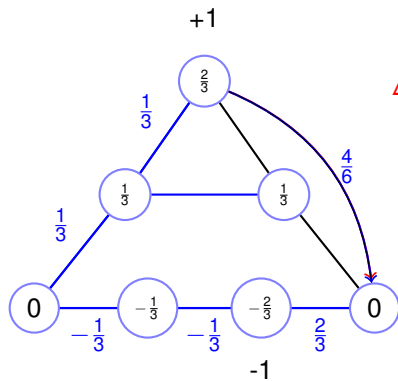
Animation



Animation



Animation



$$\Delta_c = 4, R_e = 6$$

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2$$

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 = \sum_{e' \in C_e} -2f(e')\delta + \delta^2$$

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\begin{aligned} \sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 &= \sum_{e' \in C_e} -2f(e')\delta + \delta^2 \\ &= -(2\delta \sum_{e' \in C_e} f(e')) + R_E \delta^2 \end{aligned}$$

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\begin{aligned} \sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 &= \sum_{e' \in C_e} -2f(e')\delta + \delta^2 \\ &= -(2\delta \sum_{e' \in C_e} f(e')) + R_e \delta^2 \end{aligned}$$

Note: $\sum_{e' \in C_e} f(e') = R_e \delta$

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly “Fix” edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\begin{aligned} \sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 &= \sum_{e' \in C_e} -2f(e')\delta + \delta^2 \\ &= -(2\delta \sum_{e' \in C_e} f(e')) + R_e \delta^2 \end{aligned}$$

Note: $\sum_{e' \in C_e} f(e') = R_e \delta$

$\rightarrow -\Delta_{C_e}^2 / R_e$ where $\Delta_{C_e} = \sum_{e' \in C_e} f(e')$.

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly "Fix" edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\begin{aligned} \sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 &= \sum_{e' \in C_e} -2f(e')\delta + \delta^2 \\ &= -(2\delta \sum_{e' \in C_e} f(e')) + R_e \delta^2 \end{aligned}$$

Note: $\sum_{e' \in C_e} f(e') = R_e \delta$

$\rightarrow -\Delta_{C_e}^2 / R_e$ where $\Delta_{C_e} = \sum_{e' \in C_e} f(e')$.

Fix a part of the potential difference, Δ_{C_e} around cycle!!

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly "Fix" edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\begin{aligned} \sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 &= \sum_{e' \in C_e} -2f(e')\delta + \delta^2 \\ &= -(2\delta \sum_{e' \in C_e} f(e')) + R_e \delta^2 \end{aligned}$$

Note: $\sum_{e' \in C_e} f(e') = R_e \delta$

$\rightarrow -\Delta_{C_e}^2 / R_e$ where $\Delta_{C_e} = \sum_{e' \in C_e} f(e')$.

Fix a part of the potential difference, Δ_{C_e} around cycle!!

\rightarrow reduction of $\Delta_{C_e}^2 / R_e$ in energy!

Energy reduction.

Given $T, e = (u, v)$, let $R_e = \ell_T(u, v) + 1$.

Algorithm:

Repeatedly "Fix" edge $e = (u, v)$.

Route $-\delta = -\frac{\sum_{e' \in C_e} f(e')}{R_e}$ flow around cycle induced in T : C_e
(assume e' are oriented around cycle.)

Difference in energy from f and f' .

$$\begin{aligned} \sum_{e' \in C_e} (f(e') - \delta)^2 - (f(e'))^2 &= \sum_{e' \in C_e} -2f(e')\delta + \delta^2 \\ &= -(2\delta \sum_{e' \in C_e} f(e')) + R_e \delta^2 \end{aligned}$$

Note: $\sum_{e' \in C_e} f(e') = R_e \delta$

$\rightarrow -\Delta_{C_e}^2 / R_e$ where $\Delta_{C_e} = \sum_{e' \in C_e} f(e')$.

Fix a part of the potential difference, Δ_{C_e} around cycle!!

\rightarrow reduction of $\Delta_{C_e}^2 / R_e$ in energy!

Fix $1/R_e$ of a cycle violation!

Duality Gap?

Algorithm maintains feasible $\phi, f, (B^T f = \chi)$

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

$$\text{Gap: } |f|^2 - (2\phi^T \chi - \phi^T L\phi).$$

$$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$$

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi^T \chi - \phi^T L \phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L \phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B \phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f .

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f .

$\Delta_\phi(u, v) = \sum_{e \in P_{u,v}} -f(e)$. **assume $f(e)$ is oriented around cycle.**

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f .

$\Delta_\phi(u, v) = \sum_{e \in P_{u,v}} -f(e)$. **assume $f(e)$ is oriented around cycle.**

For $e \in T$, $\Delta_\phi(e) = f(e)$.

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f .

$\Delta_\phi(u, v) = \sum_{e \in P_{u,v}} -f(e)$. **assume $f(e)$ is oriented around cycle.**

For $e \in T$, $\Delta_\phi(e) = f(e)$. For $e \notin T$ $\Delta_{C_e} = f(e) + \sum_{e \in P_e} f(e)$

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f .

$\Delta_\phi(u, v) = \sum_{e \in P_{u,v}} -f(e)$. **assume $f(e)$ is oriented around cycle.**

For $e \in T$, $\Delta_\phi(e) = f(e)$. For $e \notin T$ $\Delta_{C_e} = f(e) + \sum_{e \in P_e} f(e)$

Duality Gap: $\sum_{e \notin T} (\Delta_{C_e}(f))^2$

Duality Gap?

Algorithm maintains feasible ϕ, f , ($B^T f = \chi$)

Primal value: $|f|^2$.

Dual value: $2\phi\chi - \phi L\phi$

ϕ is tree induced voltages.

Total Duality Gap?

Gap: $|f|^2 - (2\phi^T \chi - \phi^T L\phi)$.

$= |f|^2 - 2\phi^T B^T f + \phi^T B^T B\phi$ where $B^T f = \chi$ and $L = B^T B$.

$= (f - B\phi)^T (f - B\phi)$.

Gap = $\sum_e (f(e) - \Delta_\phi(e))^2$ Difference between ϕ flow and f .

$\Delta_\phi(u, v) = \sum_{e \in P_{u,v}} -f(e)$. **assume $f(e)$ is oriented around cycle.**

For $e \in T$, $\Delta_\phi(e) = f(e)$. For $e \notin T$ $\Delta_{C_e} = f(e) + \sum_{e \in P_e} f(e)$

Duality Gap: $\sum_{e \notin T} (\Delta_{C_e}(f))^2$

Total distance from optimal is cycle violations!

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e}$

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$



Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation. □

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time!

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time! !

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time!!!

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time! !!!

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

$\rightarrow \tilde{O}(m)$ time! ! ! ! !

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

→ $\tilde{O}(m)$ time! ! ! ! ! !

Enough with the exclamations already

Polishing off.

Claim: $E[\text{change in energy} | \text{Gap}] = \frac{\text{Gap}}{\tau}$
($\tau = \sum_e \ell_T(e)$ is stretch of E in T .)

Duality Gap: $\sum_{e \notin T} \Delta_{C_e}(f)^2$

Choose edge e reduce energy by $-\frac{\Delta_{C_e}^2}{R_e}$.

Choose edge with probability $\frac{R_e}{\tau}$.

Expected reduction $-\sum_e \frac{R_e}{\tau} \frac{\Delta_{C_e}^2}{R_e} = -\sum_e \frac{\Delta_{C_e}^2}{\tau}$ □

Duality Gap reduces by $(1 - 1/\tau)$ every iteration on expectation.

$O(\tau \log(n/\epsilon))$ iterations gives $(1 + \epsilon)$ approximation.

$\tau = O(m \log n \log \log n) \dots$

$\tilde{O}(m)$ iterations

Iteration in $O(\log^2 n)$ time using balanced binary trees.

→ $\tilde{O}(m)$ time ! ! ! ! !

Enough with the exclamations already !

Wrapup...

How to get low stretch tree?

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ...,

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch?

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)

Ideas: Similar to HST construction, but much more subtle.

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)

Ideas: Similar to HST construction, but much more subtle.

How to do update along cycle?

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)

Ideas: Similar to HST construction, but much more subtle.

How to do update along cycle?

Answer: Data Structures.

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)

Ideas: Similar to HST construction, but much more subtle.

How to do update along cycle?

Answer: Data Structures.

Idea: Use a binary tree on paths.

Wrapup...

How to get low stretch tree?

Answer: Elkin-Spielman-Teng, ..., Abraham, Newman.

Open: Get $O(m \log n)$ stretch? Like for HSTs. (Later)

Ideas: Similar to HST construction, but much more subtle.

How to do update along cycle?

Answer: Data Structures.

Idea: Use a binary tree on paths.

Decompose tree into paths.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

Init: $f(e) = 0, \forall e \in G$.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

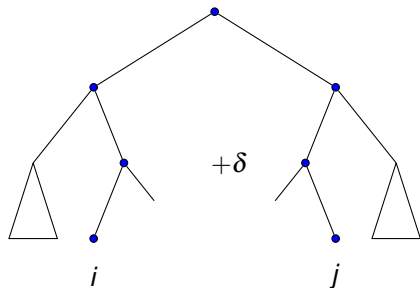
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

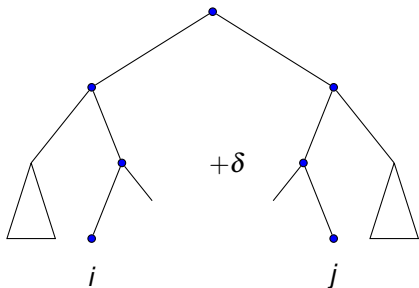
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.
For left path:

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

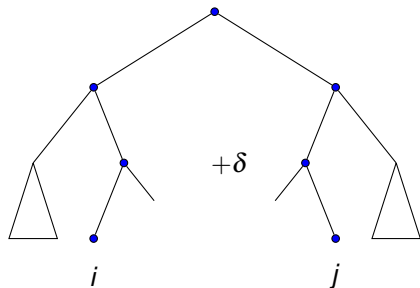
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

For left path:

Right children, add δ to edge.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

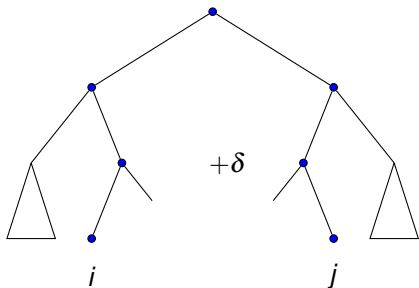
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

For left path:

Right children, add δ to edge.

Reflect for right path.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

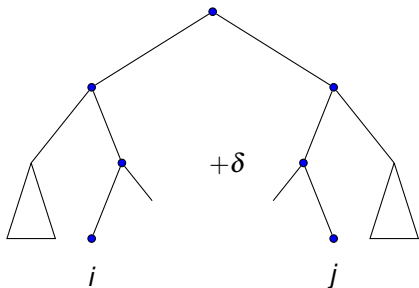
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

For left path:

Right children, add δ to edge.

Reflect for right path.

Lookup: Exercise.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

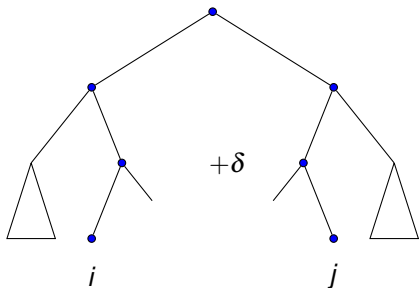
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

For left path:

Right children, add δ to edge.

Reflect for right path.

Lookup: Exercise.

Also, need number of descendants.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

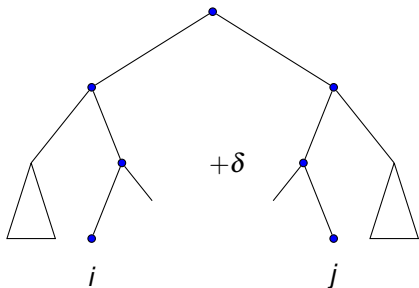
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

For left path:

Right children, add δ to edge.

Reflect for right path.

Lookup: Exercise.

Also, need number of descendants.

Interval Tree

Path: $G = (V, E)$, $V = \{1, \dots, n\}$, $E = \{(i, i+1) : i \in \{1, n-1\}\}$

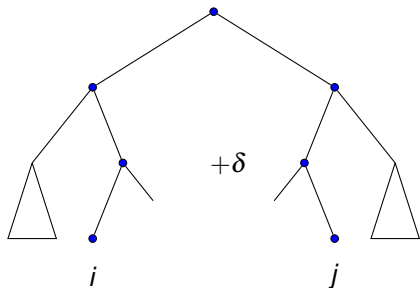
Init: $f(e) = 0, \forall e \in G$.

Update(i, j, δ)

$\forall k \in \{i, \dots, j-1\}, f(i, i+1) = f(i, i+1) + \delta$

Lookup(i, j)

Return $\sum_{k=i}^{j-1} f(k, k+1)$.



Update: find common ancestor.

For left path:

Right children, add δ to edge.

Reflect for right path.

Lookup: Exercise.

Also, need number of descendants.

$O(\log n)$ update/lookup.

Path Decomposition of Tree

Given tree T .

Path Decomposition of Tree

Given tree T .

Decomposition Procedure:

Path Decomposition of Tree

Given tree T .

Decomposition Procedure:
Longest path from root to leaf.

Path Decomposition of Tree

Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Path Decomposition of Tree

Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove.

Path Decomposition of Tree

Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse

Path Decomposition of Tree

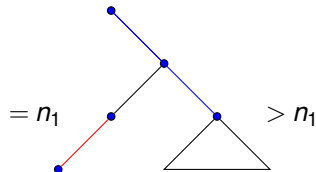
Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse



Path Decomposition of Tree

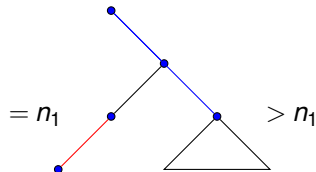
Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse



Decomposition Property:

Path Decomposition of Tree

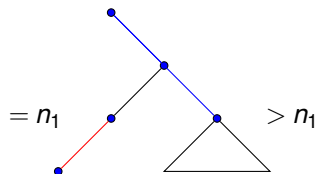
Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse



Decomposition Property:

Where a root to leaf path in tree only sees $O(\log n)$ paths.

Path Decomposition of Tree

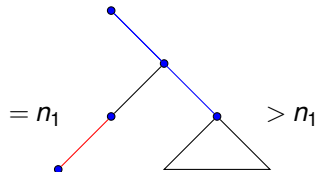
Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse



Decomposition Property:

Where a root to leaf path in tree only sees $O(\log n)$ paths.

Proof Idea:

Path Decomposition of Tree

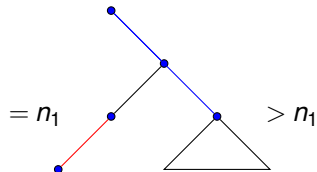
Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse



Decomposition Property:

Where a root to leaf path in tree only sees $O(\log n)$ paths.

Proof Idea:

Every path change, doubles number of vertices.

Path Decomposition of Tree

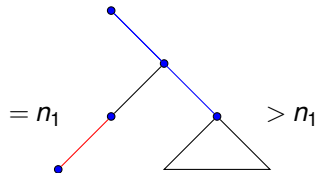
Given tree T .

Decomposition Procedure:

Longest path from root to leaf.

From root, go towards heavier side.

Remove. Recurse



Decomposition Property:

Where a root to leaf path in tree only sees $O(\log n)$ paths.

Proof Idea:

Every path change, doubles number of vertices.

$O(\log^2 n)$ update time for updating flow values on tree edges!

What's going on?

Geometric View.

What's going on?

Geometric View.

Cycles are constraints.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

Better Algorithm:

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

Better Algorithm:

Recursive algorithm give $O(m\sqrt{\log n})$ iterations to halve error.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection
into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

Better Algorithm:

Recursive algorithm give $O(m\sqrt{\log n})$ iterations to halve error.

Correspondence to Practice: Random sparsification of Cholesky factorization.

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

Better Algorithm:

Recursive algorithm give $O(m\sqrt{\log n})$ iterations to halve error.

Correspondence to Practice: Random sparsification of Cholesky factorization.

Laplacian Systems are quite general:

What's going on?

Geometric View.

Cycles are constraints.

Flow around cycle = 0.

Each cycle update is approximate projection into subspace defined by constraint.

The solution is intersection of all cycle constraints and flow conservation.

Algorithmic Power. Algebra: Solving exactly on tree.

Calculus: make a local move.

Better Algorithm:

Recursive algorithm give $O(m\sqrt{\log n})$ iterations to halve error.

Correspondence to Practice: Random sparsification of Cholesky factorization.

Laplacian Systems are quite general:

Climate, physics, SDD-matrices.