A Nearly-linear-time Spectral Algorithm for Balanced Graph Partitioning

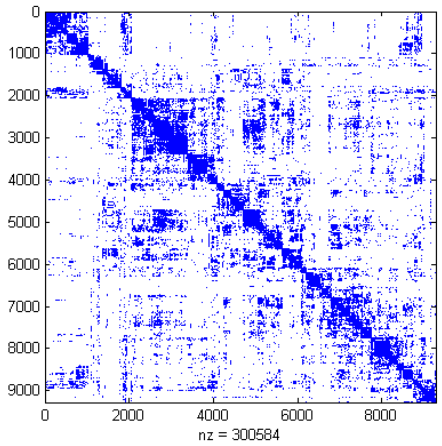
Lorenzo Orecchia
(MIT)

Joint work with
Nisheeth Vishnoi (MSR India) and Sushant Sachdeva (Princeton)

Spectral Algorithms for Graph Problems

What are they?

Spectral algorithms see the input graph as a linear operator



Adjacency matrix of a graph representing similarities between music artists.
Image by David Gleich.

Algorithmic primitives:

- matrix multiplication,
- eigenvector decomposition,
- Gaussian elimination

Spectral Algorithms for Graph Problems

Why are they useful …

… in practice?

- Simple to implement
- Can exploit very efficient linear algebra routines
- Perform provably well for many problems

… in theory?

- Many new interesting results based on spectral techniques:

  Laplacian Solvers, Sparsification, s-t Maxflow, etc.
- Geometric viewpoint
- Connections to Markov Chains and Probability Theory

Spectral Algorithms for Graph Partitioning

Spectral algorithms are widely used in many graph-partitioning applications:
clustering, image segmentation, community-detection, etc.

CLASSICAL VIEW:

- Based on Cheeger's Inequality

- Eigenvectors sweep-cuts reveal sparse cuts in the graph

NEW TREND:

- Random walk vectors replace eigenvectors:

  • Fast Algorithms for Graph Partitioning

  • Local Graph Partitioning

  • Real Network Analysis

- Different random walks: PageRank, Heat-Kernel, etc.

Why Random Walks? A Practitioner's View

Advantages of Random Walks:

1) Quick approximation to eigenvector in massive graphs

A= adjacency matrix

D = diagonal degree matrix

$W = AD^{-1}$= natural random walk matrix

$L = D - A$ = Laplacian matrix

Second Eigenvector of the Laplacian can be computed by iterating W:

For random y_0 s.t. $y_0^T D^{-1} 1 = 0$, compute

$$D^{-1} W^t y_0$$

Why Random Walks? A Practitioner's View

Advantages of Random Walks:

1) Quick approximation to eigenvector in massive graphs

A= adjacency matrix

D = diagonal degree matrix

$W = AD^{-1}$= natural random walk matrix

$L = D - A$ = Laplacian matrix

Second Eigenvector of the Laplacian can be computed by iterating W:

For random y_0 s.t. $y_0^T D^{-1} 1 = 0$, compute

$$D^{-1} W^t y_0$$

In the limit, $x_2(L) = \lim_{t \to \infty} \frac{D^{-1} W^t y_0}{||W^t y_0||_{D^{-1}}}$.

Why Random Walks? A Practitioner's View

Advantages of Random Walks:

1) Quick approximation to eigenvector in massive graphs

A= adjacency matrix

D = diagonal degree matrix

$W = AD^{-1}$= natural random walk matrix

$L = D - A$ = Laplacian matrix

Second Eigenvector of the Laplacian can be computed by iterating W:

For random y_0 s.t. $y_0^T D^{-1} 1 = 0$, compute

$$D^{-1} W^t y_0$$

In the limit, $x_2(L) = \lim_{t \to \infty} \frac{D^{-1} W^t y_0}{||W^t y_0||_{D^{-1}}}$.

Heuristic: For massive graphs, pick *t* as large as computationally affordable.

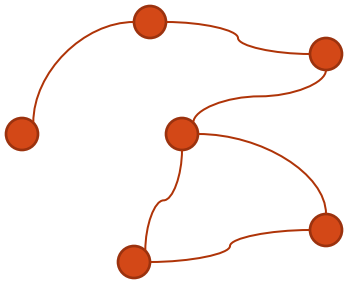
Why Random Walks? A Practitioner's View

Advantages of Random Walks:

1) Quick approximation to eigenvector in massive graphs

2) Statistically robust analogue of eigenvector

Real-world graphs are noisy



GROUND TRUTH GRAPH

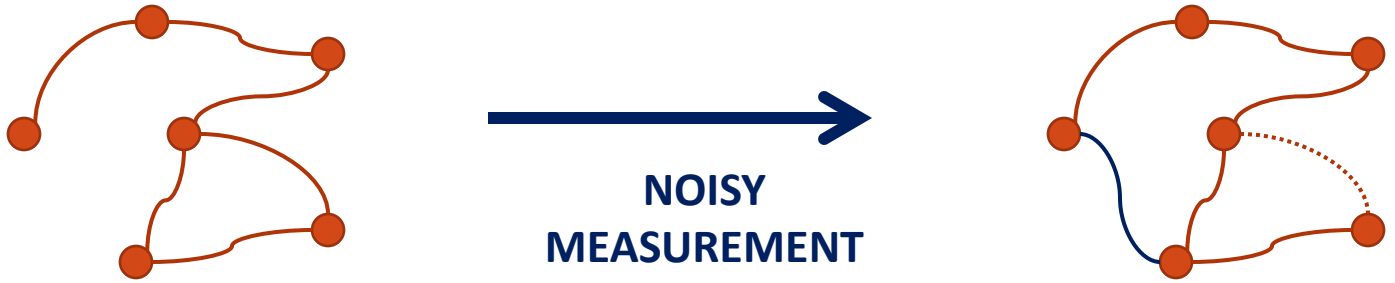
Why Random Walks? A Practitioner's View

Advantages of Random Walks:

1) Quick approximation to eigenvector in massive graphs

2) Statistically robust analogue of eigenvector

Real-world graphs are noisy



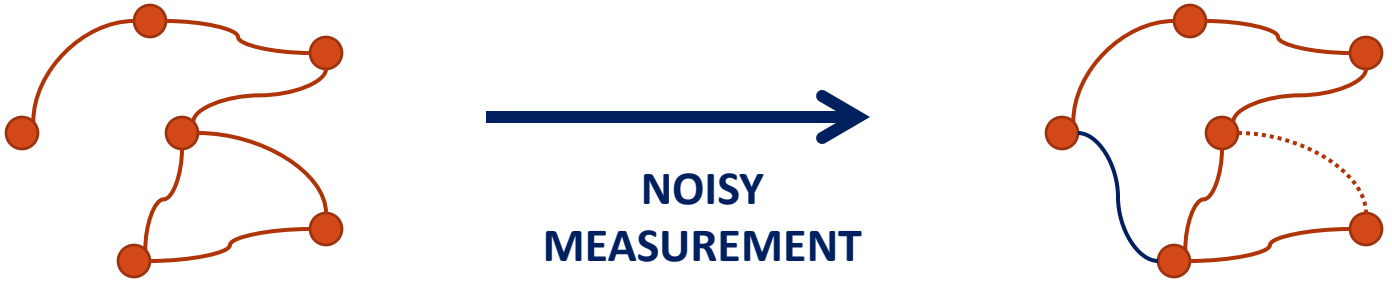
GROUND-TRUTH GRAPH **INPUT GRAPH**

Why Random Walks? A Practitioner's View

Advantages of Random Walks:

1) Quick approximation to eigenvector in massive graphs

2) Statistically robust analogue of eigenvector



GROUND-TRUTH GRAPH **INPUT GRAPH**

GOAL: estimate eigenvector of ground-truth graph.

OBSERVATION: eigenvector of input graph can have very large variance, as it can be very sensitive to noise

RANDOM-WALK VECTORS provide more stable estimates.

This Talk

QUESTION:

Why random-walk vectors in the design of fast algorithms?

This Talk

QUESTION:

Why random-walk vectors in the design of fast algorithms?

ANSWER: Stable, **regularized** version of the eigenvector

This Talk

QUESTION:

Why random-walk vectors in the design of fast algorithms?

ANSWER: Stable, **regularized** version of the eigenvector

GOALS OF THIS TALK:

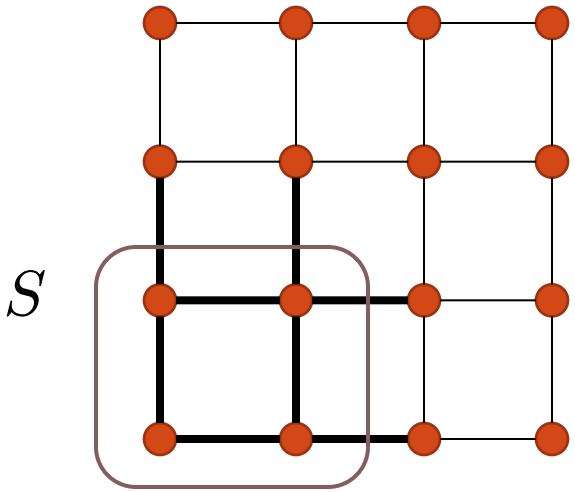
1) Showcase how these ideas apply to balanced graph partitioning

2) Show optimization perspective on why random walks arise

Problem Definition
and
Overview of Results

Partitioning Graphs - Conductance

Undirected unweighted $G = (V, E), |V| = n, |E| = m$

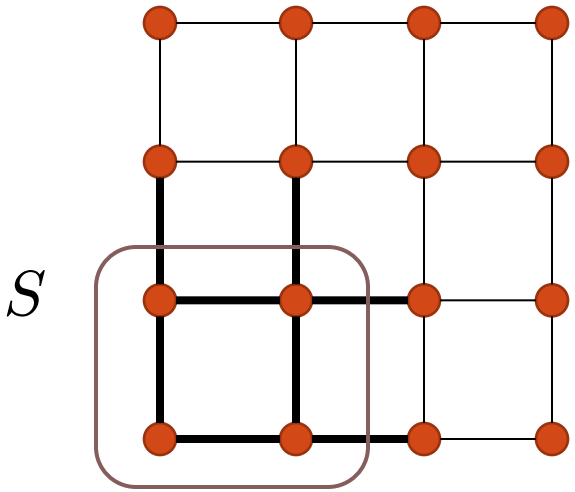


Conductance of $S \subseteq V$

$$\phi(S) = \frac{|E(S,\bar{S})|}{\min\{\mathrm{vol}(S),\mathrm{vol}(\bar{S})\}}$$

Partitioning Graphs - Conductance

Undirected unweighted $G = (V, E), |V| = n, |E| = m$



$\mathrm{vol}(S) = |E(S, V)| = 8$

$|E(S, \bar{S})| = 4$

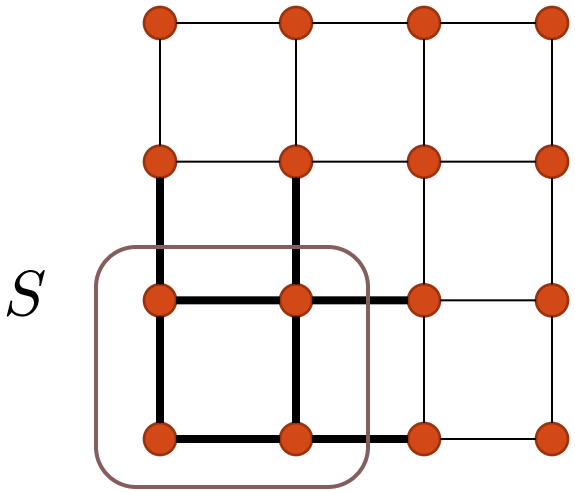
$\phi(S) = \frac{1}{2}$

Conductance of $S \subseteq V$

$$\phi(S) = \frac{|E(S,\bar{S})|}{\min\{\mathrm{vol}(S),\mathrm{vol}(\bar{S})\}}$$

Partitioning Graphs - Conductance

Undirected unweighted $G = (V, E), |V| = n, |E| = m$



$\text{vol}(S) = |E(S, V)| = 8$

$|E(S, \bar{S})| = 4$

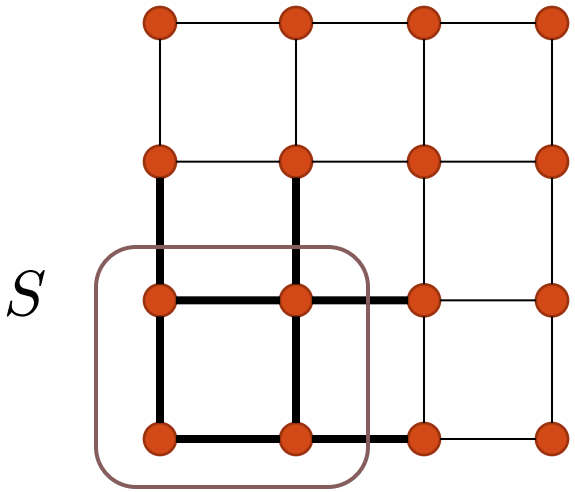
$\phi(S) = \frac{1}{2}$

Conductance of $S \subseteq V$

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

Partitioning Graphs - Conductance

Undirected unweighted $G = (V, E), |V| = n, |E| = m$



$\text{vol}(S) = |E(S, V)| = 8$

$|E(S, \bar{S})| = 4$

$\phi(S) = \frac{1}{2}$

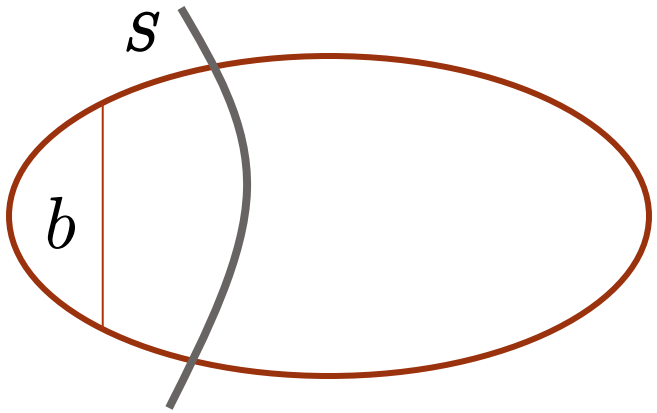
Conductance of $S \subseteq V$

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

Partitioning Graphs – Balanced Cut

NP-HARD DECISION PROBLEM

Does G have a b-balanced cut of conductance < γ ?



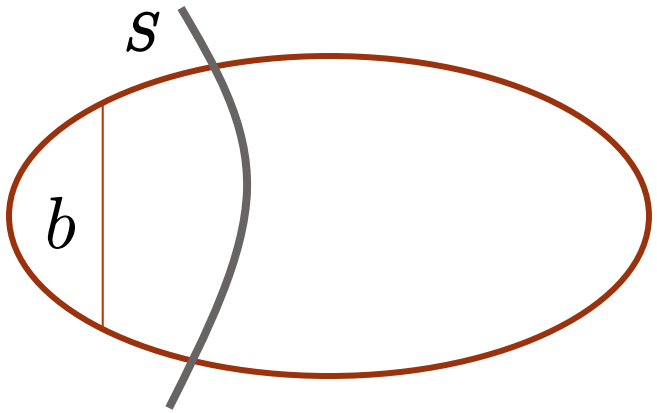
$$\phi(S) < \gamma$$

$$\frac{1}{2} > \frac{\mathrm{vol}(S)}{\mathrm{vol}(V)} > b$$

Partitioning Graphs – Balanced Cut

NP-HARD DECISION PROBLEM

Does G have a b-balanced cut of conductance < γ ?



$$\phi(S) < \gamma$$

$$\frac{1}{2} > \frac{\mathrm{vol}(S)}{\mathrm{vol}(V)} > b$$

- Important primitive for many **recursive** algorithms.
- Applications to **clustering** and **graph decomposition**.

Approximation Algorithms

NP-HARD DECISION PROBLEM

Does G have a b-balanced cut of conductance < γ ?

Algorithm	Method	Distinguishes $\geq \gamma$ and	Running Time
Recursive Eigenvector	Spectral	$O(\sqrt{\gamma})$	$\tilde{O}(mn)$
[Leighton, Rao '88]	Flow	$O(\gamma \log n)$	$\tilde{O}(m^{\frac{3}{2}})$ [AK'07, OSVV'08]
[Arora, Rao, Vazirani '04]	SDP (Flow + Spectral)	$O(\gamma\sqrt{\log n})$	$\tilde{O}(m^{\frac{3}{2}})$[Sherman '09]

Approximation Algorithms

DECISION PROBLEM

Does G have a b-balanced cut of conductance < γ ?

Algorithm	Method	Distinguishes $\geq \gamma$ and	Running Time
Recursive Eigenvector	Spectral	$O(\sqrt{\gamma})$	$\tilde{O}(mn)$
[Leighton, Rao '88]	Flow	$O(\gamma \log n)$	$\tilde{O}(m^{\frac{3}{2}})$ [AK'07, OSVV'08]
[Arora, Rao, Vazirani '04]	SDP (Flow + Spectral)	$O(\gamma\sqrt{\log n})$	$\tilde{O}(m^{\frac{3}{2}})$[Sherman '09]
[Madry '10]	SDP (Flow + Spectral)	$\text{polylog}_{\epsilon}(n)$	$\tilde{O}(m^{1+\epsilon})$

Fast Approximation Algorithms

DECISION PROBLEM

Does G have a b-balanced cut of conductance < γ ?

Algorithm	Method	Distinguishes $\geq \gamma$ and	Running Time
Recursive Eigenvector	Spectral	$O(\sqrt{\gamma})$	$\tilde{O}(mn)$
[Leighton, Rao '88]	Flow	$O(\gamma \log n)$	$\tilde{O}(m^{\frac{3}{2}})$ [AK'07, OSVV'08]
[Arora, Rao, Vazirani '04]	SDP (Flow + Spectral)	$O(\gamma\sqrt{\log n})$	$\tilde{O}(m^{\frac{3}{2}})$[Sherman '09]
[Madry '10]	SDP (Flow + Spectral)	$\mathrm{polylog}(n)$	$\tilde{O}(m^{1+\epsilon})$

Approximation Algorithms

Does G have a b-balanced cut of conductance < γ ?

Algorithm	Method	Distinguishes $\geq \gamma$ and	Running Time
Recursive Eigenvector	Spectral	$O(\sqrt{\gamma})$	$\tilde{O}(mn)$
[Spielman, Teng '04]	Local Random Walks	$O\left(\sqrt{\gamma \log^3 n}\right)$	$\tilde{O}\left(\frac{m}{\gamma^2}\right)$
[Andersen, Chung, Lang '07]	Local Random Walks	$O\left(\sqrt{\gamma \log n}\right)$	$\tilde{O}\left(\frac{m}{\gamma}\right)$
[Andersen, Peres '09]	Local Random Walks	$O\left(\sqrt{\gamma \log n}\right)$	$\tilde{O}\left(\frac{m}{\sqrt{\gamma}}\right)$
OUR ALGORITHM	Random Walks	$O(\sqrt{\gamma})$	$\tilde{O}(m)$

Approximation Algorithms

Does G have a b-balanced cut of conductance < γ ?

Algorithm	Method	Distinguishes $\geq \gamma$ and	Running Time
Recursive Eigenvector	Spectral	$O(\sqrt{\gamma})$	$\tilde{O}(mn)$
[Spielman, Teng '04]	Local Random Walks	$O\left(\sqrt{\gamma \log^3 n}\right)$	$\tilde{O}\left(\frac{m}{\gamma^2}\right)$
[Andersen, Chung, Lang '07]	Local Random Walks	$O\left(\sqrt{\gamma \log n}\right)$	$\tilde{O}\left(\frac{m}{\gamma}\right)$
[Andersen, Peres '09]	Local Random Walks	$O\left(\sqrt{\gamma \log n}\right)$	$\tilde{O}\left(\frac{m}{\sqrt{\gamma}}\right)$
OUR ALGORITHM	Random Walks	$O(\sqrt{\gamma})$	$\tilde{O}(m)$

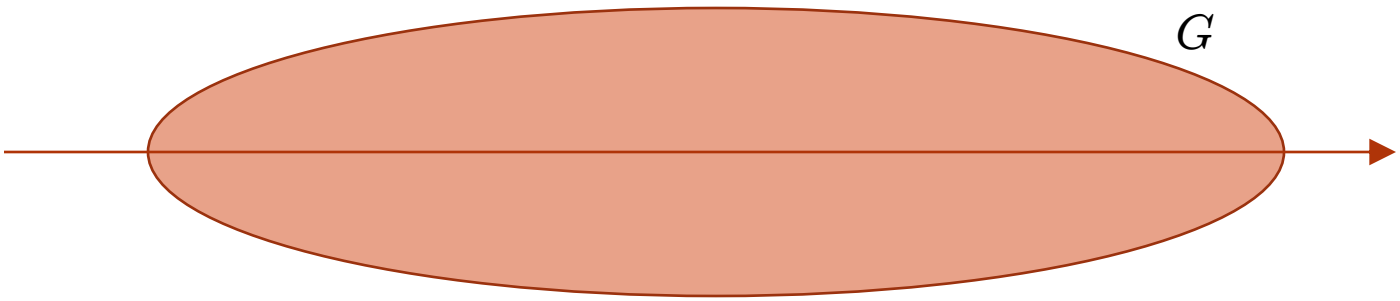
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2

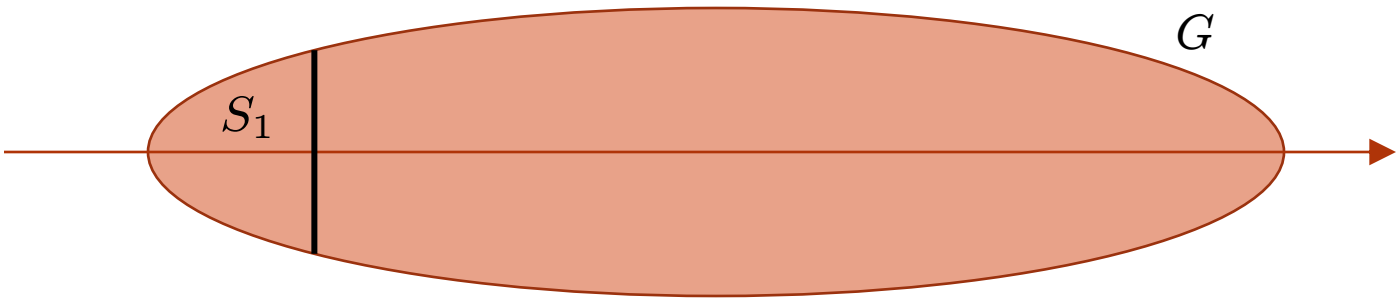


Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$



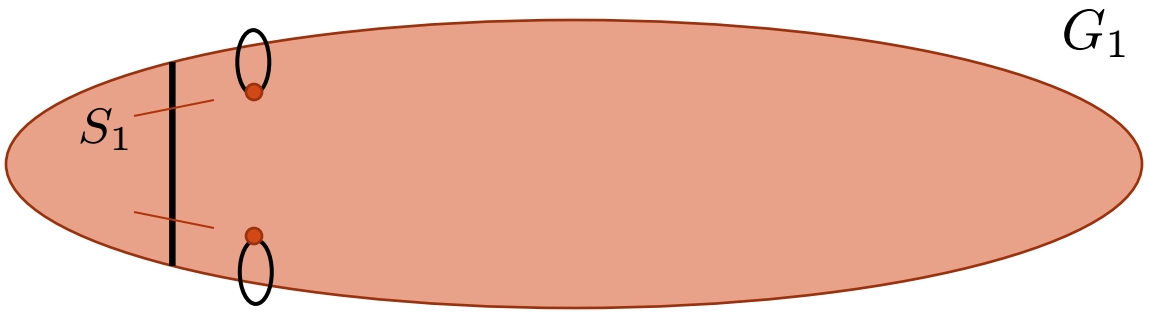
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.



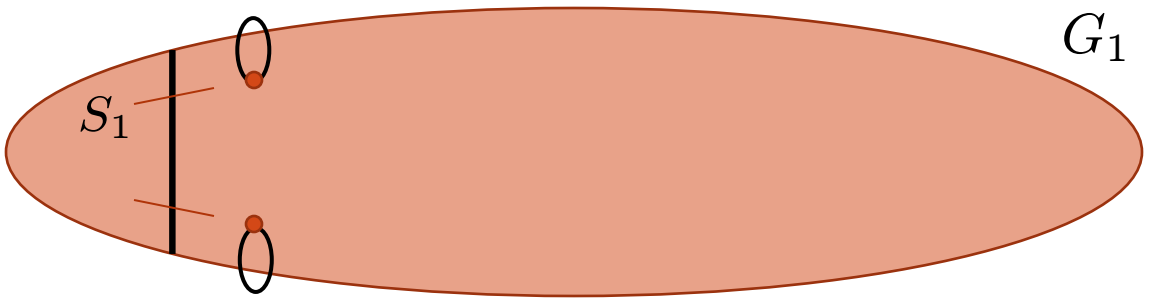
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.
- Recurse on G_1.



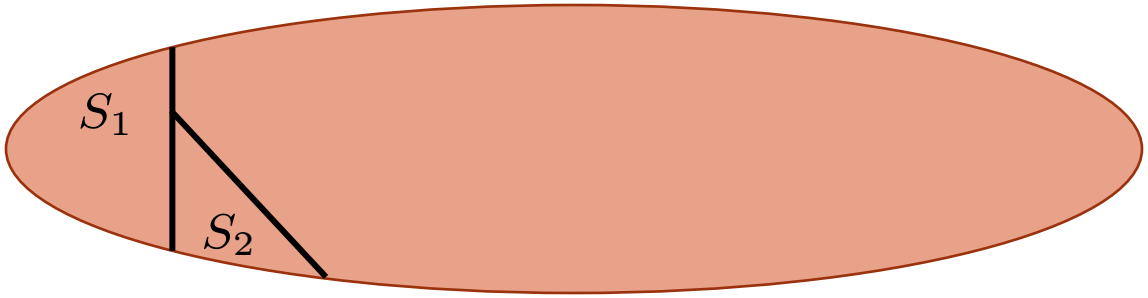
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.
- Recurse on G_1.



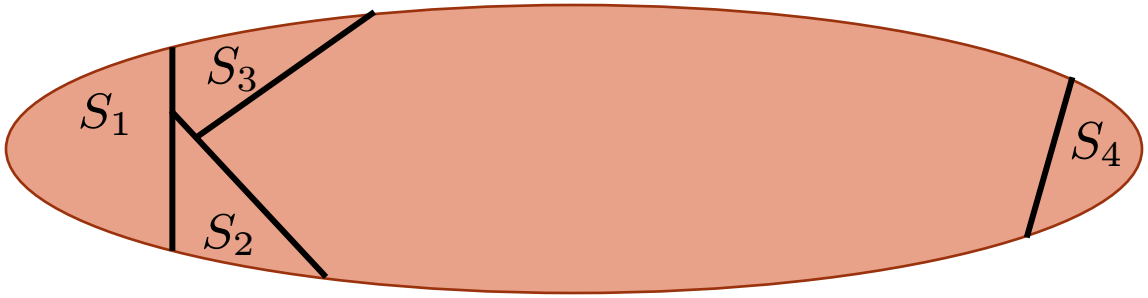
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.
- Recurse on G_1.



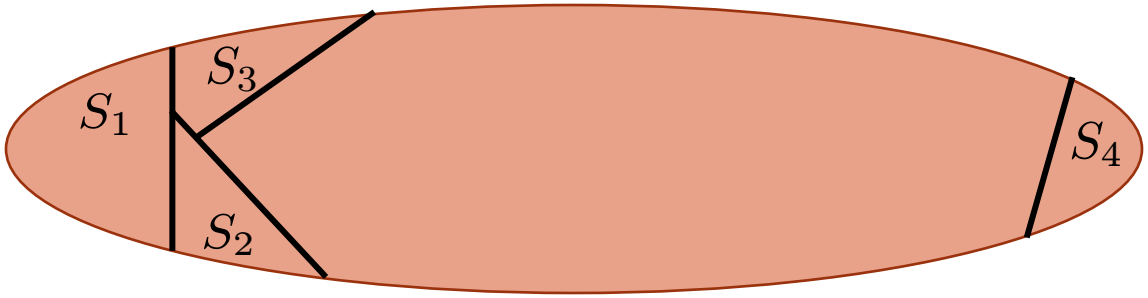
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.
- Recurse on G_1.
- If the cut $\cup S_i$ becomes $(b/2)$-balanced, output $\cup S_i$.



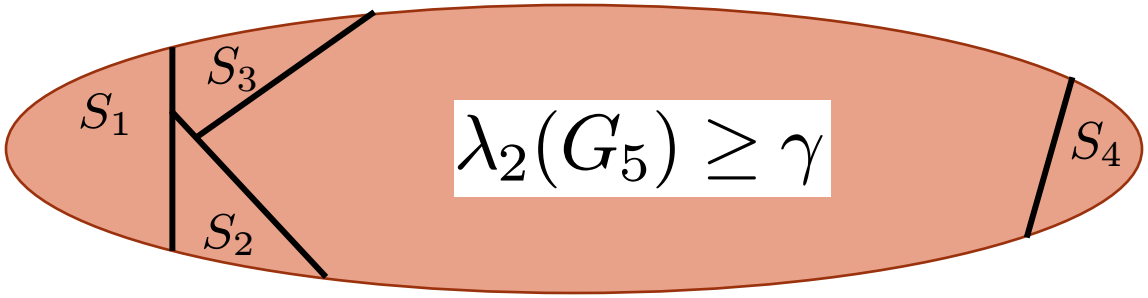
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.
- Recurse on G_1.
- If the cut $\cup S_i$ becomes $(b/2)$-balanced, output $\cup S_i$.



LARGE INDUCED EXPANDER = **NO-CERTIFICATE**

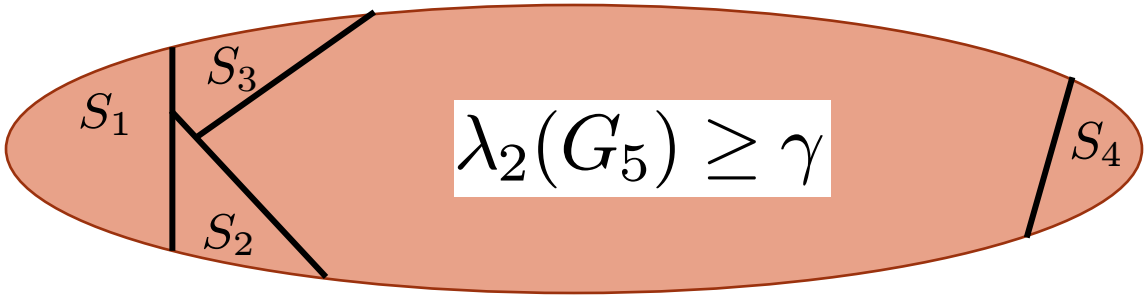
Recursive Eigenvector Algorithm

INPUT: (G, b, γ) **DECISION**: does there exists b-balanced S with $\phi(S) < \gamma$?

- Compute slowest mixing eigenvector of G and corresponding Laplacian eigenvalue λ_2
- If $\lambda_2 \geq \gamma$, output **NO**. Otherwise, sweep eigenvector to find S_1 such that

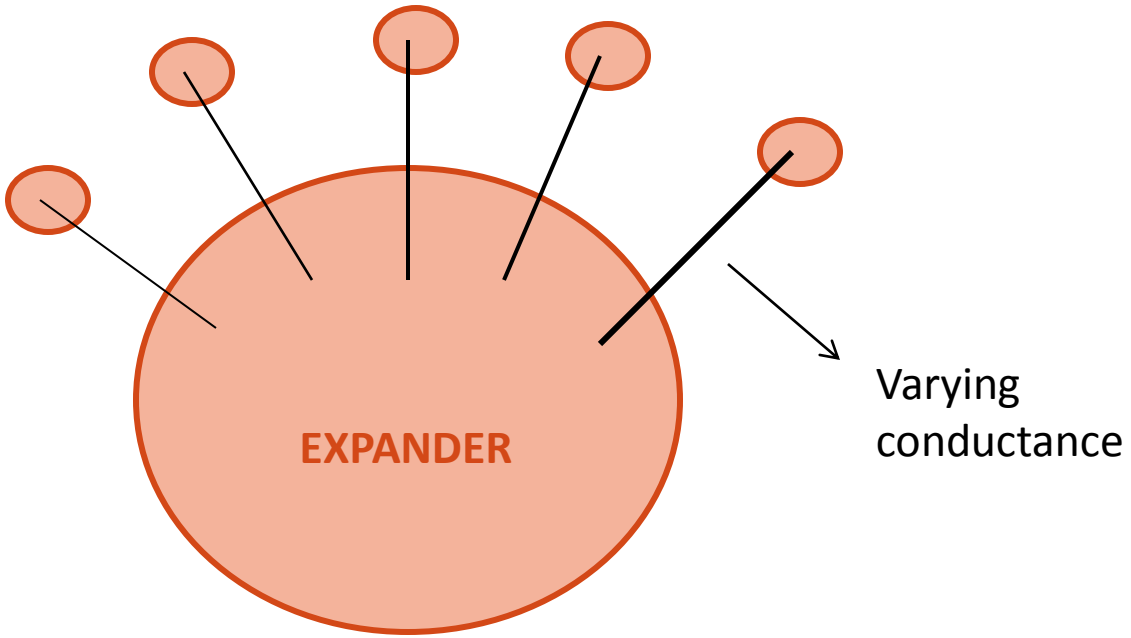
$$\phi(S_1) \cdot \quad O(\sqrt{\gamma}).$$

- If S_1 is $(b/2)$-balanced. Output S_1. Otherwise, consider the graph G_1 induced by G on V-S_1 with self-loops replacing the edges going to S_1.
- Recurse on G_1.
- If the cut $\cup S_i$ becomes $(b/2)$-balanced, output $\cup S_i$.



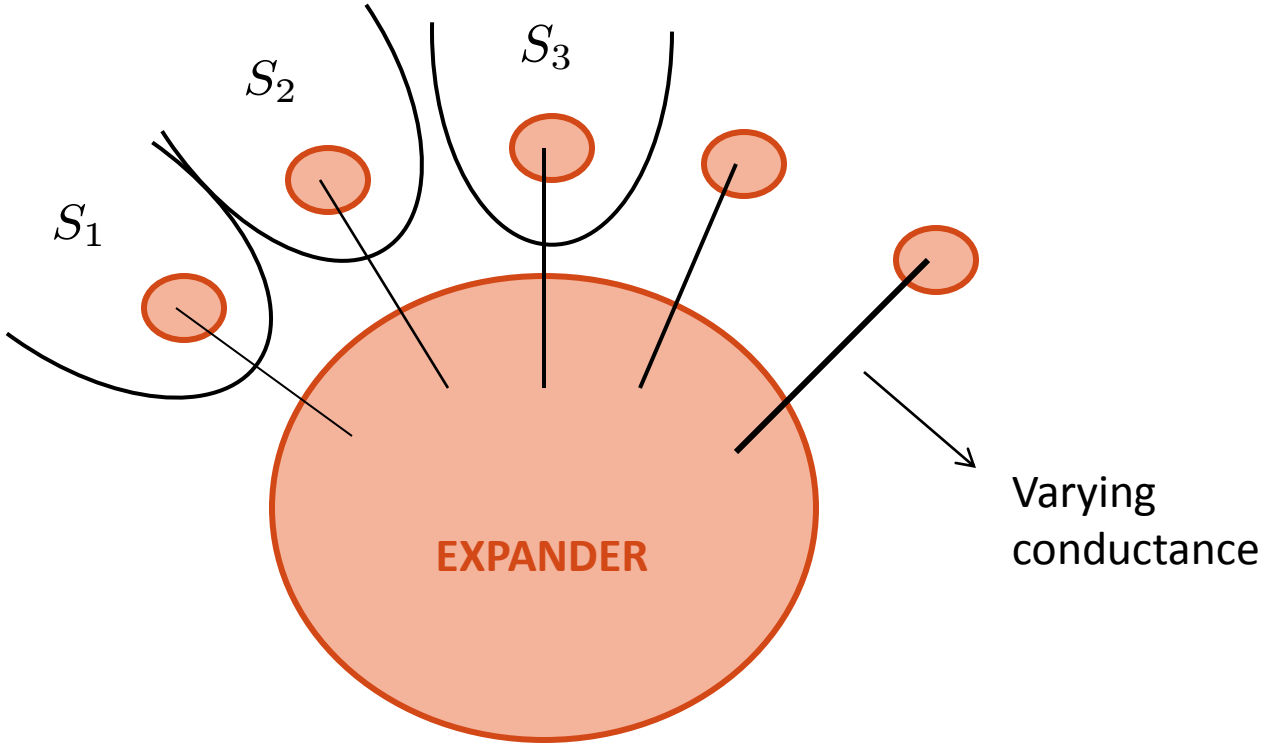
RUNNING TIME:$\tilde{O}(mn)$. $\tilde{O}(m)$ per iteration, $O(n)$ iterations.

Recursive Eigenvector: The Worst Case



- (n) nearly-disconnected components

Recursive Eigenvector: The Worst Case

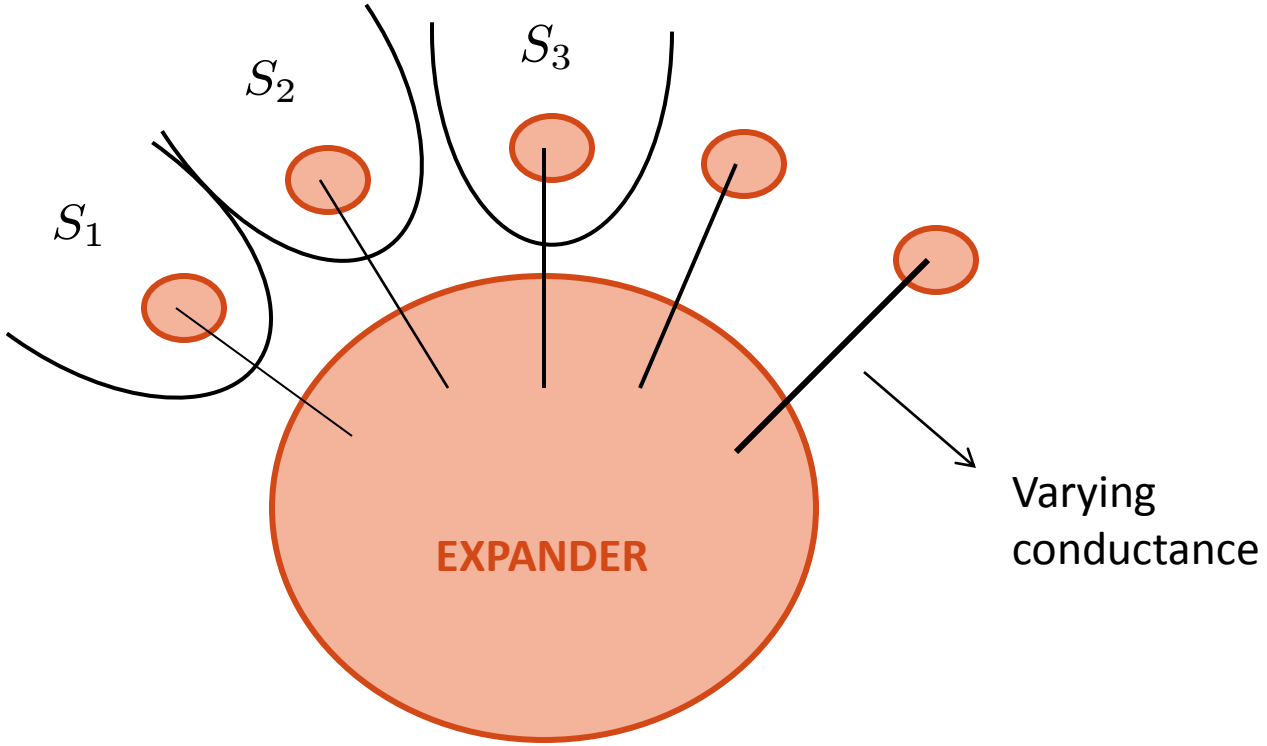


- (n) nearly-disconnected components

NB: Recursive Eigenvector eliminates one component per iteration.

- (n) iterations are necessary. Each iteration requires $\Omega(mn)$ time.

Recursive Eigenvector: The Worst Case



- (n) nearly-disconnected components

NB: Recursive Eigenvector eliminates one component per iteration.

- (n) iterations are necessary. Each iteration requires $\Omega(mn)$ time.

GOAL: Eliminate unbalanced low-conductance cuts faster.

Our Algorithm: Contributions

Algorithm	Method	Distinguishes $\geq \gamma$ and	Time
Recursive Eigenvector	Eigenvector	$O(\sqrt{\gamma})$	$\tilde{O}(mn)$
OUR ALGORITHM	Random Walks	$O(\sqrt{\gamma})$	$\tilde{O}(m)$

MAIN FEATURES:

- Compute $O(\log n)$ global heat-kernel random-walk vectors at each iteration
- Unbalanced cuts are removed in $O(\log n)$ iterations
- Method to compute heat-kernel vectors, i.e. $e^{-tL}v$, in time $\tilde{O}(m)$

TECHNICAL COMPONENTS:

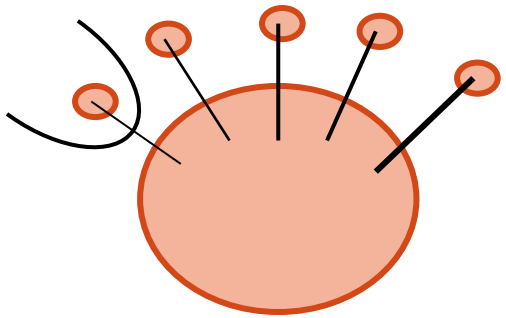
1) New iterative algorithm with a simple random walk interpretation

2) Novel analysis of Lanczos methods for computing heat-kernel vectors

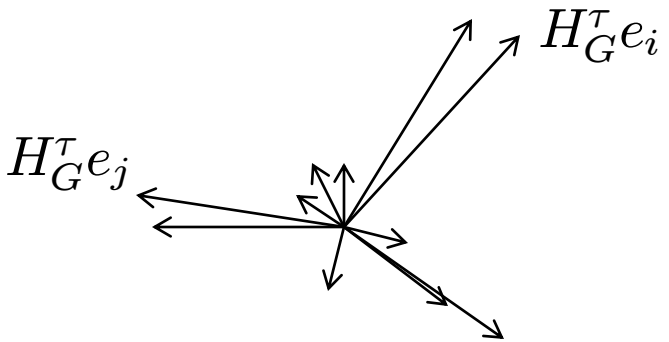
Why Random Walks

GOAL: Eliminate unbalanced cuts of low-conductance efficiently

- The graph eigenvector may be correlated with only one sparse unbalanced cut.



- Consider the Heat-Kernel random walk H_G^τ for $\tau = \log n/\gamma$.

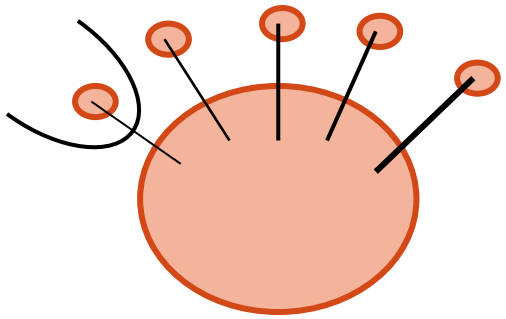


Probability vector for random walk started at vertex i

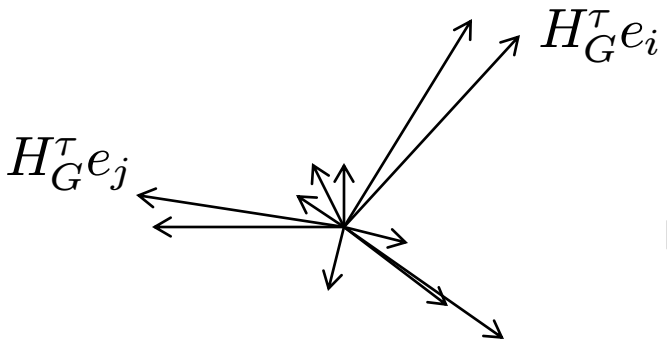
Why Random Walks

GOAL: Eliminate unbalanced cuts of low-conductance efficiently

- The graph eigenvector may be correlated with only one sparse unbalanced cut.



- Consider the Heat-Kernel random walk-matrix H_G^τ for $\tau = \log n/\gamma$.



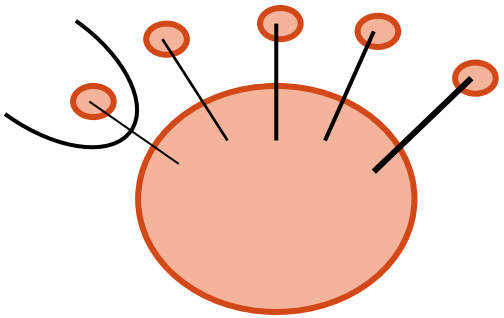
Probability vector for random walk started at vertex i

Long vectors are slow-mixing random walks

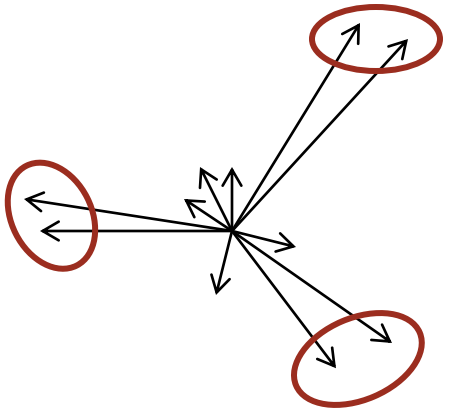
Why Random Walks

GOAL: Eliminate unbalanced cuts of low-conductance efficiently

- The graph eigenvector may be correlated with only one sparse unbalanced cut.



- Consider the Heat-Kernel random walk H_G^τ for $\tau = \log n/\gamma$.



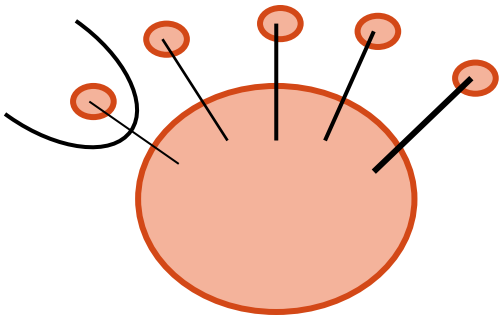
Unbalanced cuts of conductance $< \sqrt{\gamma}$

Why Random Walks

GOAL: Eliminate unbalanced cuts of low-conductance efficiently

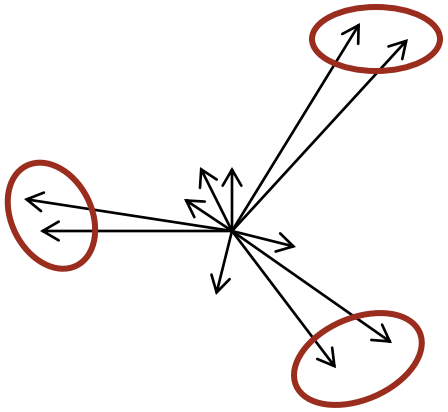
- The graph eigenvector may be correlated with only one sparse unbalanced cut.

SINGLE VECTOR
SINGLE CUT



- Consider the Heat-Kernel random walk H_G^τ for $\tau = \log n/\gamma$.

VECTOR EMBEDDING
MULTIPLE CUTS



Unbalanced cuts of conductance $< \sqrt{\gamma}$

Heat-Kernel Random Walk: A Regularization View

Background: Heat-Kernel Random Walk

For simplicity, take G to be **d-regular**.

• The Heat-Kernel Random Walk is a Continuous-Time Markov Chain over V, modeling the diffusion of heat along the edges of G.

• Transitions take place in continuous time t, with an exponential distribution.

$$\frac{\partial p(t)}{\partial t} = -L\frac{p(t)}{d}$$

$$p(t) = e^{-\frac{t}{d}L}p(0)$$

• The Heat Kernel can be interpreted as Poisson distribution over number of steps of the natural random walk $W{=}AD^{-1}$:

$$e^{-\frac{t}{d}L} = e^{-t}\sum_{k=1}^{\infty}\frac{t^k}{k!}W^k$$

Background: Heat-Kernel Random Walk

For simplicity, take G to be **d-regular**.

• The Heat-Kernel Random Walk is a Continuous-Time Markov Chain over V, modeling the diffusion of heat along the edges of G.

• Transitions take place in continuous time t, with an exponential distribution.

$$\frac{\partial p(t)}{\partial t} = -L\frac{p(t)}{d}$$

Notation

$$p(t) = e^{-\frac{t}{d}L}p(0) =: \boxed{H_G^t}\ p(0)$$

• The Heat Kernel can be interpreted as Poisson distribution over number of steps of the natural random walk $W{=}AD^{-1}$:

$$e^{-\frac{t}{d}L} = e^{-t}\sum_{k=1}^{\infty}\frac{t^k}{k!}W^k$$

Background: Heat-Kernel Random Walk

For simplicity, take G to be **d-regular**.

• The Heat-Kernel Random Walk is a Continuous-Time Markov Chain over V, modeling the diffusion of heat along the edges of G.

• Transitions take place in continuous time t, with an exponential distribution.

$$\frac{\partial p(t)}{\partial t} = -L\frac{p(t)}{d}$$

Notation

$$p(t) = e^{-\frac{t}{d}L}p(0) =: \boxed{H_G^t}\ p(0)$$

• The Heat Kernel can be interpreted as Poisson distribution over number of steps of the natural random walk $W{=}AD^{-1}$:

$$e^{-\frac{t}{d}L} = e^{-t}\sum_{k=1}^{\infty}\frac{t^k}{k!}W^k$$

• In practice, can replace Heat-Kernel with natural random walk W^t

Regularized Spectral Optimization

Eigenvector Program

$$\frac{1}{d} \min x^T L x$$
$$\text{s.t. } ||x||_2 = 1$$
$$x^T 1 = 0$$

$\longleftrightarrow$

SDP Formulation

$$\frac{1}{d} \min \quad L \bullet X$$
$$\text{s.t.} \quad I \bullet X = 1$$
$$11^T \bullet X = 0$$
$$X \succeq 0$$

Programs have same optimum. Take optimal solution

$$X^* = x^* (x^*)^T$$

Regularized Spectral Optimization

SDP Formulation

$$\frac{1}{d}\min \quad L \bullet X$$

$$\text{s.t.} \quad I \bullet X = 1$$

$$11^T \bullet X = 0$$

$$X \succeq 0$$

Density Matrix

Eigenvector decomposition of X:

$$X = \sum p_i v_i v_i^T$$

$$\forall i, p_i \geq 0,$$

$$\sum p_i = 1,$$

$$\forall i, v_i^T 1 = 0.$$

Eigenvalues of X define probability distribution

Regularized Spectral Optimization

SDP Formulation

$$\frac{1}{d} \min L \bullet X$$

$$\text{s.t.} \quad I \bullet X = 1$$

$$J \bullet X = 0$$

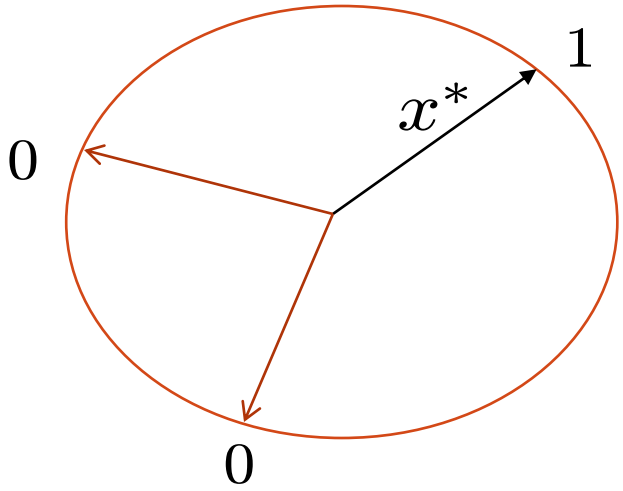
$$X \succeq 0$$

Density Matrix

Eigenvalues of X define probability distribution

$$X^* = x^*(x^*)^T$$

TRIVIAL DISTRIBUTION



Regularized Spectral Optimization

Regularized SDP

$$\frac{1}{d}\min \; L \bullet X + \boxed{\eta \cdot F(X)}$$

Regularizer F
Parameter η

$$\text{s.t.} \quad I \bullet X = 1$$

$$J \bullet X = 0$$

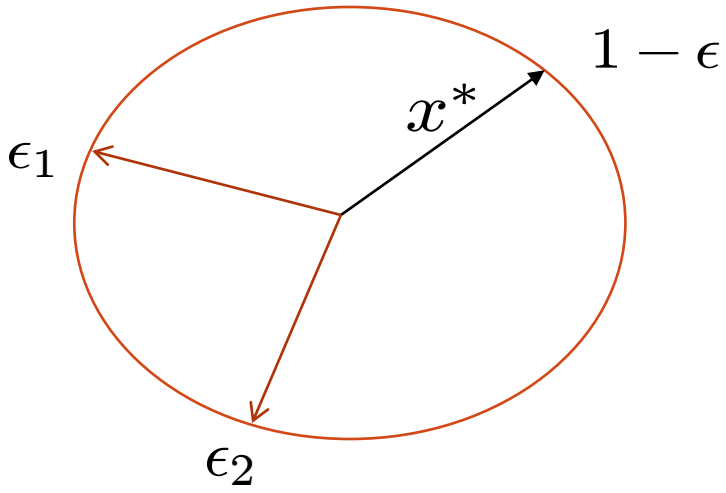
$$X \succeq 0$$

The regularizer F forces the distribution of eigenvalues of X to be non-trivial

$$X^* = x^*(x^*)^T$$

REGULARIZATION

$$X^* = \sum p_i v_i v_i^T$$



Regularizers

Regularizers are SDP-versions of common regularizers

- von Neumann Entropy

$$F_H(X) = \mathrm{Tr}(X \log X) = \sum p_i \log p_i$$

- p-Norm, $p>1$

$$F_p(X) = \frac{1}{p}||X||_p^p = \frac{1}{p}\mathrm{Tr}(X^p) = \frac{1}{p}\sum p_i^p$$

- And more, e.g. log-determinant.

Our Main Result

Regularized SDP

$$\frac{1}{d}\min \; L \bullet X + \eta \cdot F(X)$$
$$\text{s.t.} \quad I \bullet X = 1$$
$$J \bullet X = 0$$
$$X \succeq 0$$

RESULT:

Explicit correspondence between **regularizers and random walks**

REGULARIZER		OPTIMAL SOLUTION OF REGULARIZED PROGRAM	
$F = F_H$	Entropy →	$X^\star \propto H_t$	where t depends on η
			where α depends on η
$F = F_p$	p-Norm →	$X^\star \propto T_{q,\frac{1}{p-1}}$	where q depends on η

Discussion: Vector vs Density Matrix

Variable is density matrix, **not** vector

Q: Can we produce a single vector?

A: Density Matrix X describes distributions over vectors.

Assuming distribution is Gaussian, sample a vector

$$x = X^{\frac{1}{2}} u$$

where $u \sim N(0, I_{n-1})$

For example, the vector

$$x = H_{t/2} u$$

Random walk on random seed

is a sample from the solution of an entropy-regularized problem

PART 1
The New Iterative Procedure

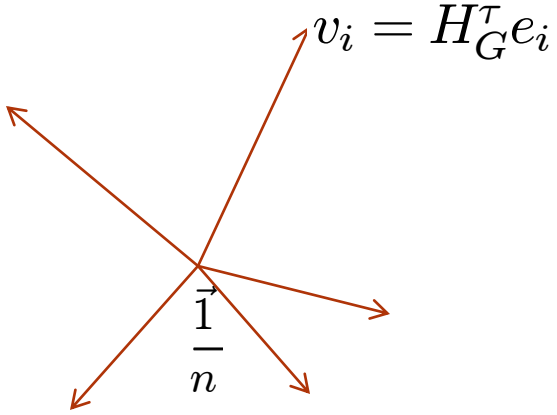
Our Algorithm for Balanced Cut

IDEA BEHIND OUR ALGORITHM:

Replace eigenvector in recursive eigenvector algorithm with

Heat-Kernel random walk H_G^τ for $\tau = \log n/\gamma$

Consider the embedding $\{v_i\}$ given by H_G^τ :



Our Algorithm for Balanced Cut

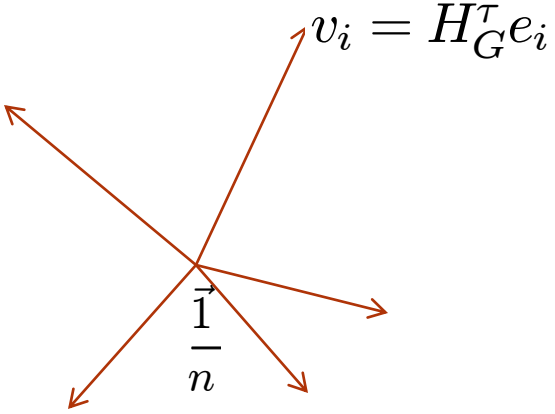
IDEA BEHIND OUR ALGORITHM:

Replace eigenvector in recursive eigenvector algorithm with

Heat-Kernel random walk H_G^τ for $\tau = \log n/\gamma$

Chosen to emphasize cuts of conductance $\approx\gamma$

Consider the embedding $\{v_i\}$ given by H_G^τ :



Stationary distribution is uniform as G is regular

Our Algorithm for Balanced Cut

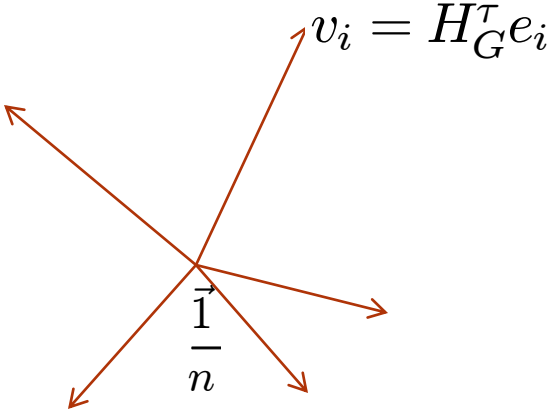
IDEA BEHIND OUR ALGORITHM:

Replace eigenvector in recursive eigenvector algorithm with

Heat-Kernel random walk H_G^τ for $\tau = \log n/\gamma$

Chosen to emphasize cuts of conductance $\approx\gamma$

Consider the embedding $\{v_i\}$ given by H_G^τ :



Stationary distribution is uniform as G is regular

MIXING:

Define the total deviation from stationary for a set $S \subseteq V$ for walk H_G^τ

$$\Psi(H_G^\tau, S) = \sum_{i \in S} ||v_i - \vec{1}/n||^2$$

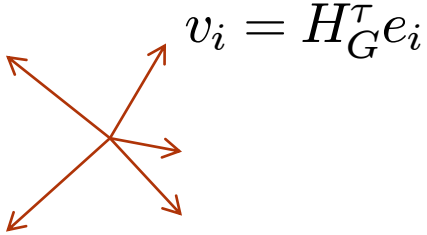
FUNDAMENTAL QUANTITY TO UNDERSTAND CUTS IN G

Our Algorithm: Case Analysis

Recall:

$$\tau = \log n/\gamma \qquad \Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$$

CASE 1: Random walks have **mixed**



ALL VECTORS ARE SHORT

$$\Psi(H_G^\tau, V) \cdot \frac{1}{\text{poly}(n)}$$

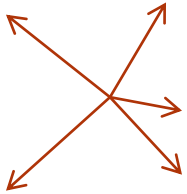
Our Algorithm: Case Analysis

Recall:

$$\tau = \log n/\gamma \qquad \Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$$

CASE 1: Random walks have **mixed**

$$v_i = H_G^\tau e_i$$



ALL VECTORS ARE SHORT

$$\Psi(H_G^\tau, V) \cdot \frac{1}{\text{poly}(n)}$$

By definition of τ

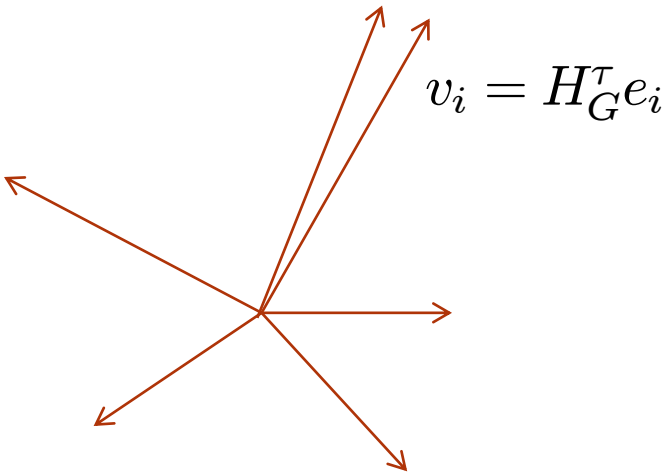
$$\lambda_2 \geq - \ (\gamma)$$

$$\phi_G \geq - \ (\gamma)$$

Our Algorithm

$\tau = \log n/\gamma$

$\Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$



CASE 2: Random walks have **not mixed**

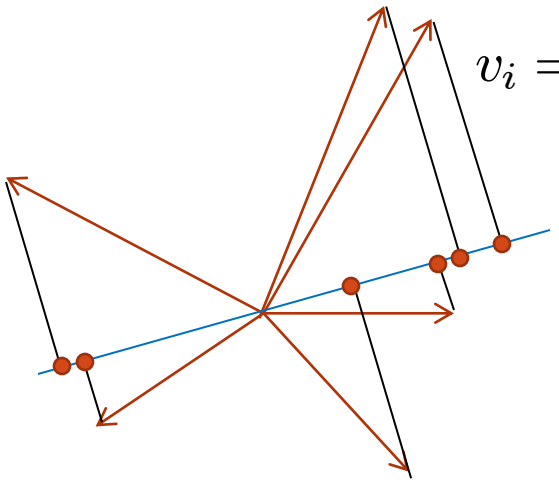
$$\Psi(H_G^\tau, V) > \frac{1}{\text{poly}(n)}$$

We can either find an $\Omega(b)$-balanced cut with conductance $O(\sqrt{\gamma})$

Our Algorithm

$\tau = \log n/\gamma$

$\Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$



$v_i = H_G^\tau e_i$

RANDOM PROJECTION
+
SWEEP CUT

CASE 2: Random walks have **not mixed**

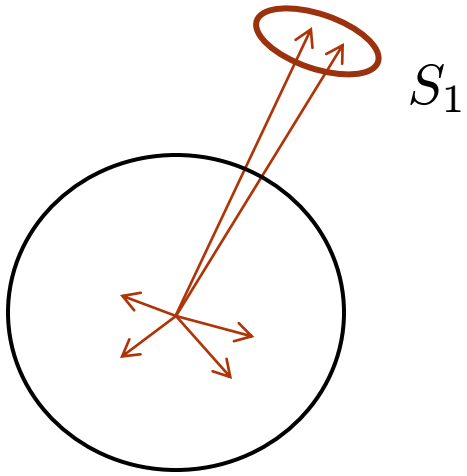
$$\Psi(H_G^\tau, V) > \frac{1}{\text{poly}(n)}$$

We can either find an $\Omega(b)$-balanced cut with conductance $O(\sqrt{\gamma})$

Our Algorithm

$\tau = \log n/\gamma$

$\Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$



BALL ROUNDING

CASE 2: Random walks have **not mixed**

$$\Psi(H_G^\tau, V) > \frac{1}{\text{poly}(n)}$$

We can either find an $\Omega(b)$-balanced cut with conductance · $O(\sqrt{\gamma})$

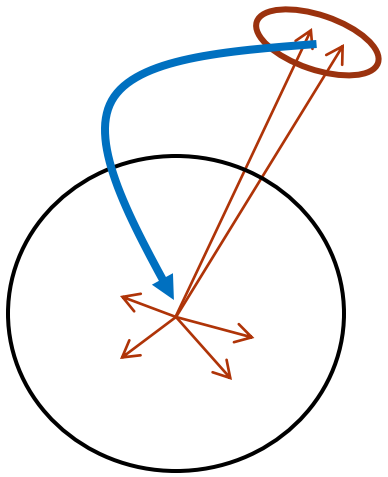
OR a ball cut yields S_1 such that $\phi(S_1)$ · $O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Our Algorithm: Iteration

$\tau = \log n/\gamma$

$\Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$



CASE 2: We found an unbalanced cut S_1 with $\phi(S_1) \cdot O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Modify G=$G^{(1)}$ by **adding edges** across $(S_1, \bar{S_1})$ to construct $G^{(2)}$.

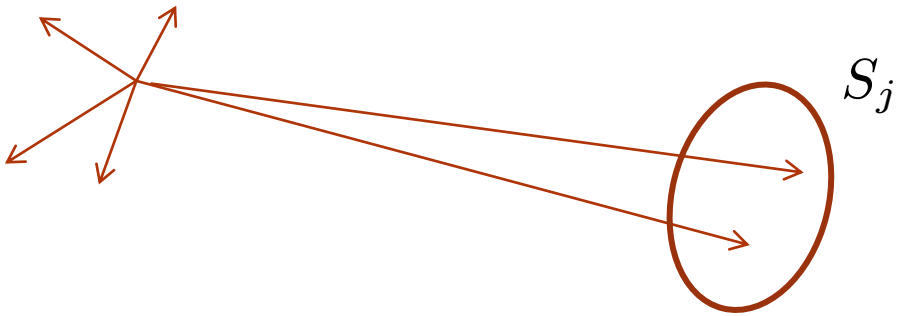
Analogous to removing unbalanced cut S_1
in Recursive Eigenvector algorithm

Our Algorithm: Modifying G

CASE 2: We found an unbalanced cut S_1 with $\phi(S_1) \cdot \quad O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Modify G=$G^{(1)}$ by **adding edges** across $(S_1, \bar{S}_1)$ to construct $G^{(2)}$.

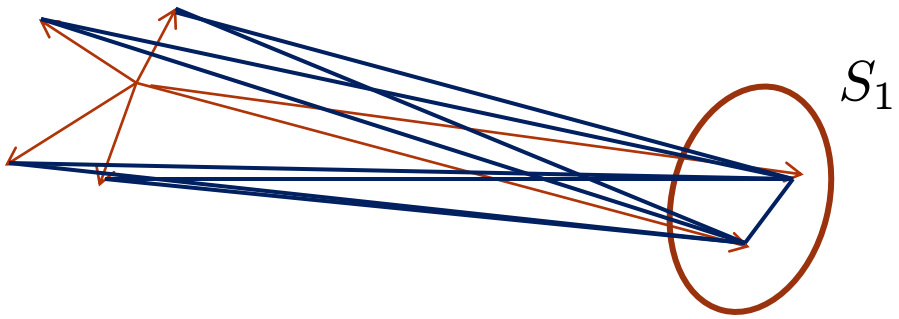


Our Algorithm: Modifying G

CASE 2: We found an unbalanced cut S_1 with $\phi(S_1) \cdot \quad O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Modify G=$G^{(1)}$ by **adding edges** across $(S_1, \bar{S}_1)$ to construct $G^{(2)}$.



$$G^{(t+1)} = G^{(t)} + \gamma \sum_{i \in S_t} \mathrm{Star}_i$$

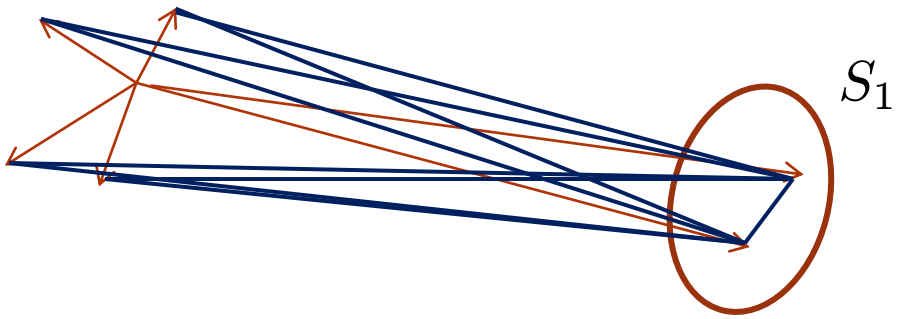
where Star_i is the star graph rooted at vertex i.

Our Algorithm: Modifying G

CASE 2: We found an unbalanced cut S_1 with $\phi(S_1) \cdot \quad O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Modify G=$G^{(1)}$ by **adding edges** across $(S_1, \bar{S}_1)$ to construct $G^{(2)}$.



$$G^{(t+1)} = G^{(t)} + \gamma \sum_{i \in S_t} \mathrm{Star}_i$$

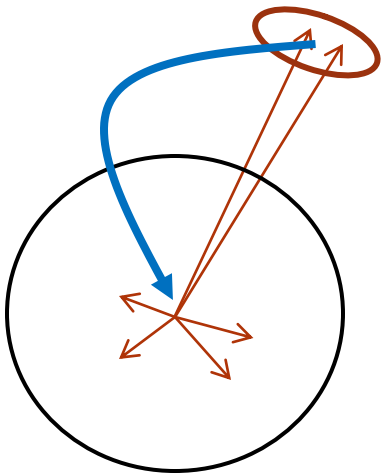
where Star_i is the star graph rooted at vertex i.

The random walk can now escape S_1 more easily.

Our Algorithm: Iteration

$\tau = \log n/\gamma$

$\Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$



CASE 2: We found an unbalanced cut S_1 with $\phi(S_1) \cdot O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Modify G=$G^{(1)}$ by **adding edges** across $(S_1, \bar{S}_1)$ to construct $G^{(2)}$.

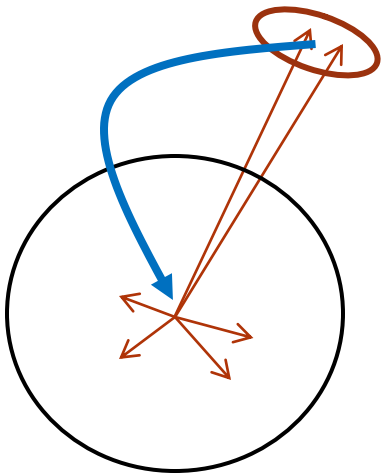
POTENTIAL REDUCTION:

$$\Psi(H^\tau_{G^{(t+1)}}, V) \cdot \Psi(H^\tau_{G^{(t)}}, V) - \tfrac{1}{2}\Psi(H^\tau_{G^{(t)}}, S_t) \cdot \tfrac{3}{4}\Psi(H^\tau_{G^{(t)}}, V)$$

Our Algorithm: Iteration

$\tau = \log n/\gamma$

$\Psi(H_G^\tau, S) = \sum_{i \in S} ||H_G^\tau e_i - \vec{1}/n||^2$



CASE 2: We found an unbalanced cut S_1 with $\phi(S_1) \cdot O(\sqrt{\phi})$ and

$$\Psi(H_G^\tau, S_1) \geq \tfrac{1}{2}\Psi(H_G^\tau, V).$$

Modify G=$G^{(1)}$ by **adding edges** across $(S_1, \bar{S_1})$ to construct $G^{(2)}$.

POTENTIAL REDUCTION:

$$\Psi(H^\tau_{G^{(t+1)}}, V) \cdot \Psi(H^\tau_{G^{(t)}}, V) - \tfrac{1}{2}\Psi(H^\tau_{G^{(t)}}, S_t) \cdot \tfrac{3}{4}\Psi(H^\tau_{G^{(t)}}, V)$$

CRUCIAL APPLICATION OF STABILITY OF RANDOM WALK

Summary and Potential Analysis

IN SUMMARY:

At every step t of the recursion, we either

1. Produce a $\Omega(b)$-balanced cut of the required conductance, OR

Potential Reduction

IN SUMMARY:

At every step t of the recursion, we either

1. Produce a $\Omega(b)$-balanced cut of the required conductance, OR
2. Find that

$$\Psi(H_G^\tau, V) \cdot \frac{1}{\text{poly}(n)}$$, OR

Potential Reduction

IN SUMMARY:

At every step t of the recursion, we either

1. Produce a $\Omega(b)$-balanced cut of the required conductance, OR
2. Find that

$$\Psi(H^{\tau}_{G}, V) \cdot \frac{1}{\mathrm{poly}(n)}$$, OR

3. Find an unbalanced cut S_t of the required conductance,
such that for the graph $G^{(t+1)}$, modified to have increased edges from S_t,

$$\boxed{\Psi(H^{\tau}_{G^{(t+1)}}, V) \cdot \frac{3}{4}\Psi(H^{\tau}_{G^{(t)}}, V)}$$

Potential Reduction

IN SUMMARY:

At every step t-1 of the recursion, we either

1. Produce a $\Omega(b)$-balanced cut of the required conductance, OR
2. Find that

$$\Psi(H_G^\tau, V) \cdot \frac{1}{\mathrm{poly}(n)}$$, OR

3. Find an unbalanced cut S_t of the required conductance, such that for the process $P^{(t+1)}$, modified to have increased transitions from S_t,

$$\Psi(H_{G^{(t+1)}}^\tau, V) \cdot \frac{3}{4}\Psi(H_{G^{(t)}}^\tau, V)$$

After $T{=}O(\log n)$ iterations, if no low-conductance $\Omega(b)$-balanced cut is found:

$$\Psi(H_{G^{(T)}}^\tau, V) \cdot \frac{1}{\mathrm{poly}(n)}$$

From this guarantee, using the definition of $G^{(T)}$, we derive an SDP-certificate that no b-balanced cut of conductance $O(\gamma)$ exists in G.

NB: Only O(log n) iterations to remove unbalanced cuts.

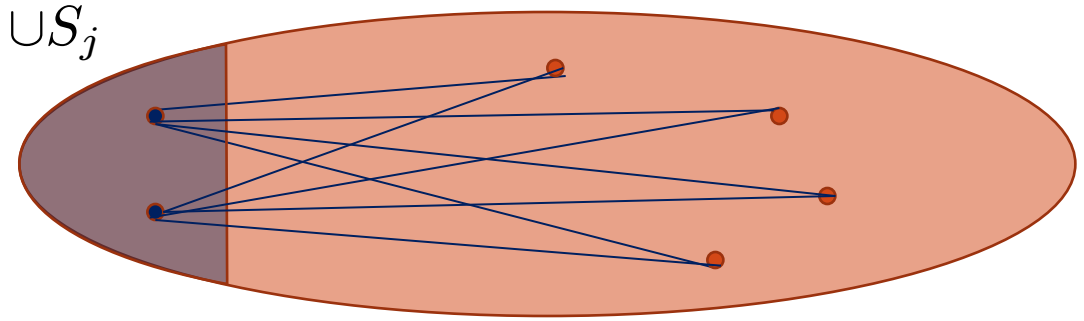
Heat-Kernel and Certificates

- If no balanced cut of conductance is found, our potential analysis yields:

$$\Psi(H^{\tau}_{G^{(T)}}, V) \cdot \frac{1}{\mathrm{poly}(n)} \longrightarrow L + \gamma \sum_{j=1}^{T-1} \sum_{i \in S_j} L(\mathrm{Star}_i) \succeq \gamma L(K_V)$$

Modified graph has $\lambda_2 \geq \gamma$

CLAIM: This is a certificate that no balanced cut of conductance < γ existed in G.



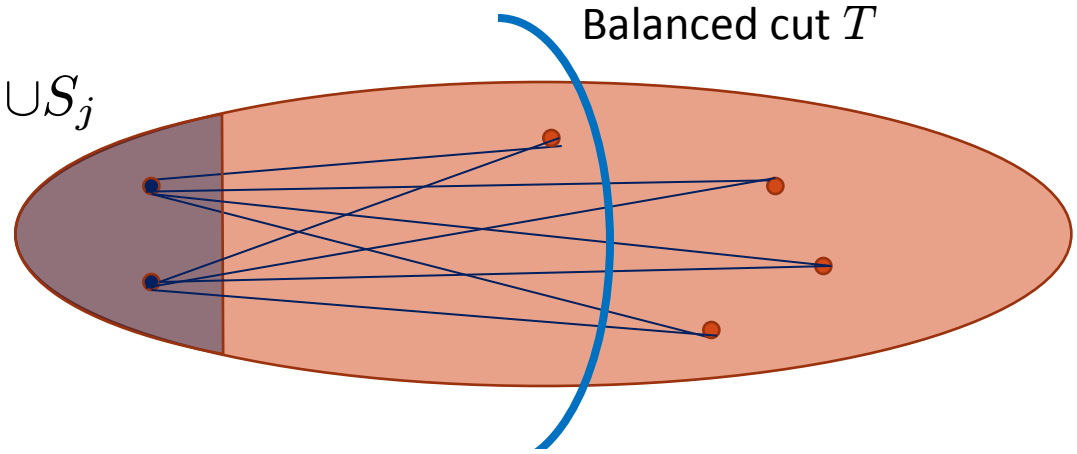
Heat-Kernel and Certificates

- If no balanced cut of conductance is found, our potential analysis yields:

$$\Psi(H^{\tau}_{G^{(T)}}, V) \cdot \frac{1}{\mathrm{poly}(n)} \longrightarrow L + \gamma \sum_{j=1}^{T-1} \sum_{i \in S_j} L(\mathrm{Star}_i) \succeq \gamma L(K_V)$$

Modified graph has $\lambda_2 \geq \gamma$

CLAIM: This is a certificate that no balanced cut of conductance < γ existed in G.



$$\phi(T) \geq \gamma - \gamma \frac{|\cup S_j|}{|T|}$$

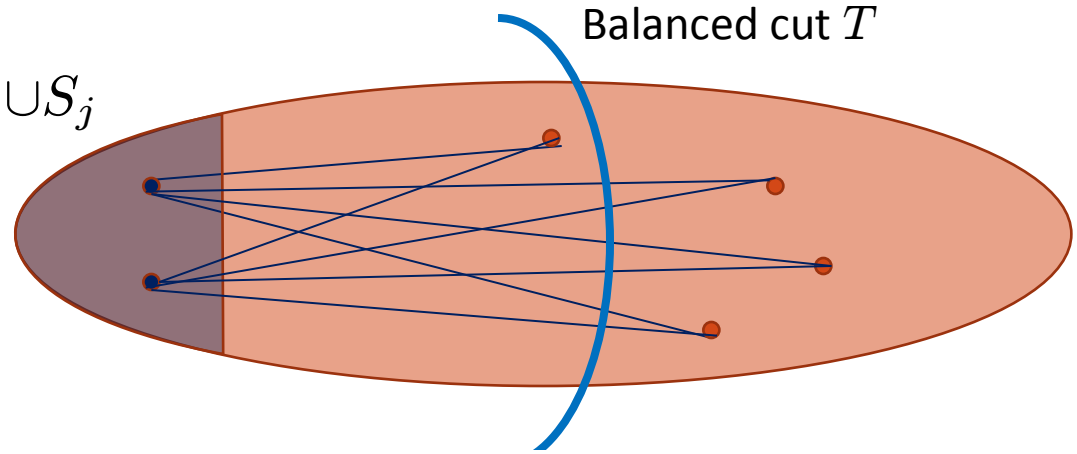
Heat-Kernel and Certificates

- If no balanced cut of conductance is found, our potential analysis yields:

$$\Psi(H^{\tau}_{G^{(T)}}, V) \cdot \frac{1}{\mathrm{poly}(n)} \longrightarrow L + \gamma \sum_{j=1}^{T-1} \sum_{i \in S_j} L(\mathrm{Star}_i) \succeq \gamma L(K_V)$$

Modified graph has $\lambda_2 \geq \gamma$

CLAIM: This is a certificate that no balanced cut of conductance < γ existed in G.



$$\phi(T) \geq \gamma - \gamma \frac{|\cup S_j|}{|T|} \geq \gamma - \gamma \frac{b/2}{b} \geq \gamma/2$$

Comparison with Recursive Eigenvector

RECURSIVE EIGENVECTOR:

We can only bound number of iterations by volume of graph removed.

$\Omega(n)$ iterations possible.

OUR ALGORITHM:

Use variance of random walk as potential.

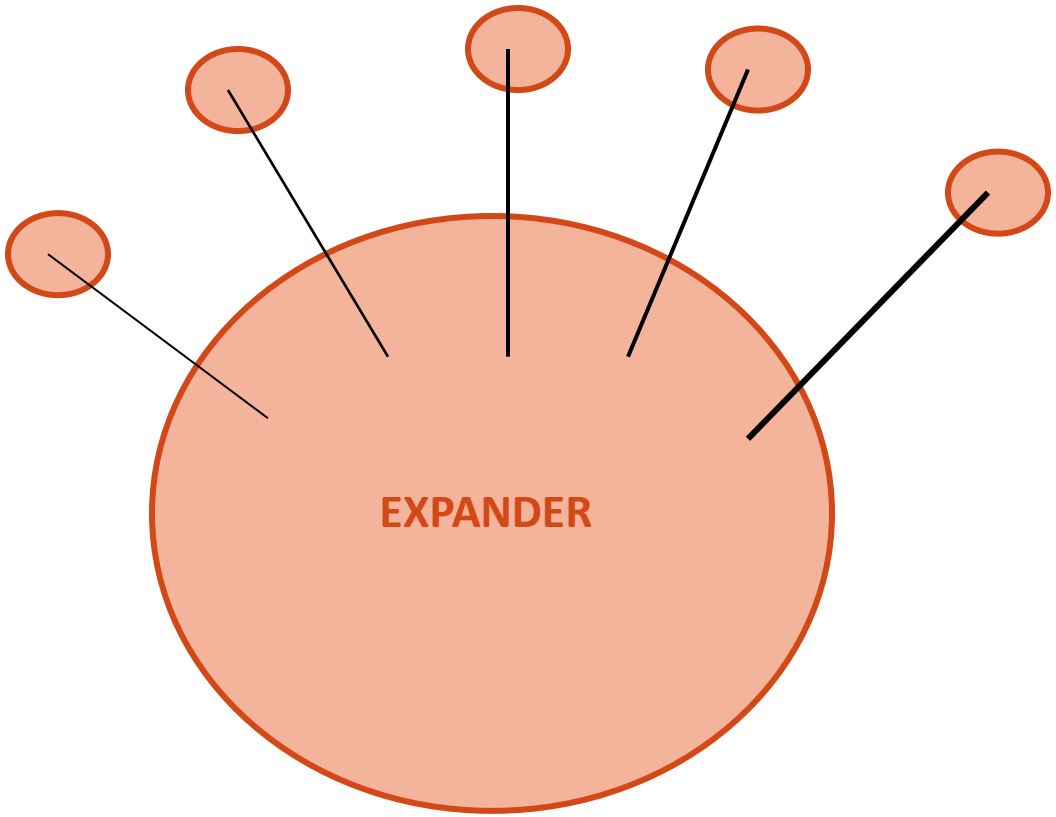
Only $O(\log n)$ iterations necessary.

$$\Psi(P, V) = \sum_{i \in V} ||Pe_i - \vec{1}/n||^2$$

RIGHT SPECTRAL NOTION OF POTENTIAL

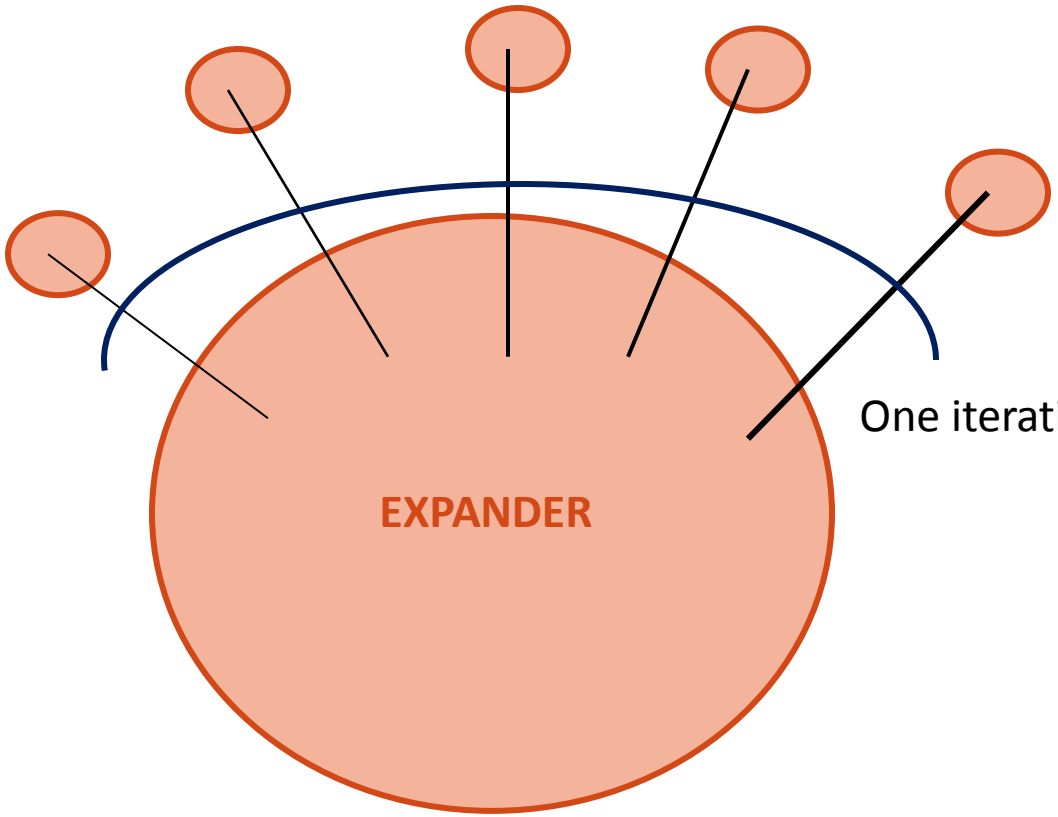
Comparison with Recursive Eigenvector

Recall worst-case example for Recursive Eigenvector:



Comparison with Recursive Eigenvector

Recall worst-case example for Recursive Eigenvector:



One iteration of our algorithm

Running Time

- Our Algorithm runs in $O(\log n)$ iterations.

- In one iteration, we perform some simple computation (projection, sweep cut) on the vector embedding $H^{\tau}_{G^{(t)}}$.
  This takes time $\tilde{O}(md)$, where d is the dimension of the embedding.

- Can use Johnson-Lindenstrauss to obtain $d = O(\log n)$.

- Hence, we only need to compute $O(\log^2 n)$ matrix-vector products

$$H^{\tau}_{G^{(t)}} u$$

- We show how to perform one such product in time $\tilde{O}(m)$ for all τ.

- OBSTACLE:

  τ, the mean number of steps in the Heat-Kernel random walk, is $\Omega(n^2)$ for path.

PART 2
Matrix Exponential Computation

Computation of Matrix Exponential

GOAL: For symmetric diagonally-dominant A, with sparsity m, compute vector v such that

$$||e^{-A}u - v|| \cdot \frac{1}{\text{poly(n)}}$$

in time $\tilde{O}(m)$.

Computation of Matrix Exponential

GOAL: For symmetric diagonally-dominant A, with sparsity m, compute vector v such that

$$||e^{-A}u - v|| \cdot \frac{1}{\text{poly(n)}}$$

in time $\tilde{O}(m)$.

KNOWN:

Lanczos method: $\tilde{O}(\sqrt{||A||})$ matrix-vector multiplications by A

Computation of Matrix Exponential

GOAL: For symmetric diagonally-dominant A, with sparsity m, compute vector v such that

$$||e^{-A}u - v|| \cdot \frac{1}{\text{poly(n)}}$$

in time $\tilde{O}(m)$.

KNOWN:

Lanczos method: $\tilde{O}(\sqrt{||A||})$ matrix-vector multiplications by A

$$A = \frac{\log n}{\gamma} L$$

$$||A|| = \tilde{O}(\frac{1}{\gamma})$$

$\longrightarrow$ Running Time: $\tilde{O}\left(\frac{m}{\sqrt{\gamma}}\right)$

Review of Lanczos Method

IDEA: Perform k matrix-vector multiplications

$$\{u, Au, A^2u, A^3u, A^4u, \ldots, A^ku\}$$

Review of Lanczos Method

IDEA: Perform k matrix-vector multiplications

$$R_k = \text{Span}\{u, Au, A^2u, A^3u, A^4u, \ldots, A^ku\}$$

k-Krylov Subspace

Compute orthonormal basis Q_k for R_k, consider A restricted to R_k

$$T_k = Q_k A Q_k^T$$

Review of Lanczos Method

IDEA: Perform k matrix-vector multiplications

$$R_k = \mathrm{Span}\{u, Au, A^2u, A^3u, A^4u, \ldots, A^ku\}$$

k-Krylov Subspace

Compute orthonormal basis Q_k for R_k, consider A restricted to R_k

$$T_k = Q_k A Q_k^T$$

As k grows, T_k is better approximation to A.

For many functions f, even for small k

$$Q_k f(T_k) Q_k^T \approx f(A)$$

Review of Lanczos Method

IDEA: Perform k matrix-vector multiplications

$$R_k = \text{Span}\{u, Au, A^2u, A^3u, A^4u, \ldots, A^ku\}$$

k-Krylov Subspace

Compute orthonormal basis Q_k for R_k, consider A restricted to R_k

$$T_k = Q_k A Q_k^T$$

As k grows, T_k is better approximation to A.

For many functions f, even for small k

$$Q_k f(T_k) Q_k^T \approx f(A)$$

True if f is close to a polynomial of degree < k

Lanczos Method and Approximation

CLAIM: There exists a polynomial p such that

$$\sup_{x \in (0, ||A||)} |p(x) - e^{-x}| \cdot \frac{1}{\text{poly}(n)}$$

and p has degree $\tilde{O}(\sqrt{||A||})$

Lanczos Method and Approximation

CLAIM: There exists a polynomial p such that

$$\sup_{x \in (0, ||A||)} |p(x) - e^{-x}| \cdot \frac{1}{\text{poly}(n)}$$

and p has degree $\tilde{O}(\sqrt{||A||})$

$$\tilde{O}(\sqrt{||A||}) = \tilde{O}\left(\frac{1}{\sqrt{\gamma}}\right)$$

matrix-vector multiplications by A

Lanczos Method and Approximation

CLAIM: There exists a polynomial p such that

$$\sup_{x \in (0, ||A||)} |p(x) - e^{-x}| \cdot \frac{1}{\text{poly}(n)}$$

and p has degree $\tilde{O}(\sqrt{||A||})$

$$\tilde{O}(\sqrt{||A||}) = \tilde{O}\left(\frac{1}{\sqrt{\gamma}}\right)$$

matrix-vector multiplications by A

OPTIMAL for this approach

Lanczos Method II: Inverse Iteration

IDEA: Apply Lanczos method to

$$B = (I + \frac{A}{k})^{-1}$$

Lanczos Method II: Inverse Iteration

IDEA: Apply Lanczos method to

$$B = (I + \frac{A}{k})^{-1}$$

k-Krylov subspace:

$$R_k = \mathrm{Span}\{u, Bu, B^2u, B^3u, B^4u, \ldots, B^ku\}$$

Lanczos Method II: Inverse Iteration

IDEA: Apply Lanczos method to

$$B = (I + \frac{A}{k})^{-1}$$

k-Krylov subspace:

$$R_k = \mathrm{Span}\{u, Bu, B^2u, B^3u, B^4u, \ldots, B^ku\}$$

APPROX: k-degree rational function approximation to e^{-x}

Lanczos Method II: Inverse Iteration

IDEA: Apply Lanczos method to

$$B = (I + \frac{A}{k})^{-1}$$

k-Krylov subspace:

$$R_k = \mathrm{Span}\{u, Bu, B^2u, B^3u, B^4u, \ldots, B^ku\}$$

APPROX: k-degree rational function approximation to e^{-x}

CLAIM: There exists a polynomial p such that

$$\sup_{x\in(0,||A||)} \left| \frac{p(x)}{(1+x/k)^k} - e^{-x} \right| \cdot \frac{1}{\mathrm{poly}(n)}$$

and p has degree $k = O(\log n)$

Lanczos Method II: Inverse Iteration

IDEA: Apply Lanczos method to

$$B = (I + \frac{A}{k})^{-1}$$

k-Krylov subspace:

$$R_k = \mathrm{Span}\{u, Bu, B^2u, B^3u, B^4u, \ldots, B^ku\}$$

APPROX: k-degree rational function approximation to e^{-x}

CLAIM: There exists a polynomial p such that

$$\sup_{x\in(0,||A||)} \left| \frac{p(x)}{(1+x/k)^k} - e^{-x} \right| \cdot \frac{1}{\mathrm{poly}(n)}$$

and p has degree $k = O(\log n)$

Lanczos Method II: Inverse Iteration

IDEA: Apply Lanczos method to

$$B = (I + \frac{A}{k})^{-1}$$

k-Krylov subspace:

$$R_k = \text{Span}\{u, Bu, B^2u, B^3u, B^4u, \ldots, B^ku\}$$

CLAIM: $\boxed{k = O(\log n)}$ matrix-vector multiplication by B

RUNNING TIME: Multiplication by B is SDD linear system

$\tilde{O}(m)$ Time per multiplication → $\tilde{O}(m)$ Total running time

Lanczos Method II: Conclusion

$$||e^{-A}u - v|| \cdot \quad \delta$$

GENERAL RESULT:

- For general SDD matrix A, running time is

$$\tilde{O}((m+n) \cdot \log(2 + ||A||) \cdot \text{polylog}(1/\delta))$$

CONTRIBUTION: Analysis has a number of obstacles

- Linear solver is approximate: error propagation in Lanczos
- Loss of symmetry

Conclusion

NOVEL ALGORITHMIC CONTRIBUTIONS

- Balanced-Cut Algorithm using Random Walks in time $\tilde{O}(m)$
- Computing the Heat-Kernel vectors in time $\tilde{O}(m)$

MAIN IDEA

Random walks provide a very useful
stable analogue of the graph eigenvector
via regularization

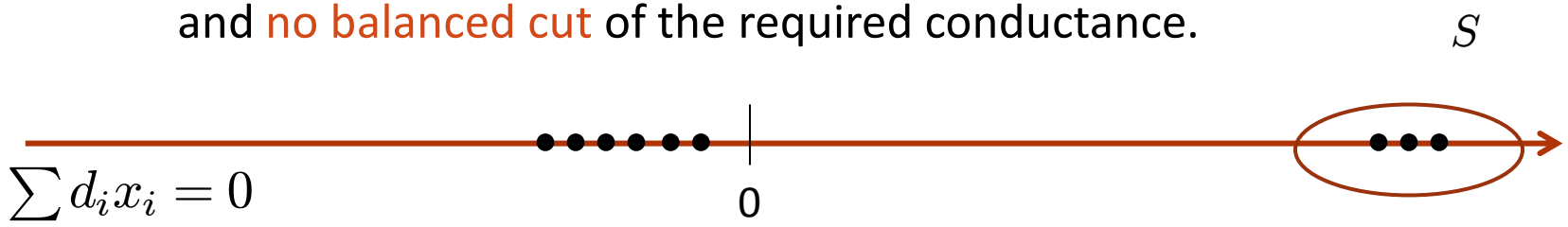
OPEN QUESTION

More applications of this idea?
Applications beyond design of fast algorithms?

A Different Interpretation

THEOREM:

Suppose eigenvector x yields an unbalanced cut S of low conductance and no balanced cut of the required conductance.



Then,

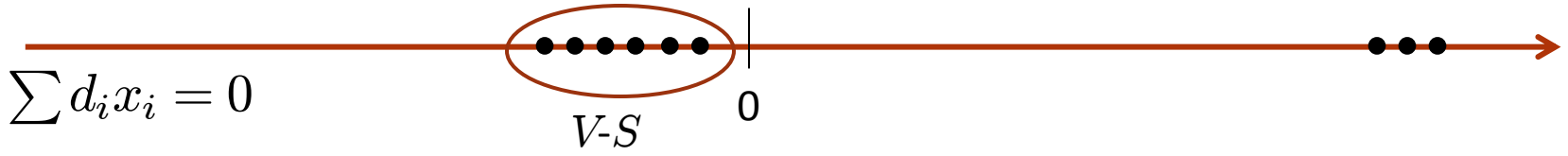
$$\sum_{i \in S} d_i x_i^2 \geq \frac{1}{2} \sum_{i \in V} d_i x_i^2.$$

In words, S contains most of the variance of the eigenvector.

A Different Interpretation

THEOREM:

Suppose eigenvector x yields an unbalanced cut S of low conductance
and no balanced cut of the required conductance.



Then,

$$\sum_{i \in S} d_i x_i^2 \geq \frac{1}{2} \sum_{i \in V} d_i x_i^2.$$

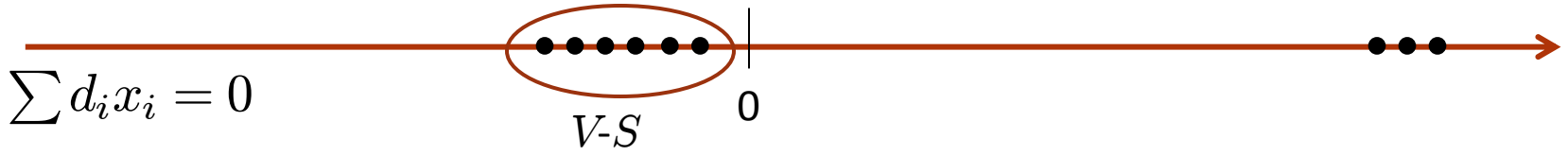
In words, S contains most of the variance of the eigenvector.

QUESTION: Does this mean the graph induced by G on V–S is much closer to have conductance at least γ?

A Different Interpretation

THEOREM:

Suppose eigenvector x yields an unbalanced cut S of low conductance
and no balanced cut of the required conductance.



Then,

$$\sum_{i \in S} d_i x_i^2 \geq \frac{1}{2} \sum_{i \in V} d_i x_i^2.$$

QUESTION: Does this mean the graph induced by G on V–S is much closer to have conductance at least γ?

ANSWER: **NO**. x may contain little or no information about G on V–S.

Next eigenvector may be only infinitesimally larger.

CONCLUSION: To make significant progress, we need an analogue of the eigenvector that captures sparse

Theorems for Our Algorithm

THEOREM 1: (WALKS HAVE NOT MIXED)

$$\Psi(P^{(t)}, V) > \frac{1}{\text{poly}(n)}$$

Can find cut of conductance $O(\sqrt{\gamma})$

Theorems for Our Algorithm

THEOREM 1: (WALKS HAVE NOT MIXED)

$$\Psi(P^{(t)}, V) > \frac{1}{\text{poly}(n)} \longrightarrow$$

Can find cut of conductance $O(\sqrt{\gamma})$

Proof: Recall that

$P^{(t)} = e^{-\tau Q^{(t)}}$ $\quad \tau = \log n/\gamma$ $\quad \Psi(P, V) = \sum_{i \in V} ||Pe_i - \vec{1}/n||^2$

Use the definition of τ. The spectrum of $P^{(t)}$ implies that

$$\sum_{ij \in E} \underbrace{||P^{(t)}e_i - P^{(t)}e_j||^2}_{\text{EDGE LENGTH}} \cdot \quad O(\gamma) \cdot \underbrace{\Psi(P^{(t)}, V)}_{\text{TOTAL VARIANCE}}$$

Theorems for Our Algorithm

THEOREM 1: (WALKS HAVE NOT MIXED)

$$\Psi(P^{(t)}, V) > \frac{1}{\text{poly}(n)} \longrightarrow$$

Can find cut of conductance $O(\sqrt{\gamma})$

Proof: Recall that

$P^{(t)} = e^{-\tau Q^{(t)}}$ $\qquad \tau = \log n/\gamma$ $\qquad \Psi(P, V) = \sum_{i \in V} ||Pe_i - \vec{1}/n||^2$

Use the definition of τ. The spectrum of $P^{(t)}$ implies that

$$\sum_{ij \in E} \underbrace{||P^{(t)}e_i - P^{(t)}e_j||^2}_{\text{EDGE LENGTH}} \cdot \quad O(\gamma) \cdot \underbrace{\Psi(P^{(t)}, V)}_{\text{TOTAL VARIANCE}}$$

Hence, by a random projection of the embedding $\{P\, e_i\}$, followed by a sweep cut, we can recover the required cut.

SDP ROUNDING TECHNIQUE

Theorems for Our Algorithm

THEOREM 2: (WALKS HAVE MIXED)

$$\Psi(P^{(t)}, V) \cdot \frac{1}{\mathrm{poly}(n)} \longrightarrow$$

No $\Omega(b)$-balanced cut of conductance $O(\gamma)$

Theorems for Our Algorithm

THEOREM 2: (WALKS HAVE MIXED)

$$\Psi(P^{(t)}, V) \cdot \frac{1}{\text{poly}(n)} \longrightarrow$$

No $\Omega(b)$-balanced cut of conductance $O(\gamma)$

Proof: Consider $S = \cup S_i$. Notice that S is unbalanced.

Assumption is equivalent to

$$L(K_V) \bullet e^{-\tau L - O(\log n) \sum_{i \in S} L(S_i)} \cdot \frac{1}{\text{poly}(n)}.$$

Theorems for Our Algorithm

THEOREM 2: (WALKS HAVE MIXED)

$$\Psi(P^{(t)}, V) \cdot \frac{1}{\mathrm{poly}(n)} \longrightarrow$$

No $\Omega(b)$-balanced cut of conductance $O(\gamma)$

Proof: Consider $S = \cup S_i$. Notice that S is unbalanced.

Assumption is equivalent to

$$L(K_V) \bullet e^{-\tau L - O(\log n) \sum_{i \in S} L(S_i)} \cdot \frac{1}{\mathrm{poly}(n)}.$$

By taking logs,

$$L + O(\gamma) \sum_{i \in S} L(S_i) \succeq -(\gamma) L(K_V).$$

SDP DUAL CERTIFICATE

Theorems for Our Algorithm

THEOREM 2: (WALKS HAVE MIXED)

$$\Psi(P^{(t)}, V) \cdot \frac{1}{\text{poly}(n)} \longrightarrow$$

No $\Omega(b)$-balanced cut of conductance $O(\gamma)$

Proof: Consider $S = \cup S_i$. Notice that S is unbalanced.

Assumption is equivalent to

$$L(K_V) \bullet e^{-\tau L - O(\log n) \sum_{i \in S} L(S_i)} \cdot \frac{1}{\text{poly}(n)}.$$

By taking logs,

$$L + O(\gamma) \sum_{i \in S} L(S_i) \succeq -(\gamma) L(K_V).$$ SDP DUAL CERTIFICATE

This is a certificate that no $\Omega(1)$-balanced cut of conductance $O(\gamma)$ exists, as evaluating the quadratic form for a vector representing a balanced cut U yields

$$\phi(U) \geq -(\gamma) - \frac{\text{vol}(S)}{\text{vol}(U)} O(\gamma) \geq -(\gamma)$$

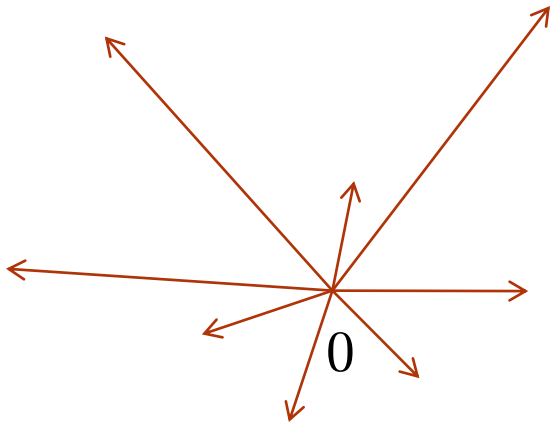
as long as S is sufficiently unbalanced.

SDP Interpretation

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

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

$$\forall i \in V \qquad \mathbb{E}_{j\in V} \quad ||v_i - v_j||^2 \cdot \frac{1}{b} \cdot \frac{1}{2m}. \qquad \text{LENGTH OF STAR EDGES}$$

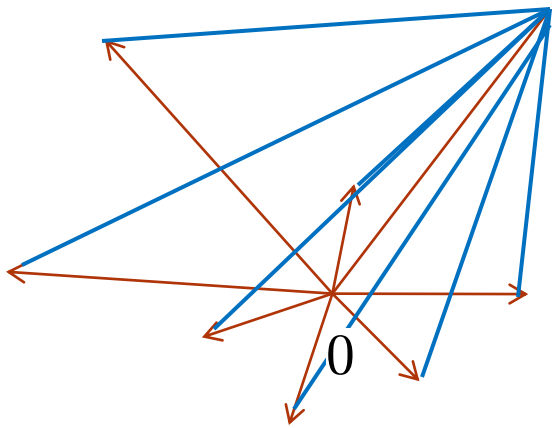


SDP Interpretation

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

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

$$\forall i \in V \qquad \mathbb{E}_{j\in V} \quad ||v_i - v_j||^2 \cdot \frac{1}{b} \cdot \frac{1}{2m}. \qquad \text{LENGTH OF STAR EDGES}$$



SHORT RADIUS