

MATRIX MULTIPLICATIVE UPDATES
AND
FAST GRAPH ALGORITHMS

Lorenzo Orecchia – UC Berkeley

Motivation and Goals

- Massive graphs in many applications
- New efficiency standard: nearly-linear time algorithms
- Heuristics vs algorithms

GOALS:

- 1) Develop fast algorithms with strong theoretical guarantees
- 2) Impact on real-world algorithms: natural, tested

Graph Problems

- Multi-commodity Flow
- Single-commodity Flow
- Bipartite matching

- Solution of SDD systems of linear equations
- Uniform sampling of spanning trees
- Sparsification

- Maxcut Approximation
- Graph Partitioning Approximation

Graph Problems

- Multi-commodity Flow
- Single-commodity Flow
- Bipartite matching

- Solution of SDD systems of linear equations
- Uniform sampling of spanning trees
- Sparsification

- Maxcut Approximation
- Graph Partitioning Approximation

NEARY-LINEAR TIME

Graph Problems

- Multi-commodity Flow
- Single-commodity Flow
- Bipartite matching

- Solution of SDD systems of linear equations
- Uniform sampling of spanning trees
- **Sparsification**

- Maxcut Approximation
- **Graph Partitioning Approximation**

NEARY-LINEAR TIME

Design of Fast Algorithms

SPECTRAL METHODS

- Sparsification
- SDD systems
- Spanning tree sampling

MULTIPLICATIVE WEIGHT UPDATE (MWU)

- Flow problems
- Maxcut
- Graph Partitioning

Design of Fast Algorithms

SPECTRAL METHODS

- Sparsification
- SDD systems
- Spanning tree sampling

MULTIPLICATIVE WEIGHT UPDATE (MWU)

- Flow problems
- Maxcut
- Graph Partitioning

OTHER TECHNIQUES: Preconditioning, Embeddings into Trees, Oblivious Routing, Cut Sparsification, ...

Design of Fast Algorithms

SPECTRAL METHODS

- Sparsification
- SDD systems
- Spanning tree sampling

MULTIPLICATIVE WEIGHT UPDATE (MWU)

- Flow problems
- Maxcut
- Graph Partitioning

COMBINING ...

- Faster Approximations for Graph Partitioning [Madry'10]
- Faster Approximate s-t Maxflow [Christiano et al.'10]

Outline

- Multiplicative Weight Updates (MWUs)
 - In Online Learning and in the solution of Linear Programs (LPs)
 - Matrix MWUs
 - Matrix MWUs and solving Semi-definite Programs (SDPs)
 - Applications
- Spectral Methods and Matrix MWUs
 - Spectral Sparsification by Sampling
 - Nearly-linear time Balanced Cut
- Future Directions

Multiplicative Weight Updates

experts e_1 e_2 e_3 e_4 \dots e_n

Multiplicative Weight Updates

experts e_1 e_2 e_3 e_4 \dots e_n

At round t :

ALGORITHM

ADVERSARY

Multiplicative Weight Updates

<u>experts</u>	e_1	e_2	e_3	e_4	\dots	e_n
	$p_1^{(t)}$	$p_2^{(t)}$	$p_3^{(t)}$	$p_4^{(t)}$	\dots	$p_n^{(t)}$

ALGORITHM

$p^{(t)}$

distribution over experts

ADVERSARY

Multiplicative Weight Updates

<u>experts</u>	e_1	e_2	e_3	e_4	\dots	e_n
	$\ell_1^{(t)}$	$\ell_2^{(t)}$	$\ell_3^{(t)}$	$\ell_4^{(t)}$	\dots	$\ell_n^{(t)}$

At round t :

ALGORITHM

$p^{(t)}$



ADVERSARY

$\ell^{(t)} \in [0, 1]^n$

Experts' losses

Multiplicative Weight Updates

experts e_1 e_2 e_3 e_4 \dots e_n

At round t :

ALGORITHM

$p^{(t)}$



ADVERSARY

$\ell^{(t)}$

$$\hat{L}^{(t)} = E_{i \leftarrow p^{(t)}} \left[\ell_i^{(t)} \right]$$

Algorithm's loss

Multiplicative Weight Updates

experts e_1 e_2 e_3 e_4 \dots e_n

At round t :

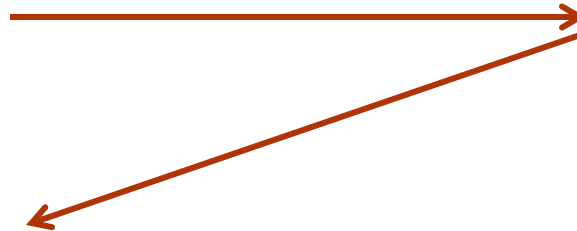
ALGORITHM

$p^{(t)}$

$p^{(t+1)}$

ADVERSARY

$\ell^{(t)}$



Update distribution

Multiplicative Weight Updates

experts e_1 e_2 e_3 e_4 \dots e_n

At round t :

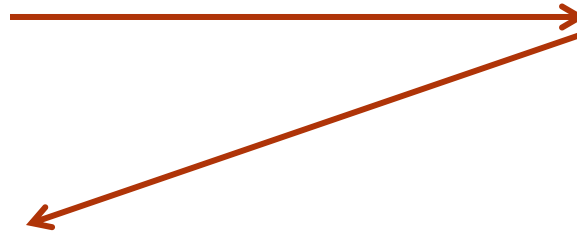
ALGORITHM

$p^{(t)}$

$p^{(t+1)}$

ADVERSARY

$\ell^{(t)}$

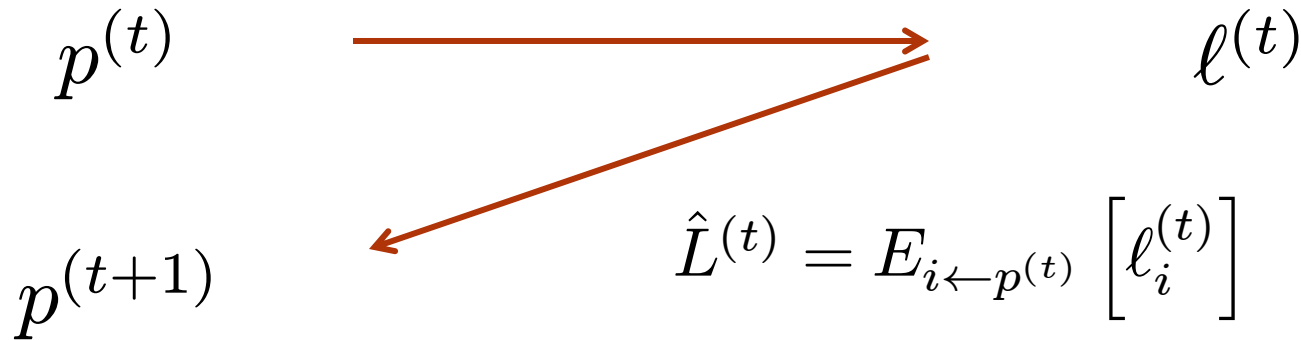


GOAL: update distribution to “minimize” total loss over time

Multiplicative Weight Updates

ALGORITHM

ADVERSARY



GOAL: update distribution to “minimize” total loss over time

$$\hat{L} = \frac{1}{T} \cdot \sum_{t=1}^T \hat{L}^{(t)} \quad \text{vs} \quad L^* = \min_i \frac{1}{T} \cdot \sum_{t=1}^T \ell_i^{(t)}$$

Average Algorithm's Loss

Average Loss of Best Expert

Multiplicative Weight Updates

ALGORITHM

$p^{(t)}$



ADVERSARY

$\ell^{(t)}$

$$p_i^{(t+1)} \propto (1 - \epsilon)^{\ell_i^{(t)}} \cdot p^{(t)}$$

MULTIPLICATIVE UPDATE

Multiplicative Weight Updates

ALGORITHM

$p^{(t)}$

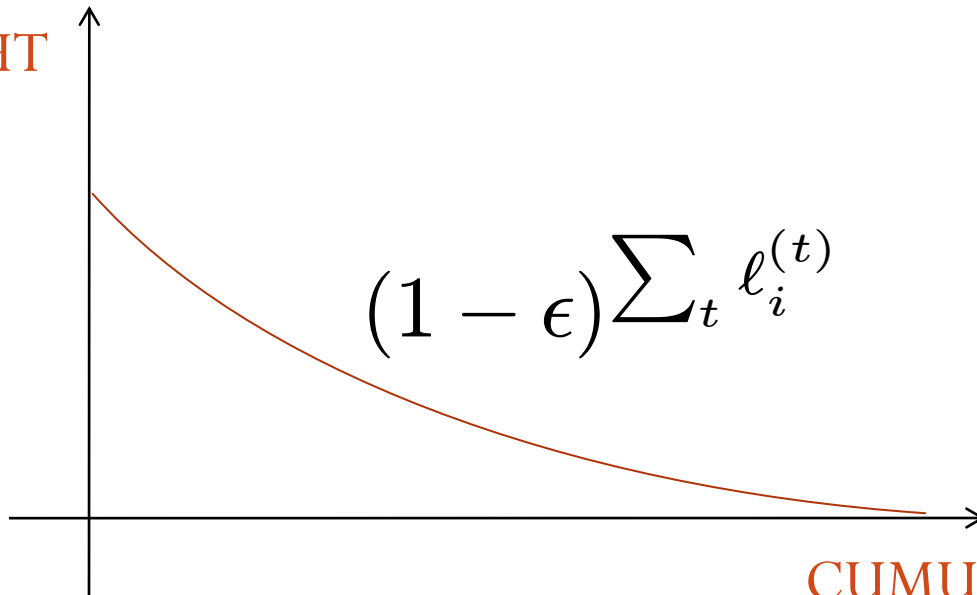


ADVERSARY

$\ell^{(t)}$

$$p_i^{(t+1)} \propto (1 - \epsilon)^{\ell_i^{(t)}} \cdot p^{(t)}$$

WEIGHT



CUMULATIVE LOSS

Multiplicative Weight Updates

ALGORITHM

$p^{(t)}$



ADVERSARY

$\ell^{(t)}$

$$p_i^{(t+1)} \propto (1 - \epsilon)^{\ell_i^{(t)}} \cdot p^{(t)}$$

If $T \geq O\left(\frac{\log n}{\epsilon \delta}\right)$ and $\ell^{(t)} \in [0, 1]^n$,

$$\hat{L} \cdot \delta + (1 + \epsilon)L^*$$

Multiplicative Weight Updates

ALGORITHM

$p^{(t)}$



ADVERSARY

$\ell^{(t)}$

$$p_i^{(t+1)} \propto (1 - \epsilon)^{\ell_i^{(t)}} \cdot p_i^{(t)}$$

If $T \geq O\left(\frac{\log n}{\epsilon \delta}\right)$ and $\ell^{(t)} \in [0, 1]^n$,

$$\hat{L} \leq \delta + (1 + \epsilon)L^*$$

NB: For large T , algorithm performs close to best expert.

Multiplicative Weight Updates

ALGORITHM

$p^{(t)}$



ADVERSARY

$\ell^{(t)}$

$$p_i^{(t+1)} \propto (1 - \epsilon)^{\ell_i^{(t)}} \cdot p_i^{(t)}$$

If $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$ and $\ell^{(t)} \in [-\sigma, \rho]^n$,

$$\hat{L} \leq \delta + (1 + \epsilon)L^* \quad \text{GENERALIZATION}$$

NB: For large T , algorithm performs close to best expert.

MWUs and LPs

RESULT: Given a separation oracle with width $(-\sigma, \rho)$, MWUs allow to check feasibility up to δ in T oracle calls,

$$T \cdot O\left(\frac{\sigma \rho \log n}{\delta^2}\right)$$

UPDATE IS FAST, RUNNING TIME DOMINATED BY ORACLE

MWUs and LPs

RESULT: Given a separation oracle with width $(-\sigma, \rho)$, MWUs allow to check feasibility up to δ in T oracle calls,

$$T \cdot O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$$

FEASIBILITY PROBLEM

$$Ax \geq b$$

$$x \geq 0$$

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

Probability distribution over constraints

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

- Check feasibility of dual
- One expert for each primal constraint

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

- Check feasibility of dual
- One expert for each primal constraint
- At round t ,

ALGORITHM

$p^{(t)}$



ADVERSARY

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

- Check feasibility of dual
- One expert for each primal constraint
- At round t ,

ALGORITHM

$p^{(t)}$



SEPARATION ORACLE

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

SEPARATION ORACLE either

- Declares $p^{(t)}$ feasible,

DONE

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

SEPARATION ORACLE either

- Declares $p^{(t)}$ feasible,
- Returns $x^{(t)}$ such that: $p^{(t)T} Ax^{(t)} \geq p^{(t)T} b$

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$x^{(t)}$$

$$p^{(t)T} Ax^{(t)} \geq p^{(t)T} b$$

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$x^{(t)}$$

$$p^{(t)T} Ax^{(t)} \geq p^{(t)T} b$$

LOSS FUNCTION ?

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

IDEA: penalize constraints that are satisfied more easily by oracle.

MWUs and LPs

PRIMAL

$$Ax \geq b$$

$$x \geq 0$$

DUAL

$$p^T A < 0$$

$$p^T b \geq 0$$

$$p \geq 0, p^T \mathbf{1} = 1$$

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

Width of oracle is width of loss function

$$-\sigma \cdot (Ax^{(t)})_i - b_i \cdot \rho$$

MWUs and LPs

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

$$-\sigma \cdot (Ax^{(t)})_i - b_i \cdot \rho$$

Regret bound yields for $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$

$$\hat{L} \cdot \delta + (1 + \epsilon)L^*$$

MWUs and LPs

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

$$-\sigma \cdot (Ax^{(t)})_i - b_i \cdot \rho$$

Regret bound yields for $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$

$$0 \cdot \hat{L} \cdot \delta + (1 + \epsilon)L^*$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

MWUs and LPs

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

$$-\sigma \cdot (Ax^{(t)})_i - b_i \cdot \rho$$

Regret bound yields for $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$

$$0 \cdot \delta + (1 + \epsilon) \left(\sum_{t=1}^T Ax_i^{(t)} - b_i \right)$$

MWUs and LPs

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

$$-\sigma \cdot (Ax^{(t)})_i - b_i \cdot \rho$$

Regret bound yields for $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$

$$\forall i, \quad A\bar{x}_i - b_i \geq -\frac{\delta}{1+\epsilon} > -\delta$$

MWUs and LPs

ALGORITHM

$$p^{(t)}$$



SEPARATION ORACLE

$$\ell^{(t)} = Ax^{(t)} - b$$

$$p^{(t)T} \ell^{(t)} \geq 0$$

$$-\sigma \cdot (Ax^{(t)})_i - b_i \cdot \rho$$

Regret bound yields for $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$

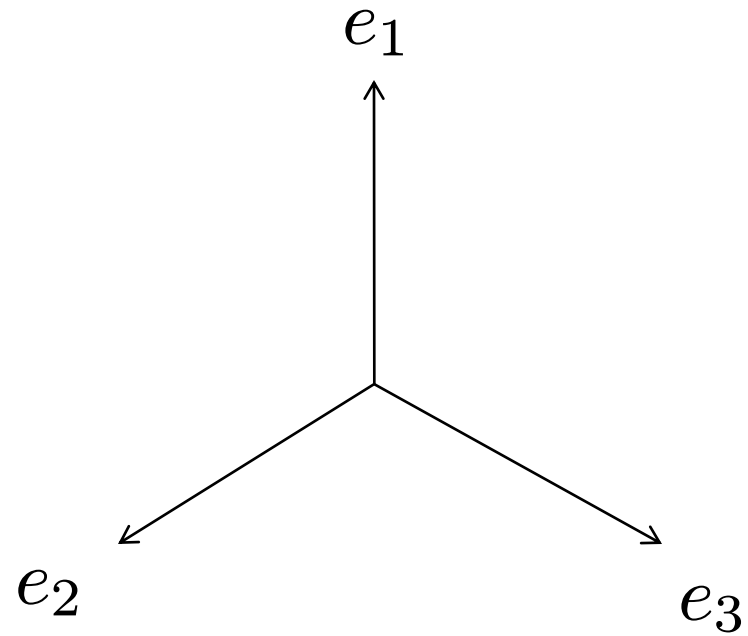
$$\forall i, \quad A\bar{x}_i - b_i \geq -\frac{\delta}{1+\epsilon} > -\delta$$

The average of the oracle responses is approximately feasible.

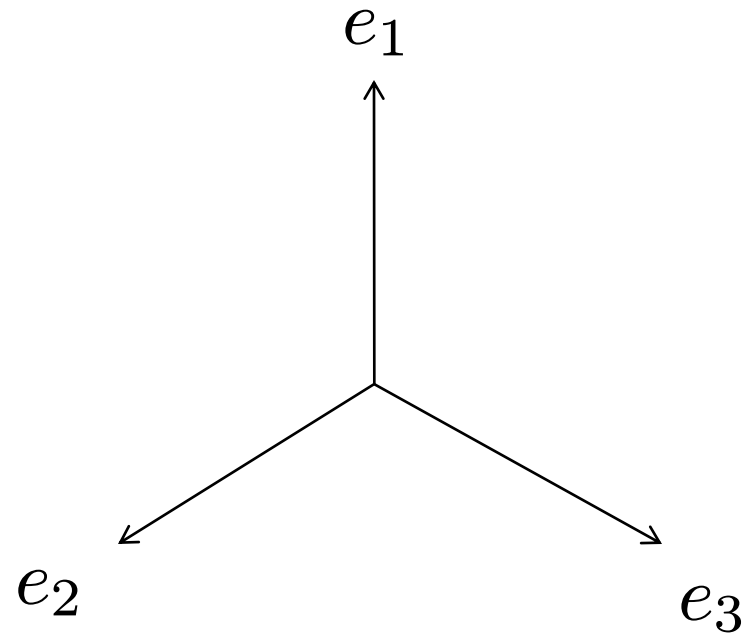
Applications

- Fast Algorithms for problems captured by LP
 - Multi-commodity Flows
 - Single-commodity Flows
 - Covering-packing problems
- Proof Technique
 - Hardcore Lemma
- Machine Learning
 - Boosting

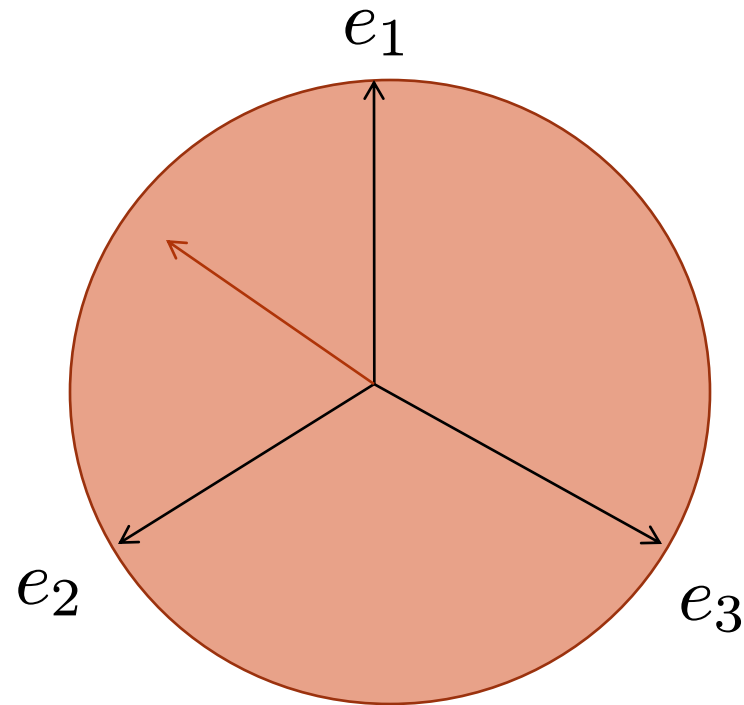
Matrix MWUs



Matrix MWUs

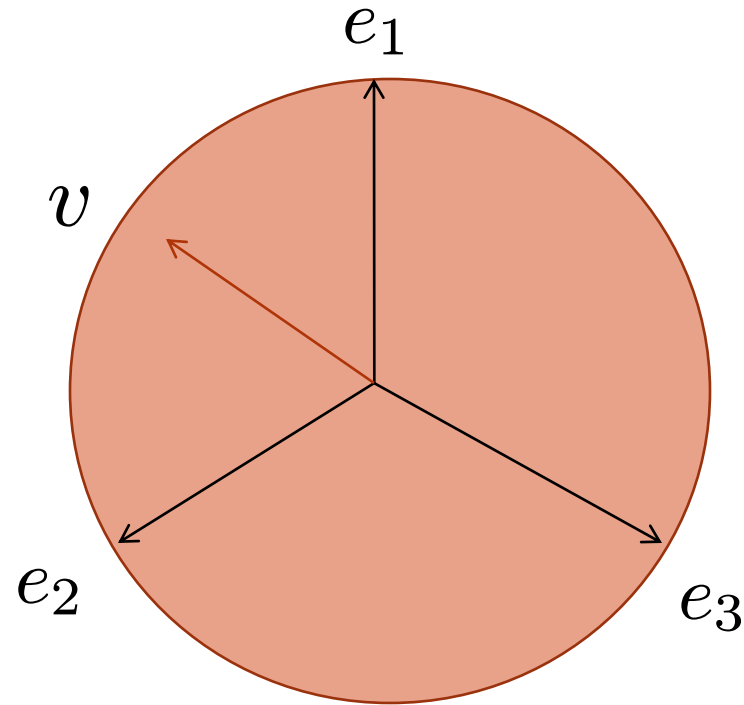


Matrix MWUs



Extend set of experts to whole sphere

Matrix MWUs



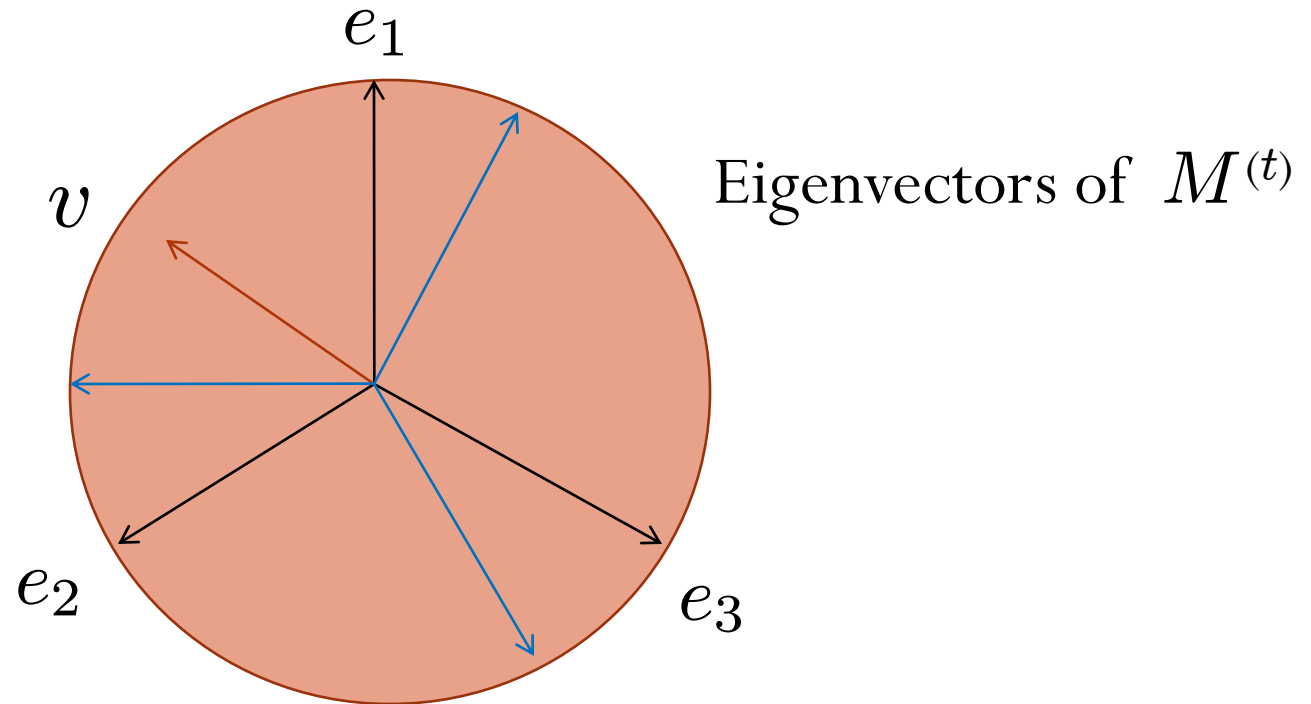
LOSS MATRIX:

$$0 \preceq M^{(t)} \preceq I$$

LOSS FUNCTION:

$$\ell^{(t)}(v) = v^T M^{(t)} v$$

Matrix MWUs



LOSS MATRIX:

$$0 \preceq M^{(t)} \preceq I$$

LOSS FUNCTION:

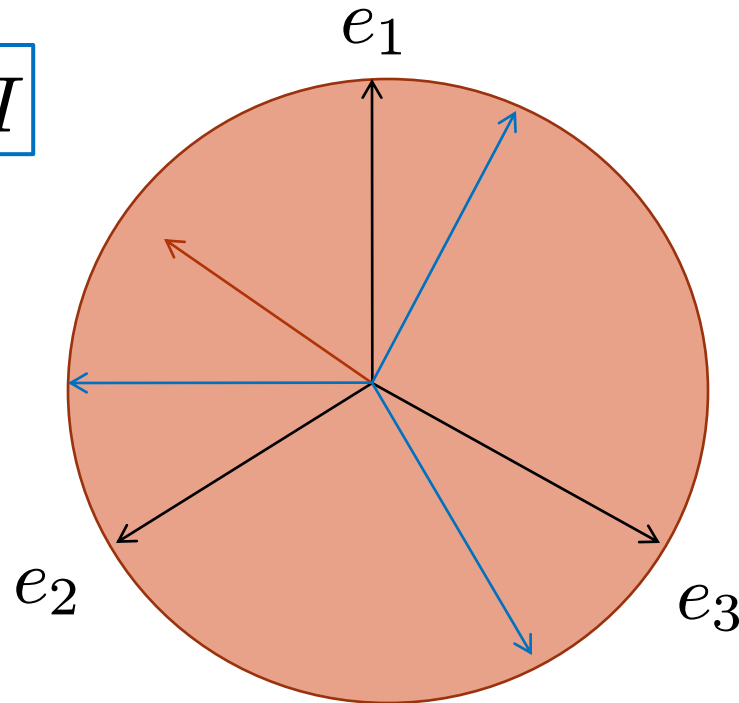
$$\ell^{(t)}(v) = v^T M^{(t)} v$$

Matrix MWUs

$$0 \preceq M^{(t)} \preceq I$$

distribution

$$D^{(t)}$$



ALGORITHM'S LOSS:

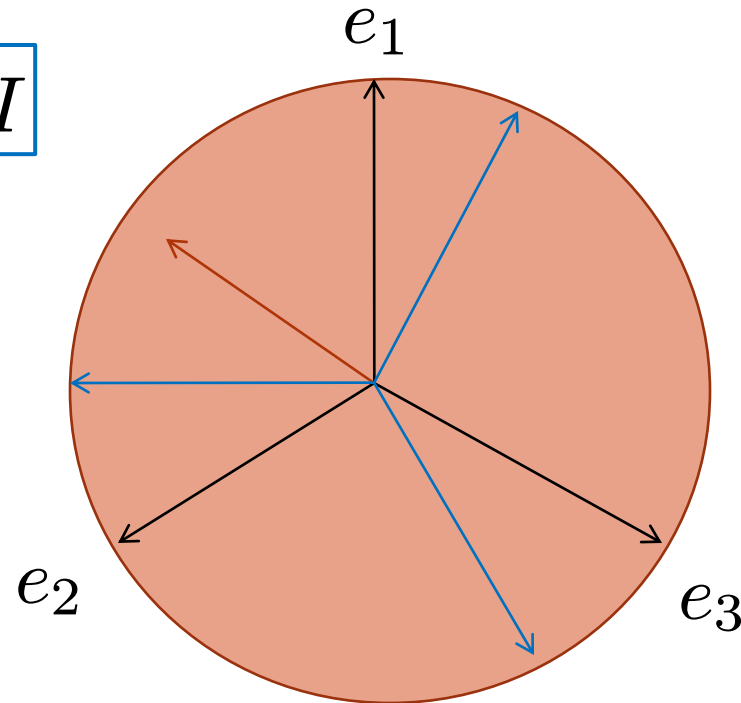
$$\hat{L}^{(t)} = E_{v \leftarrow D^{(t)}} [\ell^{(t)}(v)]$$

Matrix MWUs

$$0 \preceq M^{(t)} \preceq I$$

distribution

$$D^{(t)}$$



ALGORITHM'S LOSS:

$$\hat{L}^{(t)} = E_{v \leftarrow D^{(t)}} [\ell^{(t)}(v)] = M^{(t)} \bullet X^{(t)}$$

$X^{(t)}$ is density matrix corresponding to $D^{(t)}$

Matrix Multiplicative Weight Updates

ALGORITHM

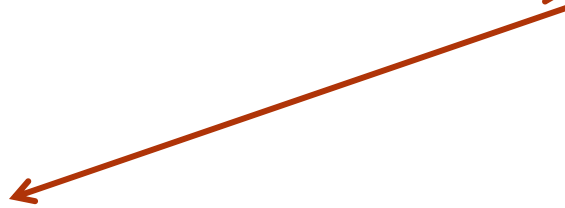
ADVERSARY

$$X^{(t)}$$



$$M^{(t)}$$

$$X^{(t+1)}$$



$$p^{(t)} \propto (1 - \epsilon) \sum_t \ell_i^{(t)}$$

MULTIPLICATIVE UPDATE

Matrix Multiplicative Weight Updates

ALGORITHM

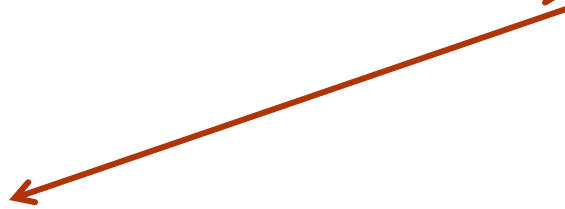
ADVERSARY

$$X^{(t)}$$



$$M^{(t)}$$

$$X^{(t+1)}$$



$$X^{(t)} \propto (1 - \epsilon) \sum_t M^{(t)}$$

MULTIPLICATIVE UPDATE

Matrix Multiplicative Weight Updates

ALGORITHM

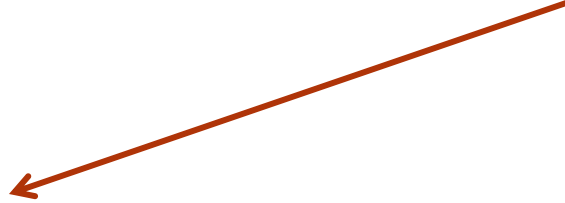
ADVERSARY

$X^{(t)}$



$M^{(t)}$

$X^{(t+1)}$



$$X^{(t)} \propto (1 - \epsilon) \sum_t M^{(t)} = e^{-\eta} \sum_t M^{(t)}$$

MATRIX EXPONENTIAL

Matrix Multiplicative Weight Updates

ALGORITHM

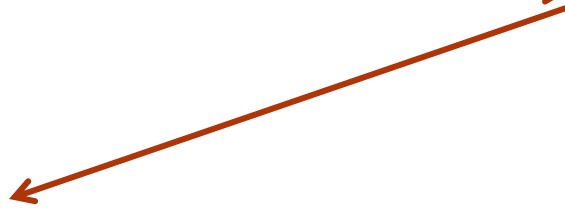
ADVERSARY

$$X^{(t)}$$



$$M^{(t)}$$

$$X^{(t+1)}$$



$$X^{(t)} \propto (1 - \epsilon) \sum_t M^{(t)}$$

Regret Bound

If $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$ and $\ell^{(t)} \in [-\sigma, \rho]^n$

$$\hat{L} \cdot \delta + (1 + \epsilon)L^*$$

Matrix Multiplicative Weight Updates

ALGORITHM

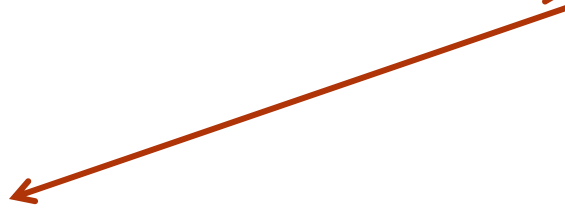
ADVERSARY

$$X^{(t)}$$



$$M^{(t)}$$

$$X^{(t+1)}$$



$$X^{(t)} \propto (1 - \epsilon) \sum_t M^{(t)}$$

Regret Bound

If $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$ and $\ell^{(t)} \in [-\sigma, \rho]^n$

$$\hat{L} \cdot \delta + (1 + \epsilon)L^*$$

where $L^* = \lambda_{\min}\left(\sum_{t=1}^T M^{(t)}\right)$

Matrix MWUs and SDPs

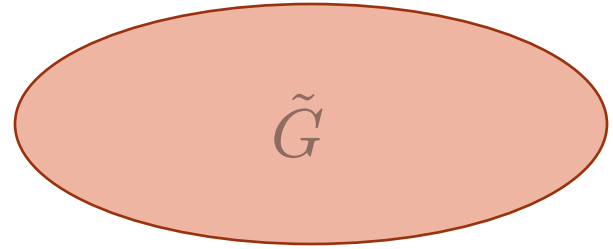
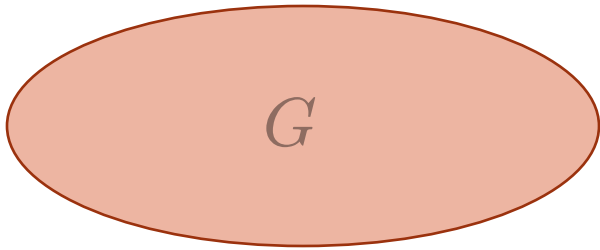
- Application to solving SDPs [Arora, Kale]
- Used in fast algorithms for
 - Graph partitioning
 - Maxcut
 - CSP problems
- Also as a proof technique
 - QIP = PSPACE
 - Concentration bounds

Matrix Updates and Spectral Methods

Two Examples

Spectral Sparsification

Spectral Sparsification

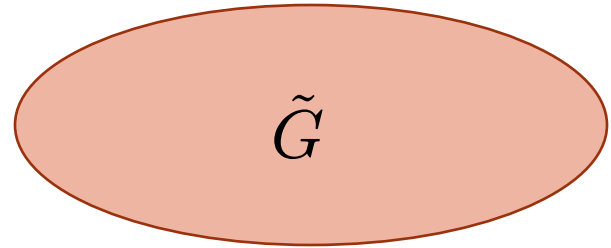
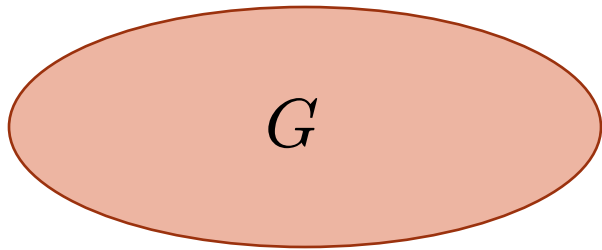


SPARSE – only T edges

$\forall x$

$$x^T L(G)x \approx x^T L(\tilde{G})x$$

Spectral Sparsification



SPARSE – only T edges

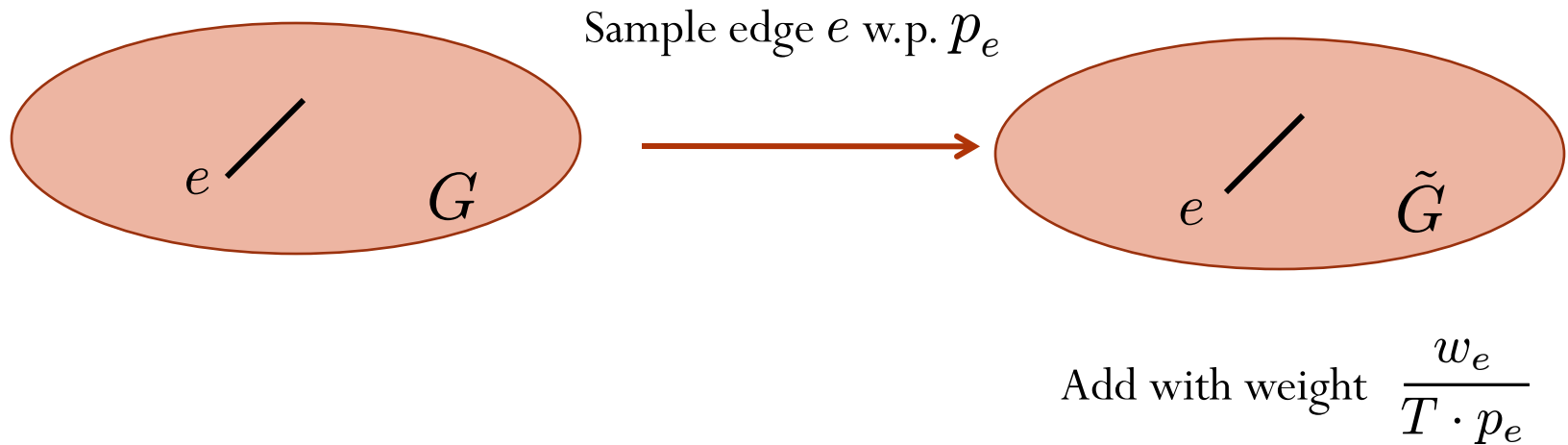
$\forall x$

$$|x^T L(G)x - x^T L(\tilde{G})x| \cdot$$

$$\left\{ \begin{array}{l} \epsilon \cdot x^T D(G)x \\ \text{ABSOLUTE ERROR} \\ \epsilon \cdot x^T L(G)x \\ \text{RELATIVE ERROR} \end{array} \right.$$

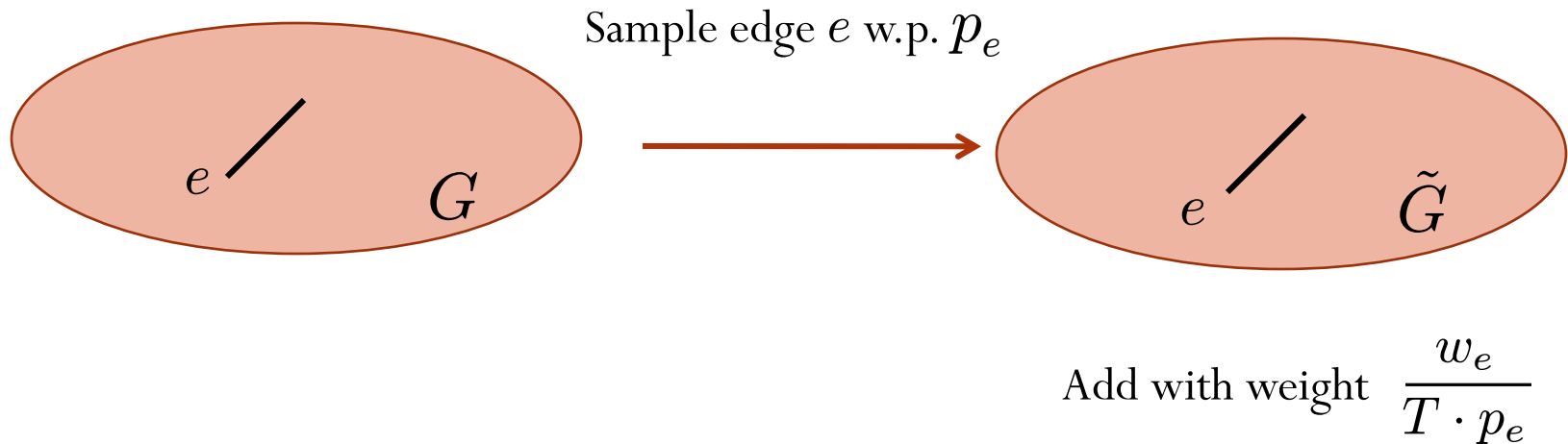
Sparsification by Sampling

For T iterations,



Sparsification by Sampling

For T iterations,



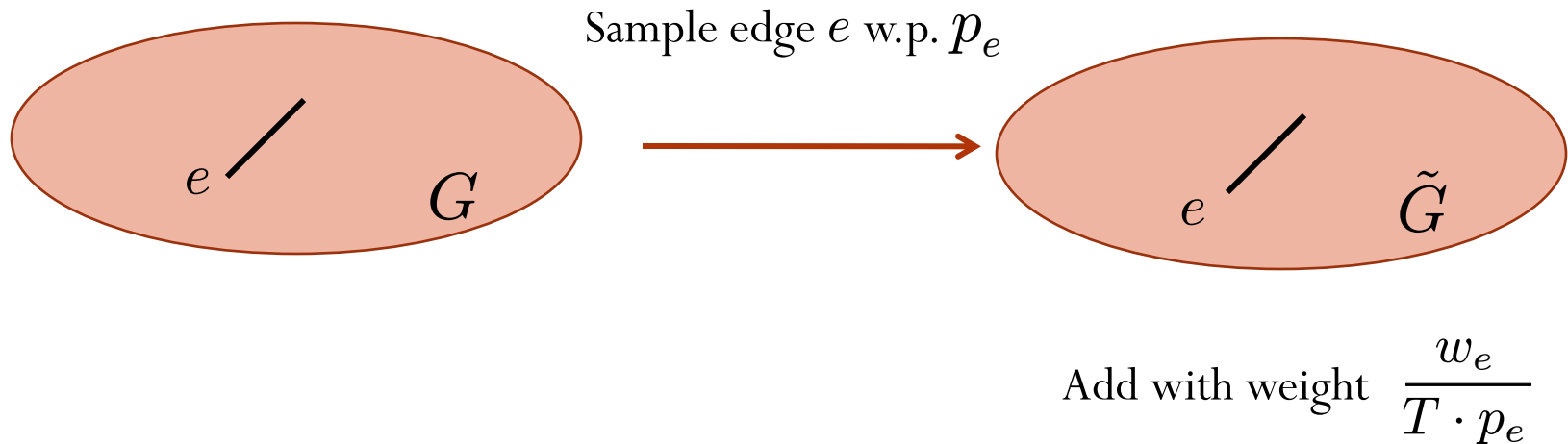
PREVIOUS RESULTS:

- Absolute error: $p_{\{i,j\}} \propto \frac{1}{n} \left(\frac{w_{ij}}{d_i} + \frac{w_{ij}}{d_j} \right)$ $T = O \left(\frac{n \log n}{\epsilon^2} \right)$

[Komlos, Furedi] [Spielman Teng]

Sparsification by Sampling

For T iterations,



PREVIOUS RESULTS:

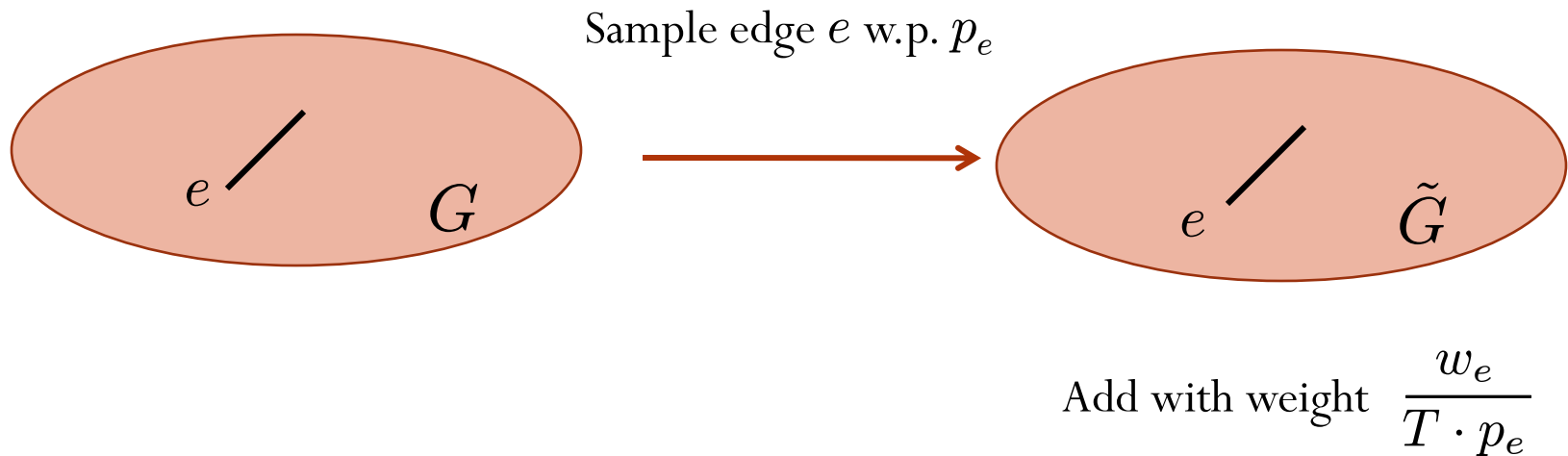
- Absolute error: $p_{\{i,j\}} \propto \left(\frac{w_{ij}}{d_i} + \frac{w_{ij}}{d_j} \right)$ $T = O\left(\frac{n \log n}{\epsilon^2}\right)$

- Relative error: $p_{\{i,j\}} \propto R_{ij}$ $T = O\left(\frac{n \log n}{\epsilon^2}\right)$

[Spielman Srivastava]

Sparsification by Sampling

For T iterations,

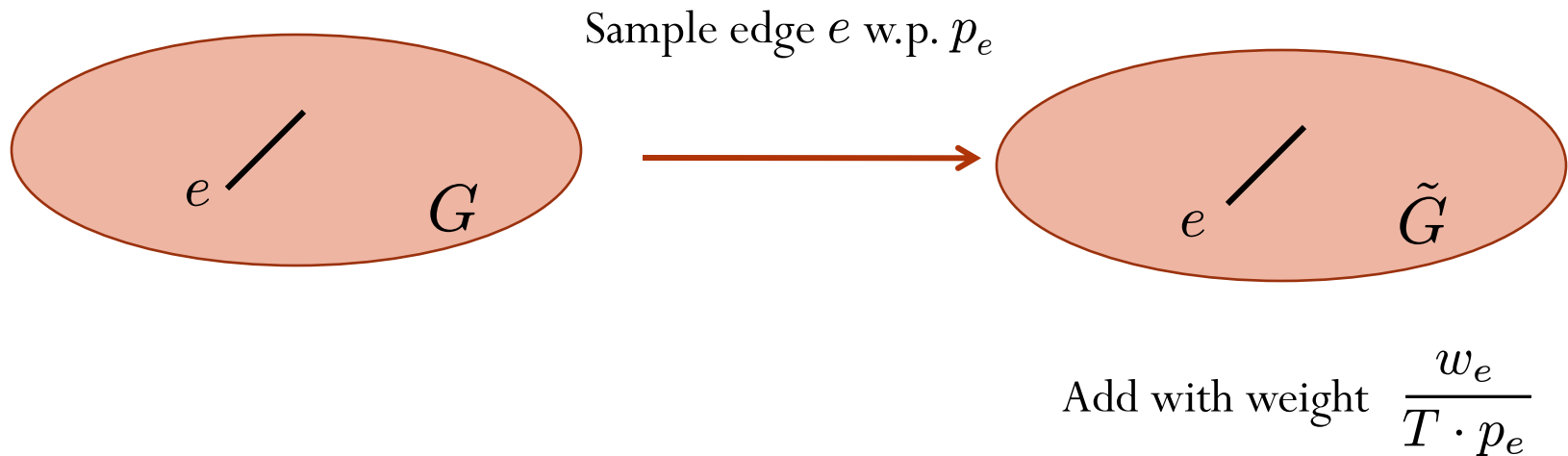


$$p_{\{i,j\}} \propto \left(\frac{w_{ij}}{d_i} + \frac{w_{ij}}{d_j} \right) = (e_i - e_j)^T D(G)^{-1} (e_i - e_j)$$

$$p_{\{i,j\}} \propto R_{ij} = (e_i - e_j)^T L(G)^+ (e_i - e_j)$$

Sparsification by Sampling

For T iterations,



$$p_{\{i,j\}} \propto \left(\frac{w_{ij}}{d_i} + \frac{w_{ij}}{d_j} \right) = (e_i - e_j)^T \underline{D(G)}^{-1} (e_i - e_j)$$

$$p_{\{i,j\}} \propto R_{ij} = (e_i - e_j)^T \underline{L(G)}^+ (e_i - e_j)$$

ERROR MATRIX

Generalization

Given **error matrix** R , $R \preceq L(G)$

$\forall x \notin \ker(R)$

$$|x^T L(G)x - x^T L(\tilde{G})x| \leq \epsilon \cdot x^T R x$$

Generalization

Given **error matrix** R , $R \preceq L(G)$

$\forall x \notin \ker(R)$

$$|x^T L(G)x - x^T L(\tilde{G})x| \leq \epsilon \cdot x^T R x$$

RESULT:

$$p_{\{i,j\}} \propto (e_i - e_j)^T R^+ (e_i - e_j) \quad T = O\left(\frac{Z \log \text{rank}(R)}{\epsilon^2}\right)$$

where $Z = \sum_{\{i,j\} \in E} (e_i - e_j)^T R^+ (e_i - e_j)$

Generalization

Given **error matrix** R , $R \preceq L(G)$

$\forall x \notin \ker(R)$

$$|x^T L(G)x - x^T L(\tilde{G})x| \leq \epsilon \cdot x^T R x$$

RESULT:

$$p_{\{i,j\}} \propto (e_i - e_j)^T R^+ (e_i - e_j) \quad T = O\left(\frac{Z \log \text{rank}(R)}{\epsilon^2}\right)$$

where $Z = \sum_{\{i,j\} \in E} (e_i - e_j)^T R^+ (e_i - e_j)$

For $R = L(G)$,

$$Z = \sum_{\{i,j\} \in E} R_{ij} = n - 1$$

Matrix MWU Proof

$$M^{(t)} = R^{-\frac{1}{2}} \left(L - \frac{1}{p_{e_t}} L_{e_t} \right) R^{-\frac{1}{2}}$$

EXPECTED LOSS :

$$E[\hat{L}^{(t)}] = E[M^{(t)} \bullet X^{(t)}] = 0$$

WIDTH:

$$-\sigma = -Z \preceq M^{(t)} \preceq 1 = \rho$$

Regret Bound

If $T \geq O\left(\frac{\rho\sigma \log n}{\delta^2}\right)$ and $\ell^{(t)} \in [-\sigma, \rho]^n$

$$\hat{L} \cdot \delta + (1 + \epsilon') L^*$$

where $L^* = \lambda_{\min}\left(\sum_{t=1}^T M^{(t)}\right)$

Matrix MWU Proof

$$M^{(t)} = R^{-\frac{1}{2}} \left(L - \frac{1}{p_{e_t}} L_{e_t} \right) R^{-\frac{1}{2}}$$

EXPECTED LOSS :

$$E[\hat{L}^{(t)}] = E[M^{(t)} \bullet X^{(t)}] = 0$$

WIDTH:

$$-\sigma = -Z \preceq M^{(t)} \preceq 1 = \rho$$

Regret Bound

$$\text{If } T \geq O \left(\frac{Z \log \text{rank}(R)}{\delta^2} \right)$$

$$0 \cdot \delta + (1 + \epsilon') L^*$$

$$\text{where } L^* = \lambda_{\min} \left(\sum_{t=1}^T M^{(t)} \right)$$

Matrix MWU Proof

$$M^{(t)} = R^{-\frac{1}{2}} \left(L - \frac{1}{p_{e_t}} L_{e_t} \right) R^{-\frac{1}{2}}$$

EXPECTED LOSS :

$$E[\hat{L}^{(t)}] = E[M^{(t)} \bullet X^{(t)}] = 0$$

WIDTH:

$$-\sigma = -Z \preceq M^{(t)} \preceq 1 = \rho$$

Regret Bound

$$\text{If } T \geq O \left(\frac{Z \log \text{rank}(R)}{\delta^2} \right)$$

$$\lambda_{\min} \left(R^{-\frac{1}{2}} \left(L(G) - L(\tilde{G}) \right) R^{-\frac{1}{2}} \right) \geq -\delta$$

Matrix MWU Proof

$$M^{(t)} = R^{-\frac{1}{2}} \left(L - \frac{1}{p_{e_t}} L_{e_t} \right) R^{-\frac{1}{2}}$$

EXPECTED LOSS :

$$E[\hat{L}^{(t)}] = E[M^{(t)} \bullet X^{(t)}] = 0$$

WIDTH:

$$-\sigma = -Z \preceq M^{(t)} \preceq 1 = \rho$$

Regret Bound

$$\text{If } T \geq O \left(\frac{Z \log \text{rank}(R)}{\delta^2} \right)$$

$$L(\tilde{G}) - L(G) \preceq \delta$$

Similar proof for other direction

Balanced Cut

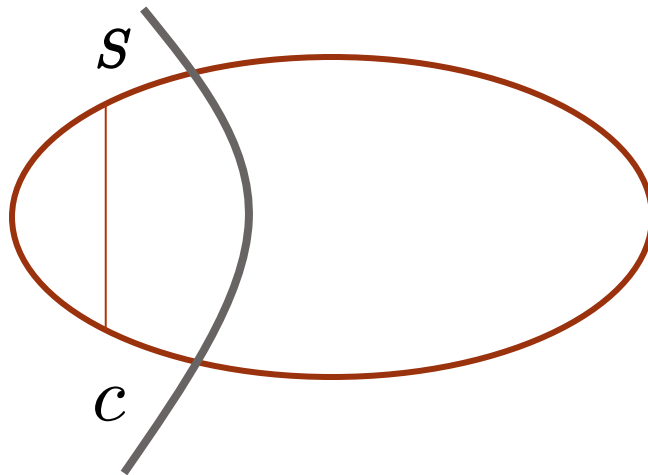
Graph Partitioning – Balanced Cut

Conductance of $S \subseteq V$

$$\phi(S) = \frac{w(S, \bar{S})}{\min\{\text{vol}(S), \text{vol}(\bar{S})\}}$$

DECISION PROBLEM:

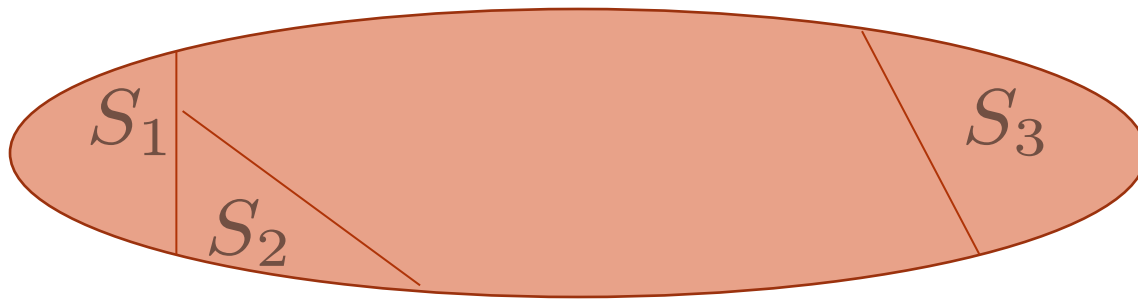
does G have a c -balanced cut of conductance $< \gamma$?



$$\frac{1}{2} > \frac{\text{vol}(S)}{\text{vol}(V)} > c$$

Previous Spectral Algorithms

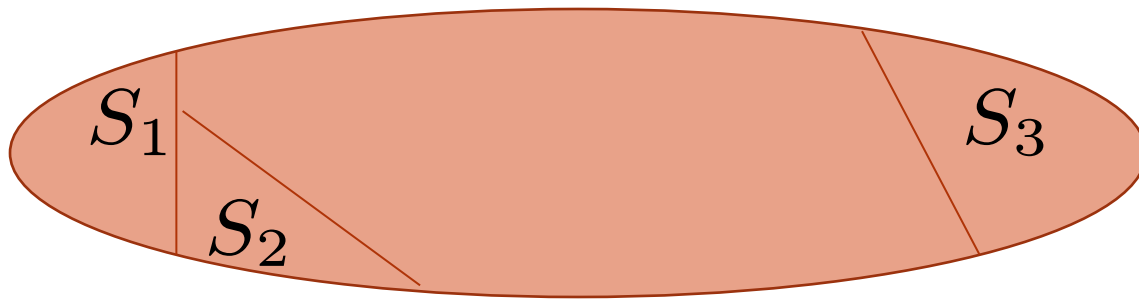
- Recursive Eigenvector



Previous Spectral Algorithms

- Recursive Eigenvector

- Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma})$



Previous Spectral Algorithms

- Recursive Eigenvector

- Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma})$
- Certificate is also sparse cut



CERTIFICATE:

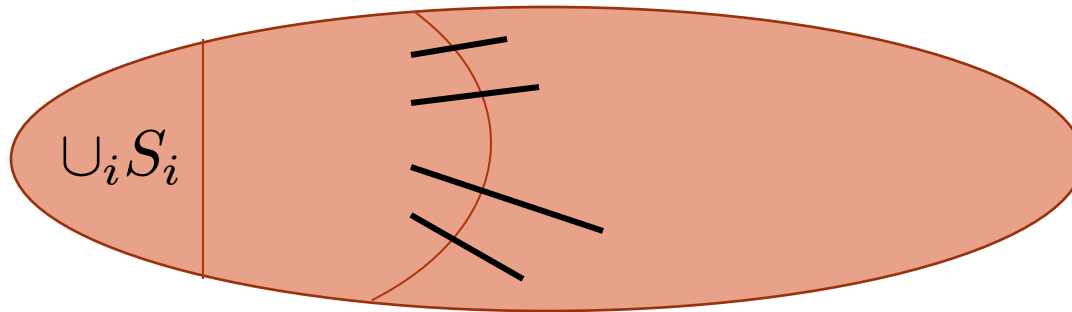
Graph induced on $V - \cup_i S_i$ has spectral gap $\geq \gamma$ and

$$\text{vol}(V - \cup_i S_i) \geq (1 - \frac{c}{2})\text{vol}(V)$$

Previous Spectral Algorithms

- Recursive Eigenvector

- Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma})$
- Certificate is also sparse cut



CERTIFICATE:

Graph induced on $V - \cup_i S_i$ has spectral gap $\geq \gamma$ and

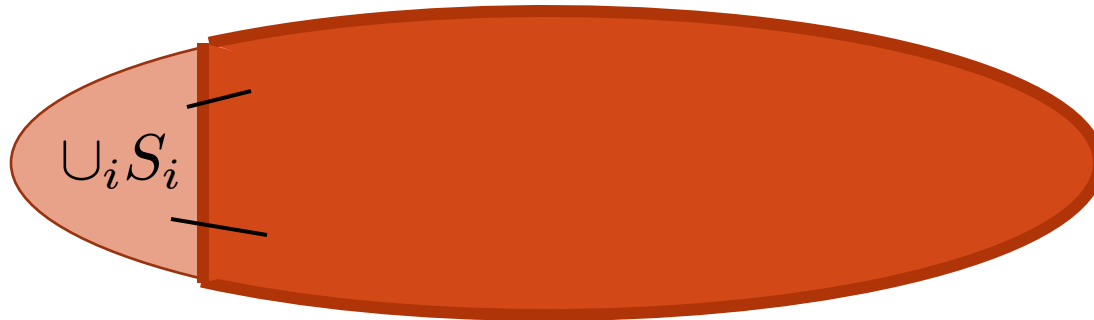
$$\text{vol}(V - \cup_i S_i) \geq (1 - \frac{c}{2})\text{vol}(V)$$

Previous Spectral Algorithms

- Recursive Eigenvector

- Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma})$
- Certificate is also **sparse** cut

$$\phi(\cup_i S_i) \cdot O(\sqrt{\gamma})$$

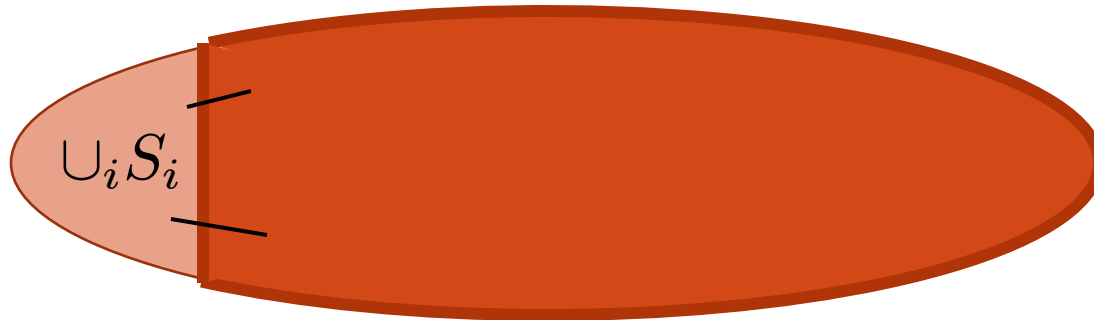


Important property in construction of **graph decompositions**.

Decompositions play role in clustering, sparsification, preconditioning, ...

Previous Spectral Algorithms

- Recursive Eigenvector
 - Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma})$
 - Certificate is also sparse cut , useful in decomposition



BUT

- n iterations may be necessary.
- Running time can be $\Omega(n^2)$

New Algorithms for Balanced Partitioning

- Local Random Walks [Spielman, Teng]
 - Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma \log^c n})$
 - Runs in time $\tilde{O}(m/\gamma)$
 - Certificate cut is sparse

New Algorithms for Balanced Partitioning

- Local Random Walks [Spielman, Teng]
 - Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma \log^c n})$
 - Runs in time $\tilde{O}(m/\gamma)$
 - Certificate cut is sparse
 - Crucial advance in nearly-linear-time algorithms

New Algorithms for Balanced Partitioning

- Local Random Walks [Spielman, Teng]
 - Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma \log^c n})$
 - Runs in time $\tilde{O}(m/\gamma)$
 - Certificate cut is sparse
 - Crucial advance in nearly-linear-time algorithms

BUT

- Suboptimal approximation
- Involved algorithm and analysis
- Based on many truncated local random walks

Our Result

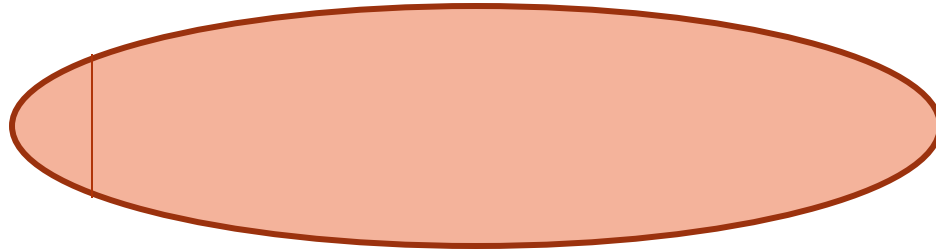
- SDP-based Spectral Algorithm [Orecchia, Vishnoi]
 - Distinguish between $\phi_G \leq \gamma$ and $\phi_G > O(\sqrt{\gamma})$
 - Runs in time $\tilde{O}(m/\gamma)$
 - Certificate cut is sparse

MOREOVER

- Simple algorithm
- Uses random walks in a natural way
- Achieves asymptotically optimal bounds for spectral algorithms
- Uses combination of Matrix MWUs and spectral methods

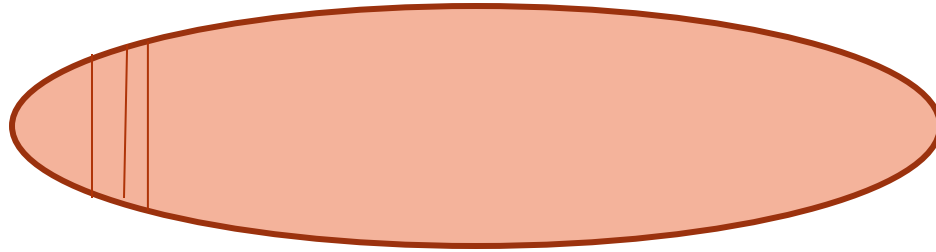
Intuition Behind Result

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.



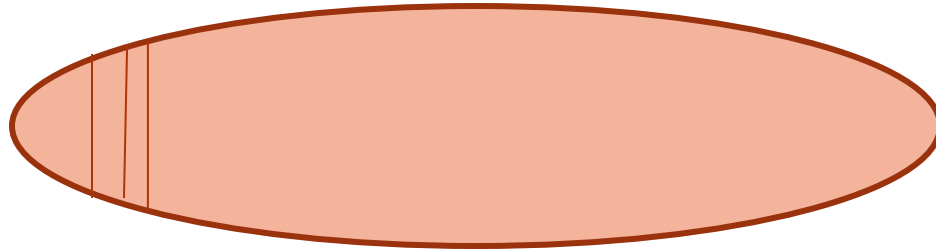
Intuition Behind Result

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.



Intuition Behind Result

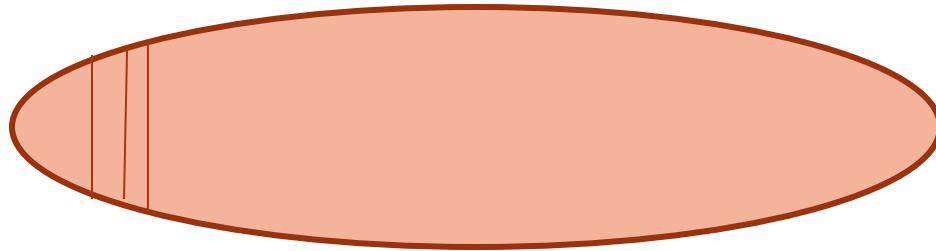
- Recall: small sparse cuts are **obstacle** to finding balanced cuts.



- Can we remove multiple small cuts at once?

Intuition Behind Result

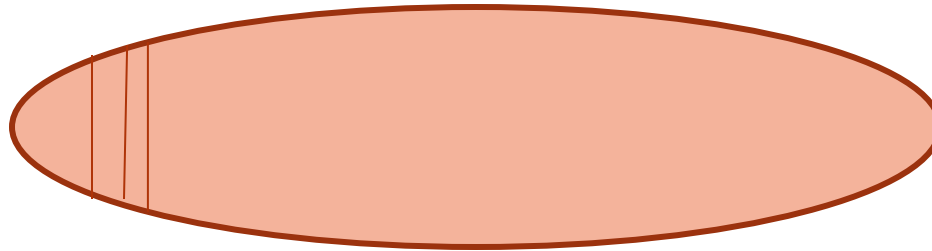
- Recall: small sparse cuts are **obstacle** to finding balanced cuts.



- Can we remove multiple small cuts at once?
- Not possible with eigenvector, use randomized approximation

Intuition Behind Result

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.

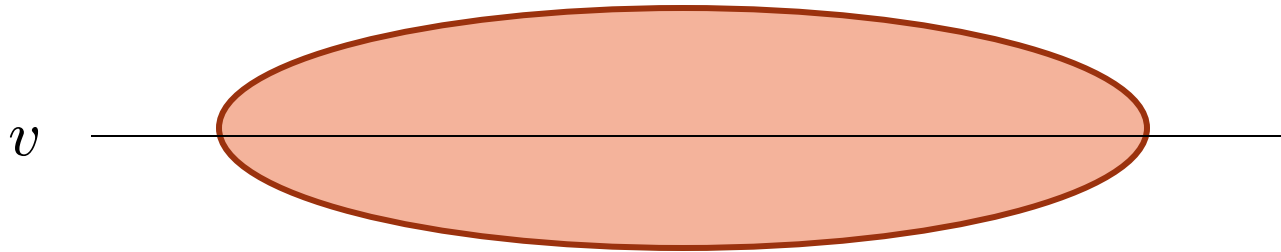


- Can we remove multiple small cuts at once?
- Not possible with eigenvector, use randomized approximation

Final distribution of random walk started at random seed

Intuition Behind Result

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.

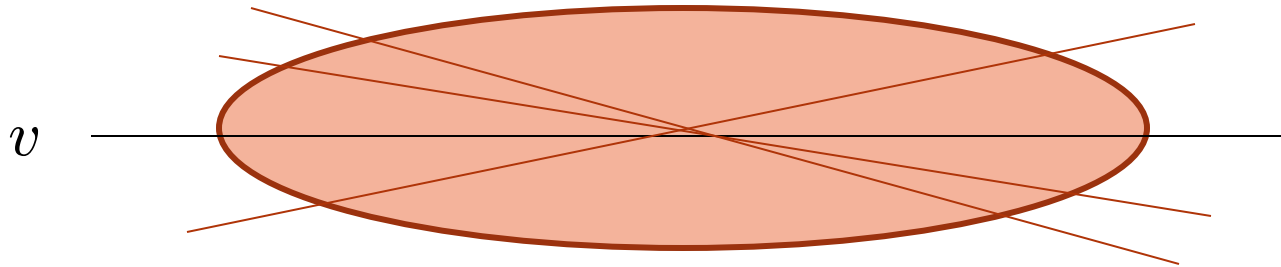


- Can we remove multiple small cuts at once?
- Not possible with eigenvector, use randomized approximation

Final distribution of random walk started at random seed

Intuition Behind Result

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.

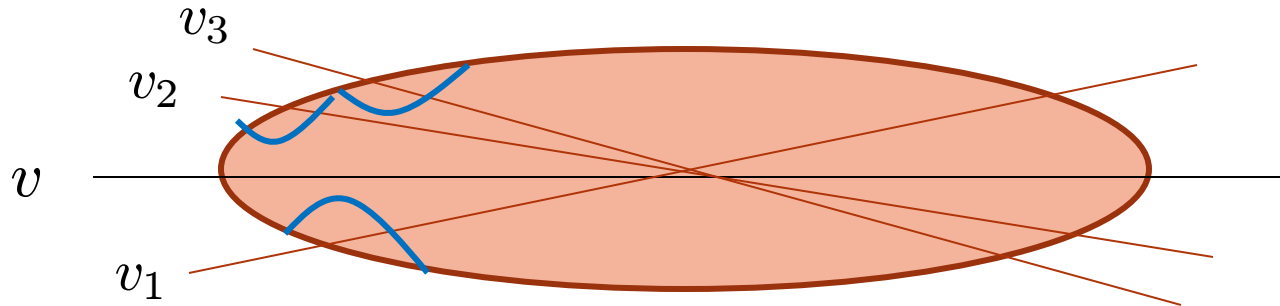


- Can we remove multiple small cuts at once?
- Not possible with eigenvector, use randomized approximation

Final distribution of random walk started at random seed

Intuition Behind Result

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.



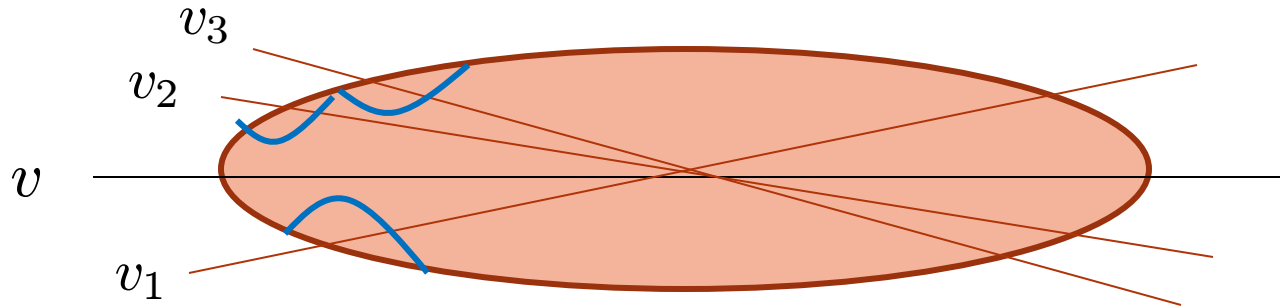
- Can we remove multiple small cuts at once?
- Not possible with eigenvector, use randomized approximation

Final distribution of random walk started at random seed

- Either one v_i has a sparse balanced sweep cut, or we can remove “many” **sparse small cuts**.

Intuition Behind Result -

- Recall: small sparse cuts are **obstacle** to finding balanced cuts.



- Can we remove multiple small cuts at once?
- Not possible with eigenvector, use randomized approximation

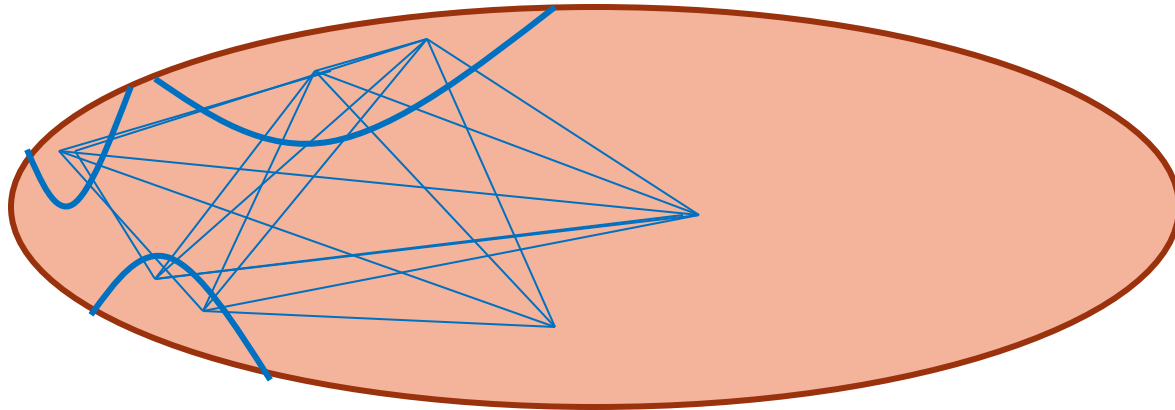
Final distribution of random walk started at random seed

- Either some v_i has a sparse balanced sweep cut, or we can remove “many” **sparse small cuts**.

NB: Matrix MWUs effectively capture **HEDGING**.

Intuition Behind Result - Hedging

- Matrix MWUs effectively capture **HEDGING**
 1. Among approximate eigenvectors.
 2. Among unbalanced cuts to remove. **SOFT REMOVAL**



Add weighted $K_{S,V}$ to every sparse unbalanced S found.

Algorithm Description

- At iteration t , we start with graph G_t and a rate η_t . $G_0 = G$ and $\eta_0 = 1$.

Algorithm Description

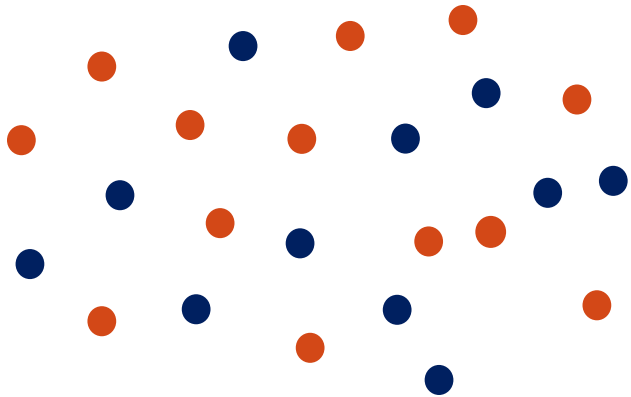
- For each t , keep graph G_t and a rate η_t . $G_0 = G$ and $\eta_0 = 1$.

● = +1 charge

● = -1 charge

At iteration t , repeat $O(\log n)$ times:

- Consider G_t .
- Pick random assignment of ± 1 charge to the vertices.



Algorithm Description

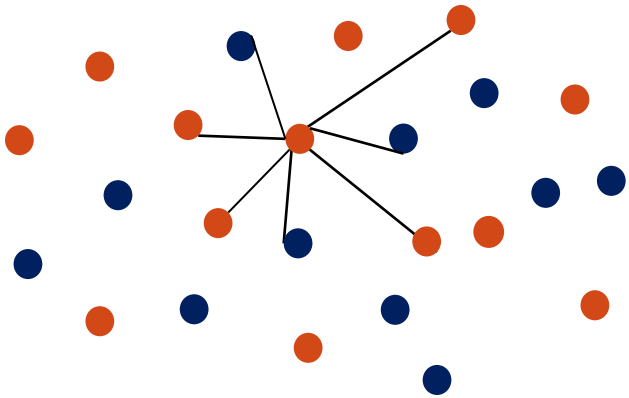
- For each t , keep graph G_t and a rate η_t . $G_0 = G$ and $\eta_0 = 1$.

● = +1 charge

● = -1 charge

At iteration t , repeat $O(\log n)$ times:

- Consider G_t .
- Pick random assignment of ± 1 charge to the vertices.
- Mix charge along edges of G_t using heat kernel with rate η_t .



Algorithm Description

- For each t , keep graph G_t and a rate η_t . $G_0 = G$ and $\eta_0 = 1$.

● = +1 charge



● = -1 charge

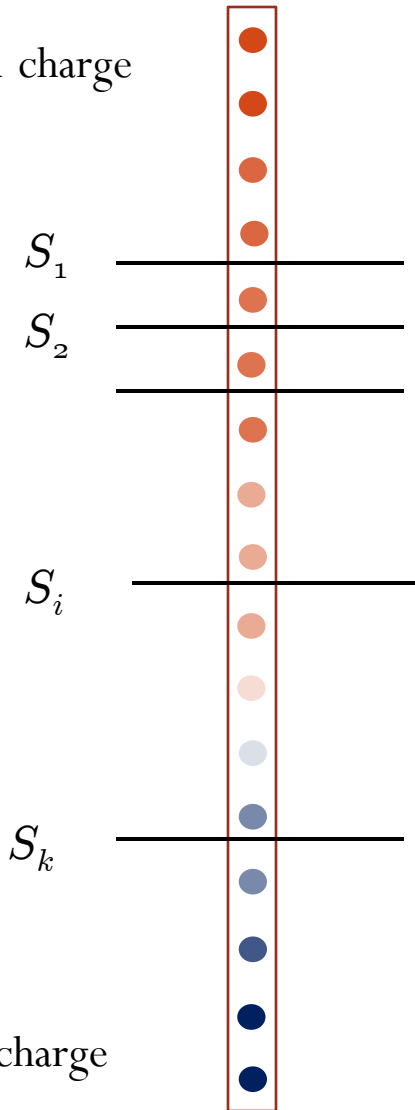
At iteration t , repeat $O(\log n)$ times:

- Consider G_t .
- Pick random assignment of ± 1 charge to the vertices.
- Mix charge along edges with rate η_t .
- Consider final distribution, sorted by charge.

Algorithm Description

- For each t , keep graph G_t and a rate η_t . $G_0 = G$ and $\eta_0 = 1$.

● = +1 charge



● = -1 charge

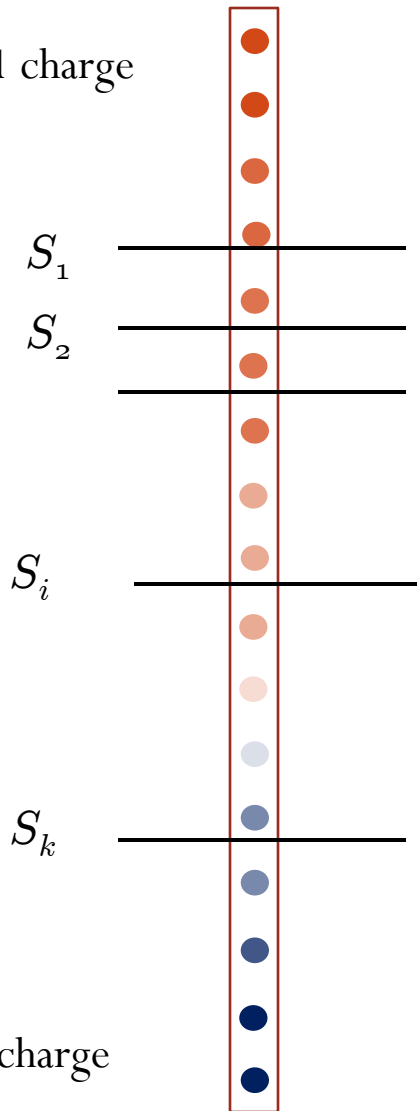
At iteration t , repeat $O(\log n)$ times:

- Consider G_t .
- Pick random assignment of ± 1 charge to the vertices.
- Mix along edges with rate η_t .
- Consider final distribution, sorted by charge.
- Check all $\Omega(b)$ -balanced sweep cuts S_1, \dots, S_k for $\phi(S_i) \leq O(\gamma^{1/2})$ in G .

Algorithm Description

- For each t , keep graph G_t and a rate η_t . $G_0 = G$ and $\eta_0 = 1$.

● = +1 charge



● = -1 charge

At iteration t , repeat $O(\log n)$ times:

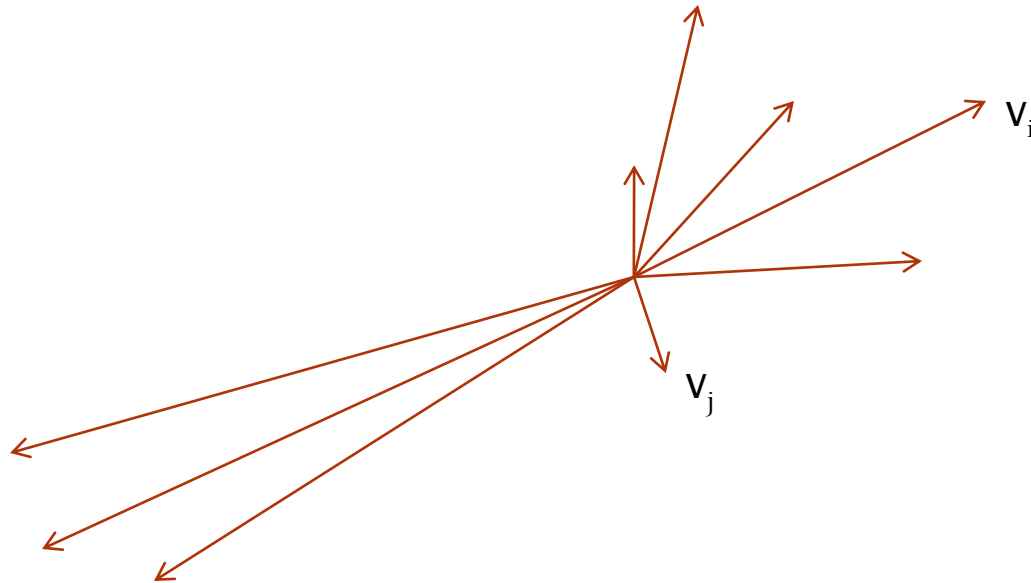
- Consider G_t .
- Pick random assignment of ± 1 charge to the vertices.
- Mix along edges with rate η_t .
- Consider final distribution, sorted by charge.
- Check all $\Omega(b)$ -balanced sweep cuts S_1, \dots, S_k for $\phi(S_i) \leq O(\gamma^{1/2})$ in G .

What if no sparse balanced cut is found?

Algorithm Description

- What if no sparse balanced cut is found?

Consider the $O(\log n)$ -dimensional vector embedding of vertices given by the final distributions.

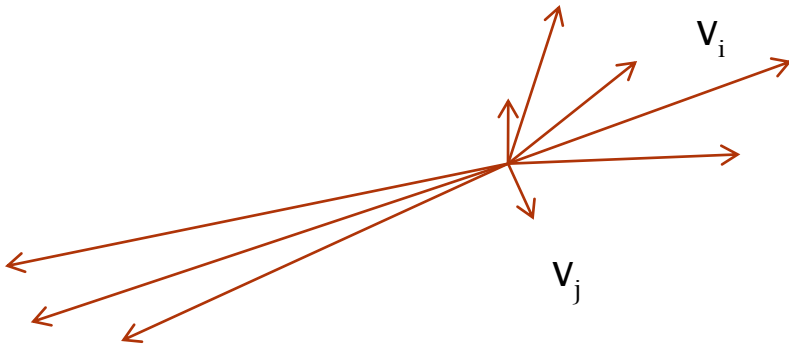


Algorithm Description

- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$



Algorithm Description

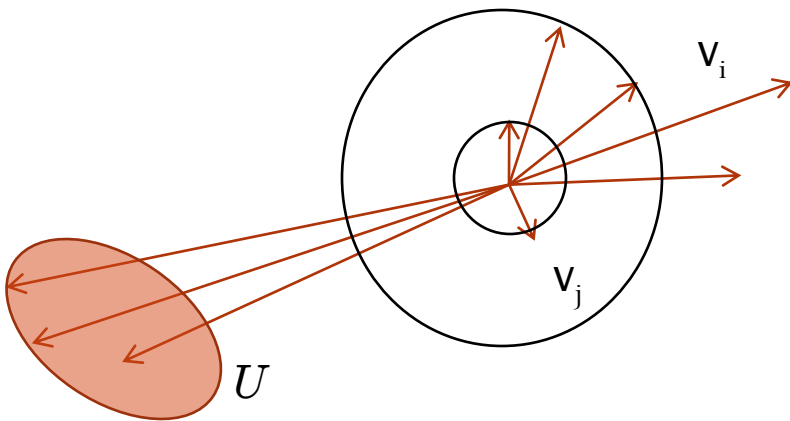
- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$

- Otherwise, grow ball to find an unbalanced cut U with

$$\phi(U) \leq O(\gamma^{1/2}).$$



Algorithm Description

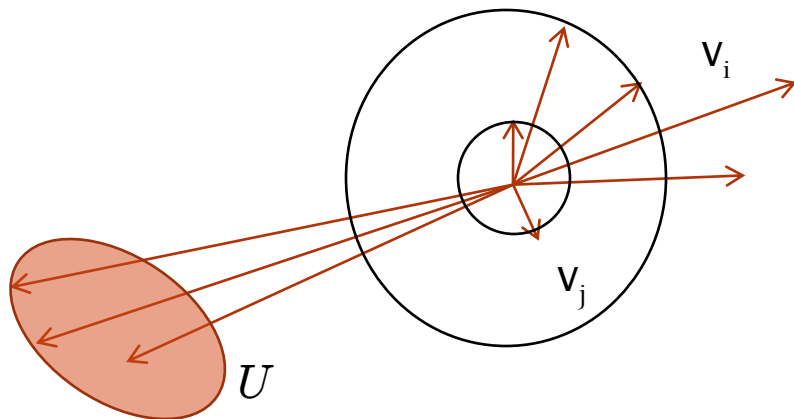
- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$

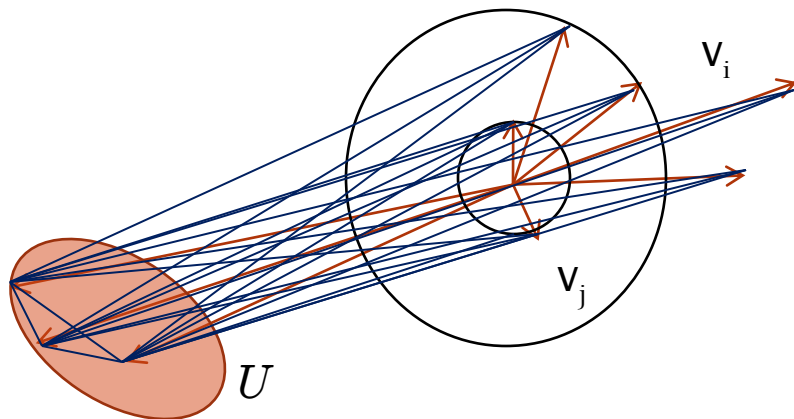
- Otherwise, grow ball to find an unbalanced cut U with

$$\phi(U) \leq O(\gamma^{1/2}).$$



Long vectors imply random walks got stuck in U

Algorithm Description



Long vectors imply random walks got stuck in U

- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$

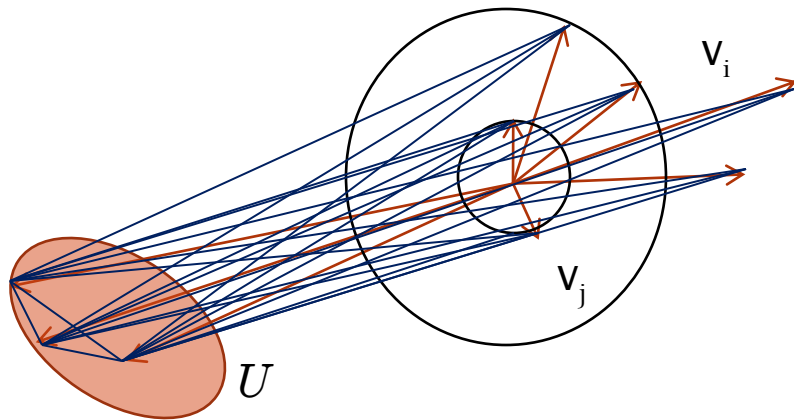
- Otherwise, grow ball to find an unbalanced cut U with

$$\phi(U) \leq O(\gamma^{1/2}).$$

How to fix it?

$$G_{t+1} = G_t + 2 \frac{\gamma}{n} \sum_{i \in U} \text{Star}_i$$

Algorithm Description



SEPARATION ORACLE

- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$

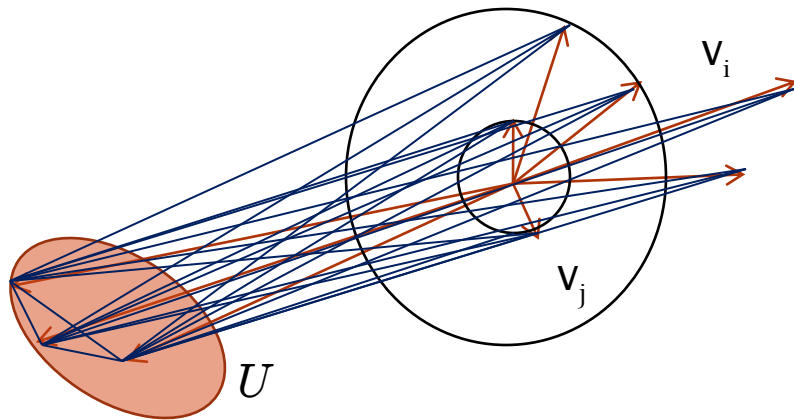
- Otherwise, grow ball to find an unbalanced cut U with

$$\phi(U) \leq O(\gamma^{1/2}).$$

How to fix it?

$$G_{t+1} = G_t + 2 \frac{\gamma}{n} \sum_{i \in U} \text{Star}_i$$

Algorithm Description



EXTRA GUARANTEE:

- Uses Cheeger's argument on radius
- Allows for sparse certificate cut

- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$

- Otherwise, grow ball to find an unbalanced cut U with

$$\phi(U) \leq O(\gamma^{1/2}).$$

How to fix it?

$$G_{t+1} = G_t + 2 \frac{\gamma}{n} \sum_{i \in U} \text{Star}_i$$

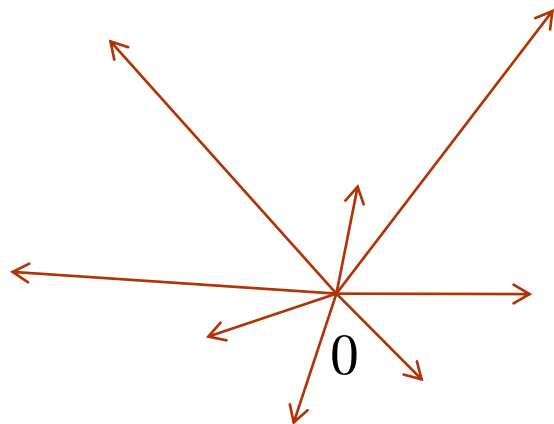
SDP Formulation

- Our algorithm is approximately solving this SDP

$$\mathbb{E}_{\{i,j\} \in E_G} \|v_i - v_j\|^2 \leq \gamma,$$

$$\mathbb{E}_{\{i,j\} \in V \times V} \|v_i - v_j\|^2 = \frac{1}{2m},$$

$$\forall i \in V \quad \mathbb{E}_{j \in V} \|v_i - v_j\|^2 = \frac{1}{c} \cdot \frac{1}{2m}.$$



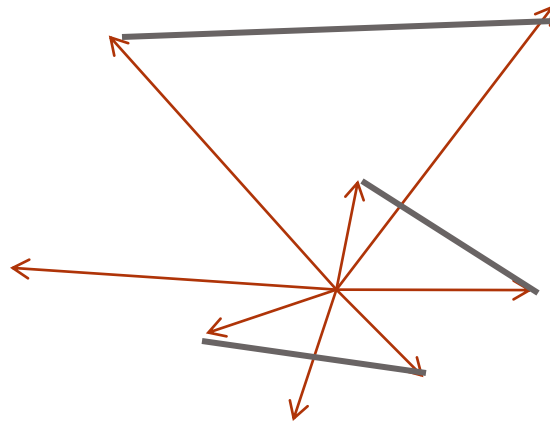
SDP Formulation

- Our algorithm is approximately solving this SDP

$$\mathbb{E}_{\{i,j\} \in E_G} \|v_i - v_j\|^2 \leq \gamma, \quad \text{SHORT EDGES}$$

$$\mathbb{E}_{\{i,j\} \in V \times V} \|v_i - v_j\|^2 = \frac{1}{2m}, \quad \text{FIXED VARIANCE}$$

$$\forall i \in V \quad \mathbb{E}_{j \in V} \|v_i - v_j\|^2 = \frac{1}{c} \cdot \frac{1}{2m}.$$



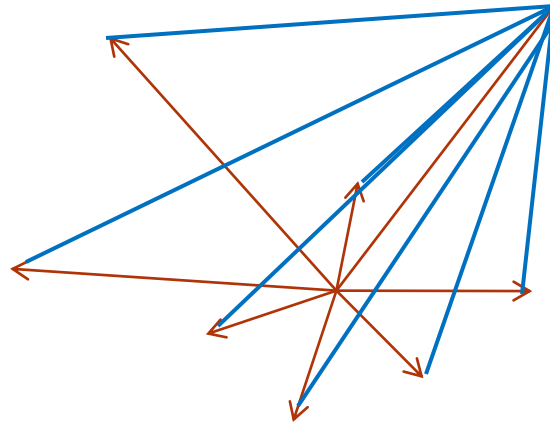
SDP Formulation

- Our algorithm is approximately solving this SDP

$$\mathbb{E}_{\{i,j\} \in E_G} \|v_i - v_j\|^2 \leq \gamma, \quad \text{SHORT EDGES}$$

$$\mathbb{E}_{\{i,j\} \in V \times V} \|v_i - v_j\|^2 = \frac{1}{2m}, \quad \text{FIXED VARIANCE}$$

$$\forall i \in V \quad \mathbb{E}_{j \in V} \|v_i - v_j\|^2 \leq \frac{1}{c} \cdot \frac{1}{2m}. \quad \text{SHORT RADIUS}$$



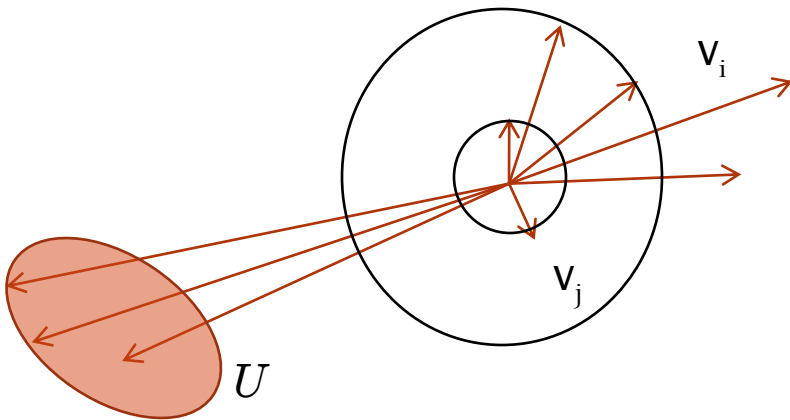
Oracle Description

BROKEN FIRST CONSTRAINT

- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$



BROKEN STAR CONSTRAINTS

- Otherwise, grow ball to find an unbalanced cut U with $\phi(U) \leq O(\gamma^{1/2})$.

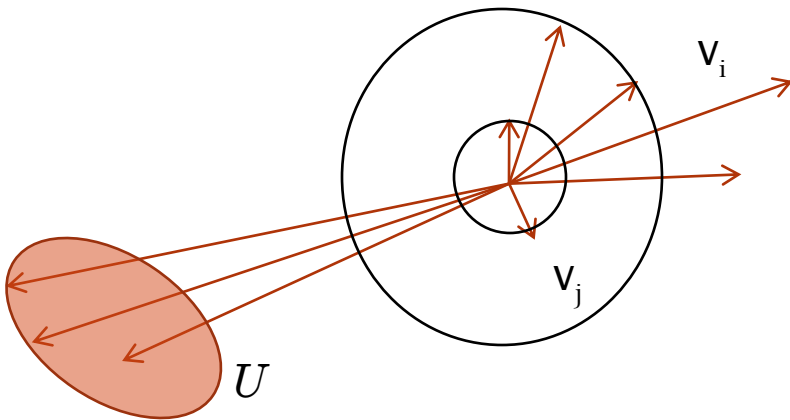
Oracle Description

BROKEN FIRST CONSTRAINT

- If $\sum_{\{i,j\} \in E} \|v_i - v_j\|^2 \geq \gamma \cdot \sum_{i \in V} \|v_i\|^2$

random walks may not have mixed enough. **How to fix it?**

Increase rate. $\eta_{t+1} = \eta_t + 1$



BROKEN STAR CONSTRAINTS

- Otherwise, grow ball to find an unbalanced cut U with

$$\phi(U) \leq O(\gamma^{1/2}).$$

EXTRA GUARANTEE

Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$

Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$

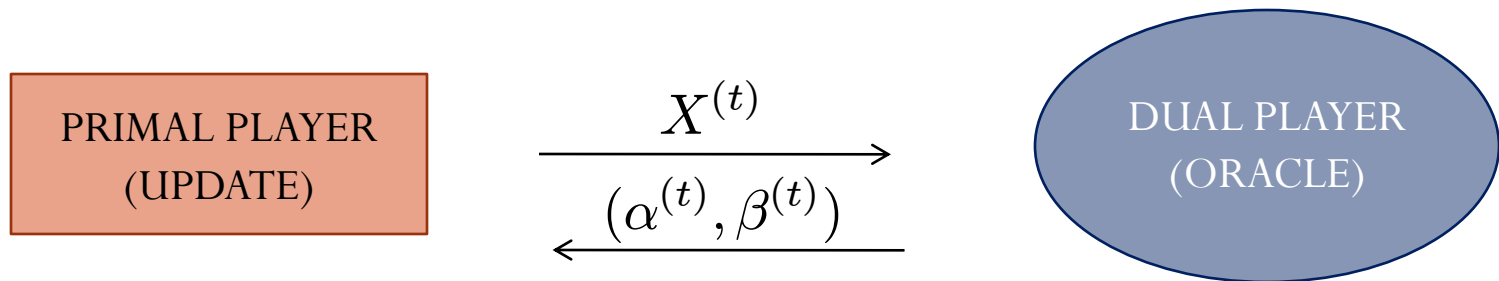
Add few stars to G to **push its conductance over γ**

Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$



Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$

- Given current dual solution

$$\alpha = \frac{1}{t} \sum_{i=0}^t \alpha^{(i)}, \quad \beta = \frac{1}{t} \sum_{i=0}^t \beta^{(i)}.$$

UPDATE:

$$X^{(t+1)} = Z \cdot e^{-t \left(L + \sum_{i \in V} \beta_i L(\text{Star}_i) - \alpha L(K_n) \right)}$$

Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$

- Given current dual solution

$$\alpha = \frac{1}{t} \sum_{i=0}^t \alpha^{(i)}, \quad \beta = \frac{1}{t} \sum_{i=0}^t \beta^{(i)}.$$

UPDATE:

$$X^{(t+1)} = Z \cdot e^{-t \left(L + \sum_{i \in V} \beta_i L(\text{Star}_i) - \alpha L(K_n) \right)}$$

Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$

- Given current dual solution

$$\alpha = \frac{1}{t} \sum_{i=0}^t \alpha^{(i)}, \quad \beta = \frac{1}{t} \sum_{i=0}^t \beta^{(i)}.$$

UPDATE:

$$X^{(t+1)} = Z' \cdot e^{-t \left(L + \sum_{i \in V} \beta_i L(\text{Star}_i) \right)}$$

Heat kernel random walk on modified graph G_t

Dual SDP and Random Walks

- Oracle produces tentative **dual** solutions

$$\alpha - \frac{1}{b} \sum_{i \in V} \beta_i \geq \gamma \cdot 2m,$$

$$L(G) + \sum_{i \in V} \beta_i L(\text{Star}_i) \succeq \alpha L(K_n).$$

- Given current dual solution

$$\alpha = \frac{1}{t} \sum_{i=0}^t \alpha^{(i)}, \quad \beta = \frac{1}{t} \sum_{i=0}^t \beta^{(i)}.$$

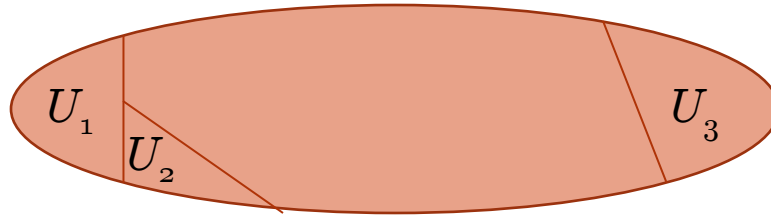
UPDATE:

$$X^{(t+1)} = Z' \cdot e^{-t \left(L + \sum_{i \in V} \beta_i L(\text{Star}_i) \right)}$$

Random seeds correspond to random projections

Algorithm – Termination

- Terminate if:
 - Sparse balanced cut is found,
 - Union of sparse unbalanced cuts found is balanced.



Algorithm – Termination

- Terminate if:
 - Sparse balanced cut is found,
 - Union of sparse unbalanced cuts found is balanced.
- If no sparse balanced cut found in T iterations and

$$T = O\left(\frac{\log n}{\gamma}\right),$$

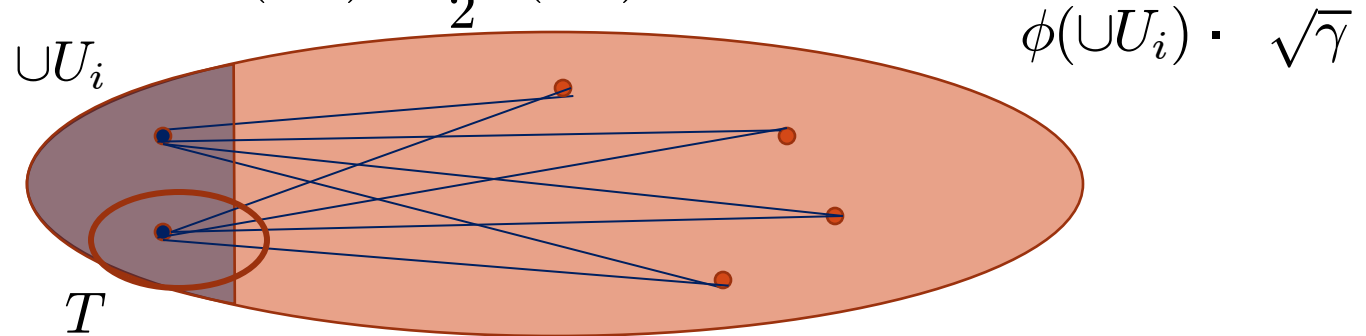
we can show that G_T is a **certificate** that G has **no** b -balanced cuts of conductance less than γ .

- Running Time

$$\tilde{O}(m) \times O\left(\frac{\log n}{\gamma}\right) = \tilde{O}\left(\frac{m}{\gamma}\right)$$

Our Algorithm - Certificate

- We can show that $L(G_T) \geq \frac{\gamma}{2}L(K_n)$



Any **sparse small cut** T must contain the root of many stars.

- T is well-correlated with $\cup U_i$

Suffices for producing decompositions

Final Remarks

Take-home messages

- SDP helps in design of fast algorithms, in particular spectral ones.
- Power of combining Matrix MWUs and spectral methods.

Research directions

- Apply this framework to other problems: **s-t maxflow**, **matching**.
- Extensions of Matrix MWU framework
- Empirical evaluations
- More applications of graph decompositions.

Final Remarks

Take-home messages

- SDP helps in design of fast algorithms, in particular spectral ones!
- Power of combining Matrix MWUs and spectral methods.

Research directions

- Apply this framework to other problems: **s-t maxflow**, **matching**.
- Extensions of Matrix MWU framework
- Empirical evaluations
- More applications of graph decompositions.

THANK YOU