Fractionally Log-Concave & Sector-Stable Polynomials: Counting Planar Matchings and More

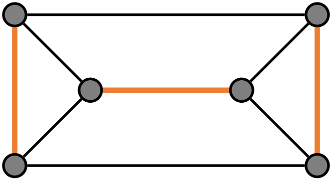
Yeganeh Alimohammadi   Nima Anari
Kirankumar Shiragur   **Thuy-Duong "June" Vuong**

Stanford

Northwestern Theory Seminar
November 2, 2022

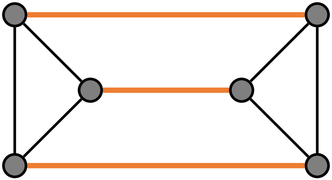


Counting Matching



Orange edges form a matching

Counting Matching



Orange edges form a matching
Total # matchings: 2

Q: Efficient algorithm for counting fixed-size matching in graph?
A:

Q: Efficient algorithm for counting fixed-size matching in graph?
A: Intractable
Q: Why do we care?
A: Counting matching might shed light on P=NP question.

Q: Efficient algorithm for approximate counting fixed-size matching in graph?
A:

Q: Efficient algorithm for approximate counting fixed-size matching in graph?
A: Open for decades

Q: Efficient algorithm for approximate counting fixed-size matching in graph?
A: Open for decades
Q: Efficient algorithm for approximate counting fixed-size matching in planar graph?
A: This work

Overview

1. Background
   - Counting Problems
   - Matchings
2. Technique
   - Reduce Counting to Sampling
   - Sampling via Random Walks
   - Fast Mixing From Sector-Stability
3. Other Applications

Decision vs. Counting

Given description for $L \subseteq \{0,1\}^n$

Decision
Decide if L is empty

- SAT: Decide if ϕ has a satisfying assignment

Counting

Approximate Counting

Decision vs. Counting

Given description for $L \subseteq \{0,1\}^n$ e.g. L is set of x satisfying $(x_1 \vee x_2 \vee \bar{x_3}) \wedge (x_1 \vee x_4 \vee x_5)$.

Decision	**Counting**	**Approximate Counting**
Decide if L is empty	Compute $\lvert L \rvert$	
■ SAT: Decide if ϕ has a satisfying assignment	■ #SAT: Count # satisfying assignments	

Decision vs. Counting

Given description for $L \subseteq \{0,1\}^n$ e.g. L is set of x satisfying $(x_1 \vee x_2 \vee \bar{x_3}) \wedge (x_1 \vee x_4 \vee x_5)$.

Decision

Decide if L is empty

- SAT: Decide if ϕ has a satisfying assignment

Counting

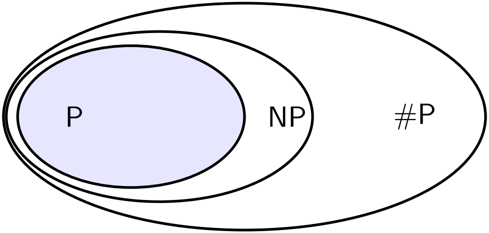
Compute $|L|$

- #SAT: Count # satisfying assignments

Approximate Counting

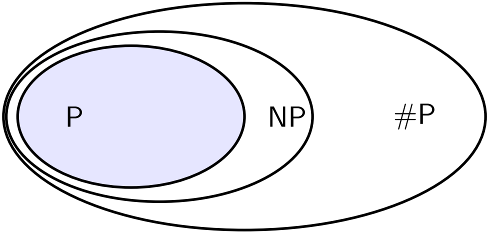
Compute $\hat{Z}$ s.t. $0.9\hat{Z} \leq |L| \leq \hat{Z}$

Decision vs. Counting



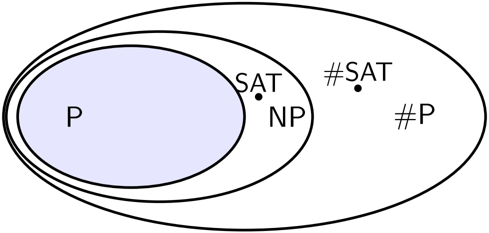
- Counting is harder than Decision

Decision vs. Counting



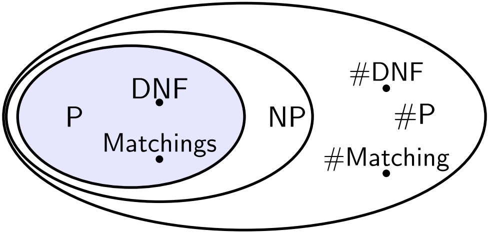
- Counting is harder than Decision
- #P-complete problems: at least as hard as all problems in #P.

Decision vs. Counting



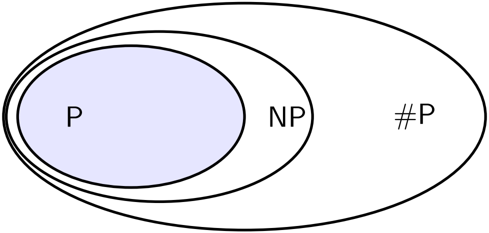
- Counting is harder than Decision
- #P-complete problems: at least as hard as all problems in #P. E.g.: #SAT

Decision vs. Counting



- Counting is harder than Decision
- #P-complete problems: at least as hard as all problems in #P.
- Easy to decide–hard to count: DNF formulas, matchings

Decision vs. Counting



- Counting is harder than Decision
- #P-complete problems: at least as hard as all problems in #P.
- Easy to decide–hard to count: DNF formulas, matchings
- Easier to approximate count?

Concrete example: matchings in graph

Complexity of Counting Matching

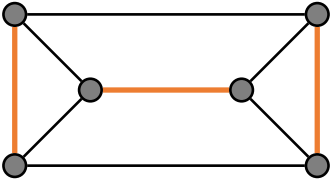
	General	Bipartite	Planar
Perfect matching			
k-matching			

: approximate, : exact

: in P, : #P-complete, ?: Open

Perfect Matching

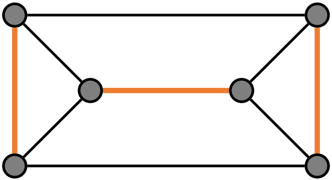
A set of $n/2$ edges that meets every vertex at most once.



Decision $\equiv$ decide if G has a perfect matching (PM): efficient [Edmonds'65]

Perfect Matching

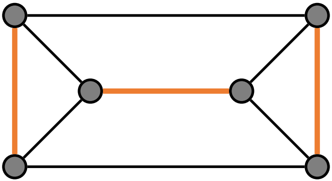
A set of $n/2$ edges that meets every vertex at most once.



Decision $\equiv$ decide if G has a perfect matching (PM): efficient [Edmonds'65]
Counting $\equiv$ compute #PM: intractable (#P-complete [Valiant'87])

Perfect Matching

A set of $n/2$ edges that meets every vertex at most once.



Decision $\equiv$ decide if G has a perfect matching (PM): efficient [Edmonds'65]
Counting $\equiv$ compute #PM: intractable (#P-complete [Valiant'87])
Approximate counting $\equiv$ output $\hat{Z}$ s.t. $0.9\hat{Z} \leq \#PM \leq \hat{Z}$: open

	General	Bipartite	Planar
Perfect matching	☹ [Val87] ?		
k-matching			

: approximate, : exact

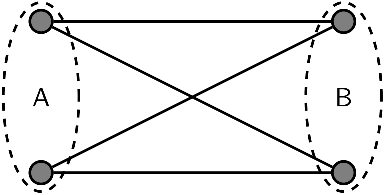
☺: in P, ☹: #P-complete, ?: Open

Partial results for specific graph families.

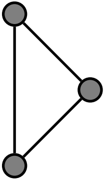
Next: bipartite and planar graphs

Bipartite Graphs

Bipartite graph $G = G(A \cup B, E)$: $A \cap B = \emptyset, E \subseteq A \times B$



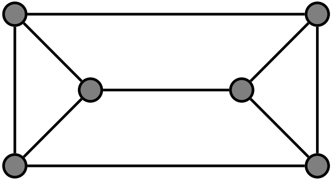
bipartite graph



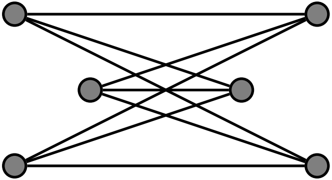
non-bipartite graph

Planar Graphs

Planar graphs: can be drawn on plane without crossing edges.



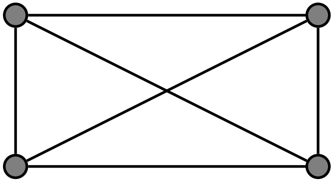
planar graph



non-planar graph

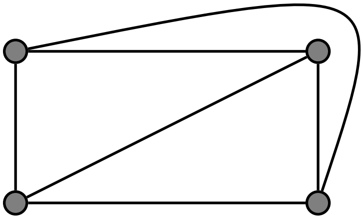
Planar Graphs

Planar graphs: can be drawn on plane without crossing edges.



Planar Graphs

Planar graphs: can be drawn on plane without crossing edges.



	General	Bipartite	Planar
Perfect matching	☹ [Val87] ?		
k-matching			

: approximate, : exact
☺: in P, ☹: #P-complete, ?: Open

Perfect Matching–Bipartite Graphs

- Counting: #P-complete [Valiant'87]

Perfect Matching–Bipartite Graphs

- Counting: #P-complete [Valiant'87]

Perfect Matching–Bipartite Graphs

- Counting: #P-complete [Valiant'87]
- Approximate counting: in P [Jerrum-Sinclair-Vigoda'04]

	General	Bipartite	Planar
Perfect matching (exact)	☹ [Val87]	☹ [Val87]	
Perfect matching (approximate)	?	☺ [JSV04]	
k-matching (exact)			
k-matching (approximate)			

□: approximate, □: exact
☺: in P, ☹: #P-complete, ?: Open

Perfect Matching–Planar Graphs

- Counting perfect matching in planar graph is in P!! [Kasteleyn'67]

	General	Bipartite	Planar
Perfect matching	☹ [Val87]	☹ [Val87]	☺ [Kas67]
	?	☺ [JSV04]	☺ [Kas67]
k-matching			

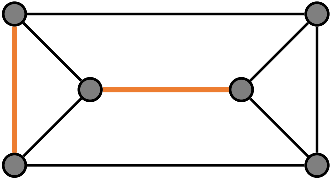
: approximate, : exact

☺: in P, ☹: #P-complete, ?: Open

What about non-perfect matching?

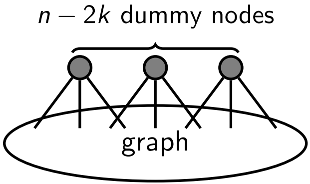
Non-Perfect Matchings: k-Matchings

A set of $k/2$ edges that meets every vertex at most once.
Equivalently, a perfect matching on $S \subseteq V$ with $|S| = k$.



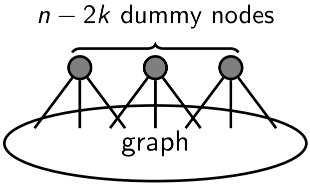
k-Matchings vs. Perfect Matchings

- Decision/Counting: $\exists$ reduction from k-matching in G to perfect matching in G'



k-Matchings vs. Perfect Matchings

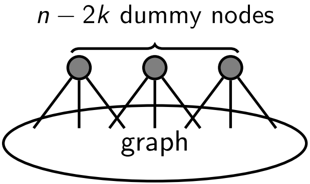
- Decision/Counting: $\exists$ reduction from k-matching in G to perfect matching in G'



- If G is bipartite, G' is bipartite

k-Matchings vs. Perfect Matchings

- Decision/Counting: $\exists$ reduction from k-matching in G to perfect matching in G'



- If G is bipartite, G' is bipartite

	General	Bipartite	Planar
Perfect matching	☹ [Val87]	☹ [Val87]	☺ [Kas67]
	?	☺ [JSV04]	☺ [Kas67]
k-matching	☹ [Val87]	☹ [Val87]	
	?	☺ [JSV04]	

: approximate, : exact

☺: in P, ☹: #P-complete, ?: Open

k-Matchings vs. Perfect Matchings

- But G' is not planar even if G is planar

k-Matchings vs. Perfect Matchings

- But G' is not planar even if G is planar
- Counting k-matching in planar graph is #P-complete [Jerrum'87]

k-Matchings vs. Perfect Matchings

- But G' is not planar even if G is planar
- Counting k-matching in planar graph is #P-complete [Jerrum'87]
- Approximate counting k-matching in planar graph is in P [this work]

	General	Bipartite	Planar
Perfect matching	☹ [Val87]	☹ [Val87]	☺ [Kas67]
	?	☺ [JSV04]	☺ [Kas67]
k-matching	☹ [Val87]	☹ [Val87]	☹ [Jer87]
	?	☺ [JSV04]	☺ [this]

□: approximate, □: exact
☺: in P, ☹: #P-complete, ?: Open

Main Result: Counting Matching

Theorem

Efficient algorithm to approximately count k-matching (runtime $\approx \mathrm{poly}(|V|, k, \log \frac{1}{\epsilon})$*)*

- Planar graphs ☺

Main Result: Counting Matching

Theorem

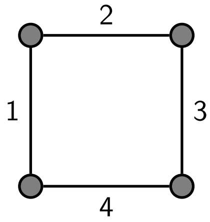
Efficient algorithm to approximately count k-matching (runtime $\approx \text{poly}(|V|, k, \log \frac{1}{\epsilon})$*)*

- Planar graphs ☺
- Any graph where counting #PM of subgraphs is easy ☺

Main Result: Counting Matching

Theorem

Efficient algorithm to approximately count k-matching (runtime $\approx \text{poly}(|V|, k, \log \frac{1}{\epsilon})$)



$$\sum_{M\text{:perfect matching}} w(M) = 1 \times 3 + 2 \times 4$$

- Planar graphs ☺
- Any graph where counting #PM of subgraphs is easy ☺
- Weighted graphs ☺

Overview

Reduce Counting to Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

Reduce Counting to Sampling

Approximate Counting $\equiv$ Compute $\hat{Z}$ s.t. $\frac{\hat{Z}}{\#\text{ k-matchings}} \in [1-\epsilon, 1]$
Approximate Sampling $\equiv$ Output a k-matching according to a distribution that is ϵ-away from the uniform dist. over k-matchings

Reduce Counting To Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

- Approximate Counting $\Leftrightarrow$ Approximate Sampling [Jerrum-Vazirani-Vazirani'86]

$$\#\text{M} = \frac{\#\text{M}}{\#\text{M contains 1}} \times \frac{\#\text{M contains 1}}{\#\text{M contains 1,2}} \cdots$$

Reduce Counting To Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

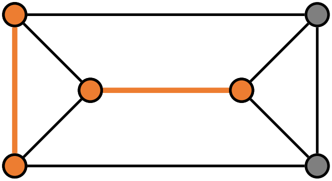
- Approximate Counting $\Leftrightarrow$ Approximate Sampling

$$\#\mathrm{M} = \frac{1}{\mathbb{P}[\mathrm{M}\text{ contains }1]} \times \frac{1}{\mathbb{P}[\mathrm{M}\text{ contains }2 \;\mid \text{contain }1]} \cdots$$

Reduce Counting To Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

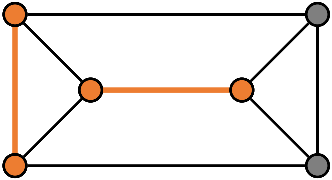
- Approximate Counting $\Leftrightarrow$ Approximate Sampling
- Sample endpoints of k-matchings



Reduce Counting To Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

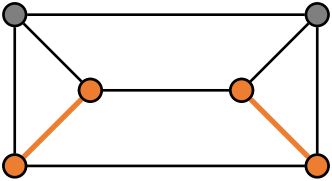
- Approximate Counting $\Leftrightarrow$ Approximate Sampling
- Sample endpoints of k-matchings
- Given endpoints set S ($|S| = k$), sample a matching on S



Reduce Counting To Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

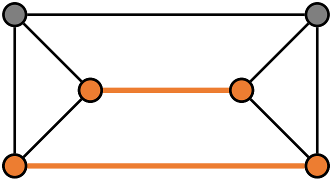
- Approximate Counting $\Leftrightarrow$ Approximate Sampling
- Sample endpoints of k-matchings
- Given endpoints set S ($|S| = k$), sample a matching on S



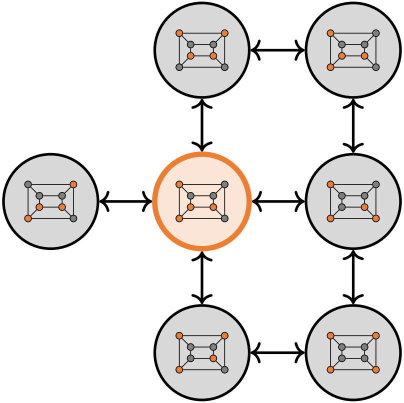
Reduce Counting To Sampling

Counting $\equiv$ Compute # k-matchings in graph
Sampling $\equiv$ Output a random k-matching

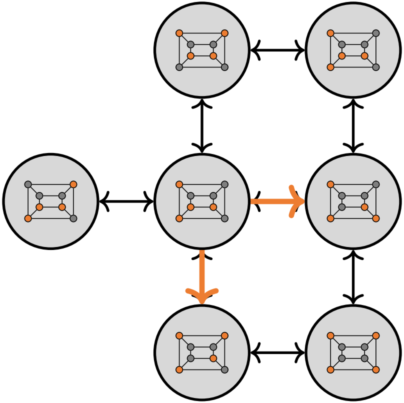
- Approximate Counting $\Leftrightarrow$ Approximate Sampling
- Sample endpoints of k-matchings
- Given endpoints set S ($|S| = k$), sample a matching on S



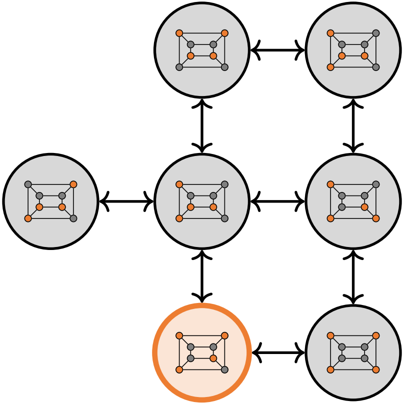
Sample Endpoints Using Random Walk



Sample Endpoints Using Random Walk



Sample Endpoints Using Random Walk



Sample Endpoints Using Random Walk

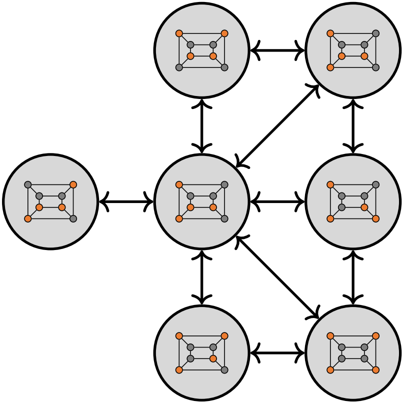
- Start at distribution μ_0, apply transition rule for T steps to reach desired distribution μ

Sample Endpoints Using Random Walk

- Start at distribution μ_0, apply transition rule for T steps to reach desired distribution μ
- Want: T is small ($=\text{poly}(|V|, k)$) i.e. fast mixing

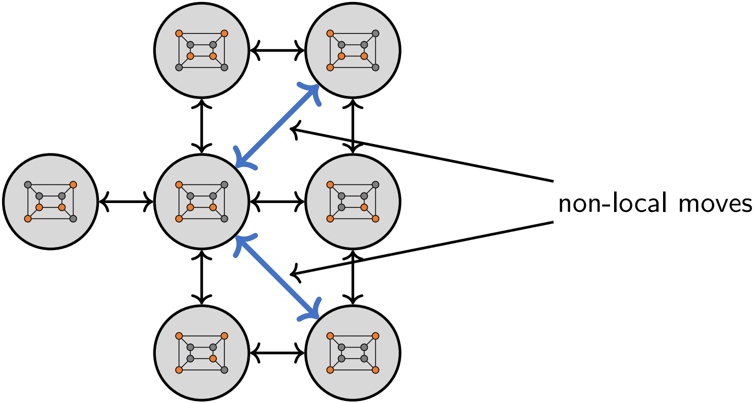
Sample Endpoints Using Random Walk

Local Walk: Only allow local moves between S, T that are close
i.e. $|S \setminus T| \leq 1 \rightarrow$ easy to transition between S, T ☺



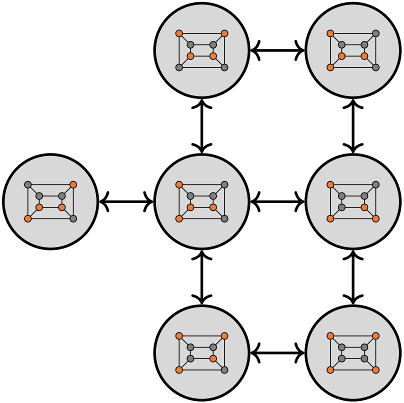
Sample Endpoints Using Random Walk

Local Walk: Only allow local moves between S, T that are close i.e. $|S \setminus T| \leq 1 \to$ easy to transition between S, T ☺

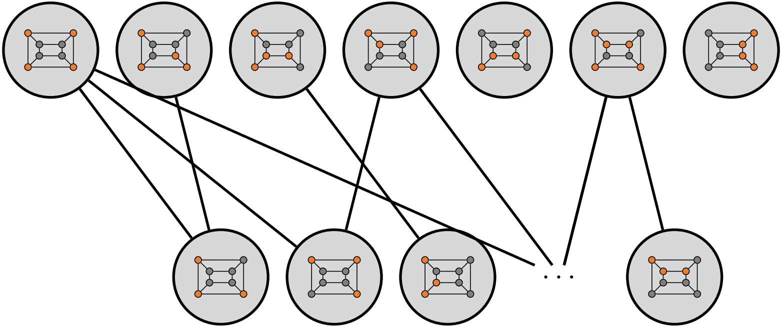


Sample Endpoints Using Random Walk

Local Walk: Only allow local moves between S, T that are close
i.e. $|S \setminus T| \leq 1 \rightarrow$ easy to transition between S, T ☺



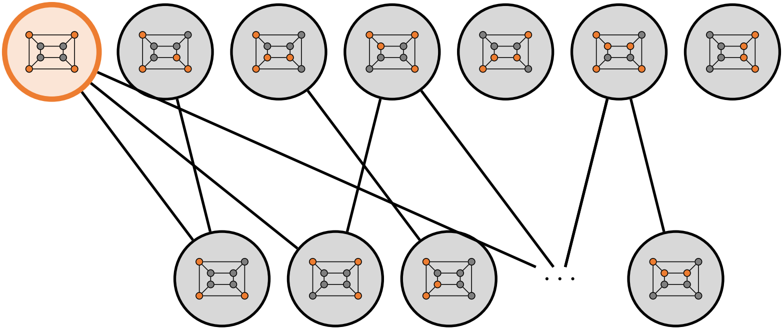
Random Walk $k \leftrightarrow (k-1)$ (1-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop an element uniformly at random.

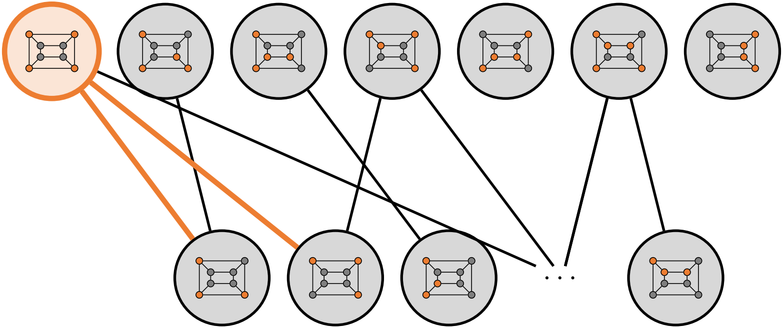
Random Walk $k \leftrightarrow (k-1)$ (1-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop an element uniformly at random.

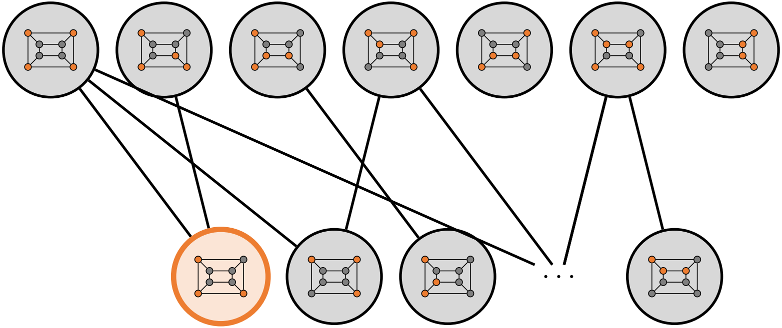
Random Walk $k \leftrightarrow (k-1)$ (1-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop an element uniformly at random.

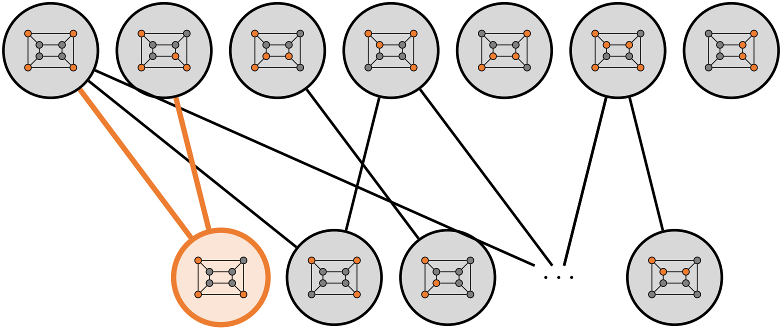
Random Walk $k \leftrightarrow (k-1)$ (1-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop an element uniformly at random.

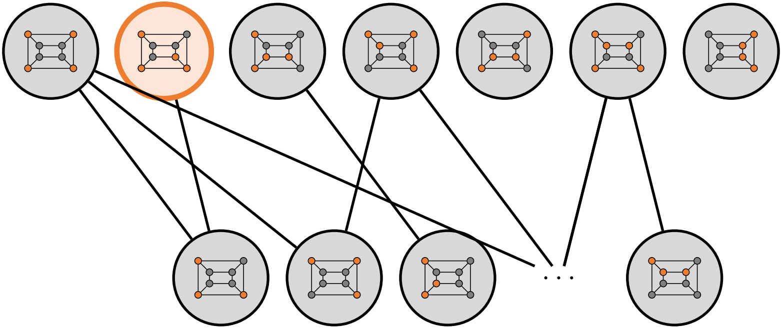
Random Walk $k \leftrightarrow (k-1)$ (1-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop an element uniformly at random.
2. Add an element with probability $\propto \mu$(resulting set).

Random Walk $k \leftrightarrow (k-1)$ (1-Step Down-Up Walk)

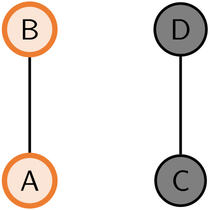


Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop an element uniformly at random.
2. Add an element with probability $\propto \mu$(resulting set).

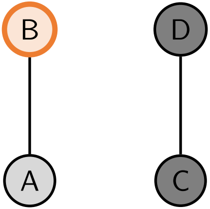
Apply To μ_k^{endpoint} ...

- $k \leftrightarrow (k-1)$-random walk won't mix. ☹



Apply To μ_k^{endpoint} ...

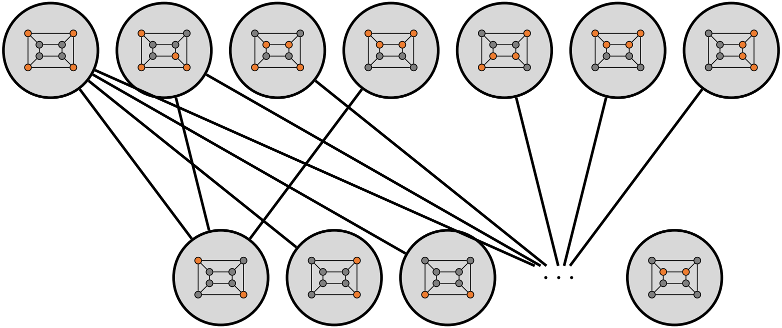
- $k \leftrightarrow (k-1)$-random walk won't mix. ☹



Apply To μ_k^{endpoint}...

- $k \leftrightarrow (k-1)$-random walk won't mix. ☹
- $k \leftrightarrow (k-2)$ walk does! ☺

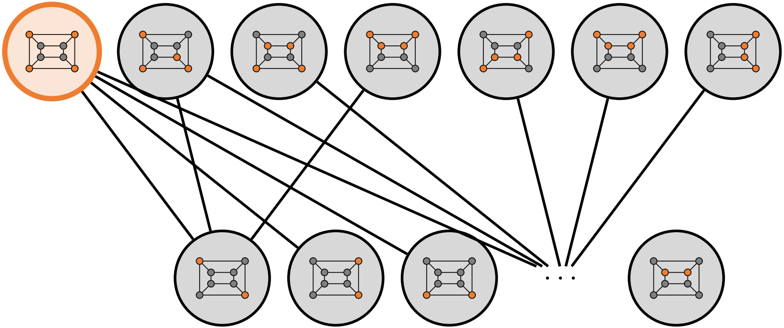
Random Walk $k \leftrightarrow (k-2)$ (Multi-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop 2 element uniformly at random.

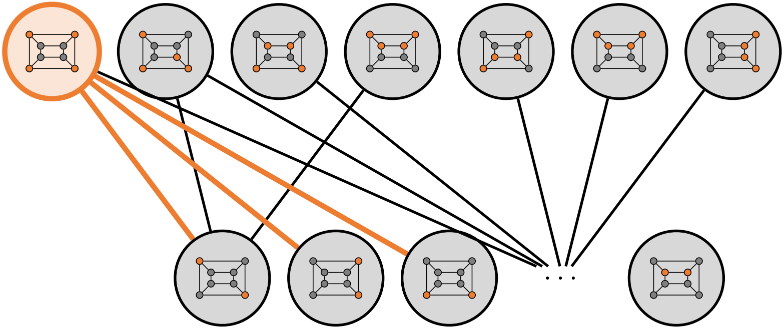
Random Walk $k \leftrightarrow (k-2)$ (Multi-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop 2 element uniformly at random.
2. Add 2 element with probability $\propto \mu$(resulting set).

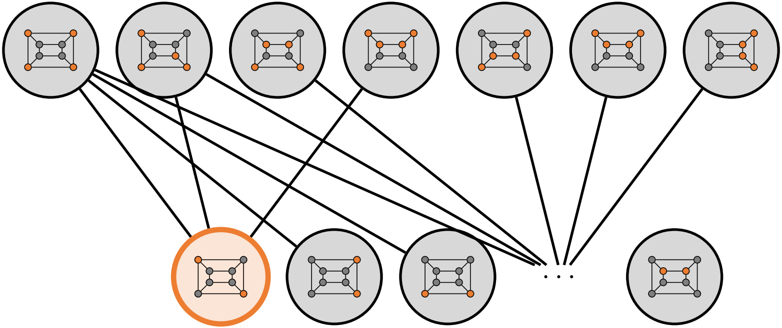
Random Walk $k \leftrightarrow (k-2)$ (Multi-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop 2 element uniformly at random.
2. Add 2 element with probability $\propto \mu$(resulting set).

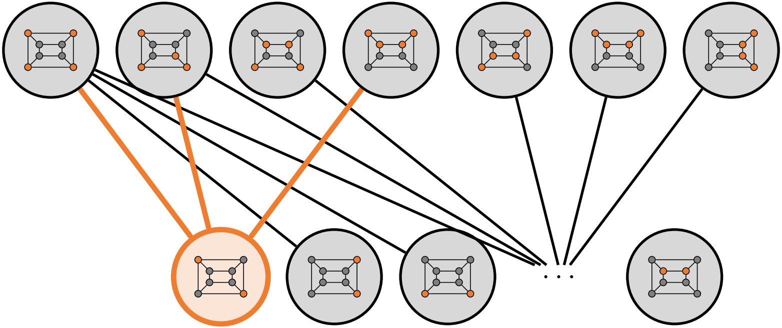
Random Walk $k \leftrightarrow (k-2)$ (Multi-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop 2 element uniformly at random.
2. Add 2 element with probability $\propto \mu$(resulting set).

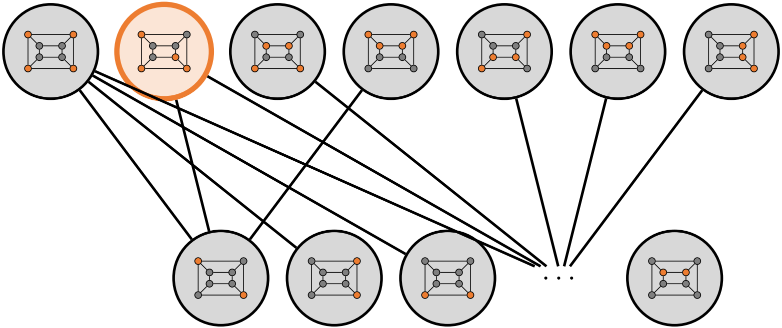
Random Walk $k \leftrightarrow (k-2)$ (Multi-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop 2 element uniformly at random.
2. Add 2 element with probability $\propto \mu$(resulting set).

Random Walk $k \leftrightarrow (k-2)$ (Multi-Step Down-Up Walk)



Sample from homogeneous distribution μ over $\binom{[n]}{k}$.

1. Drop 2 element uniformly at random.
2. Add 2 element with probability $\propto \mu$(resulting set).

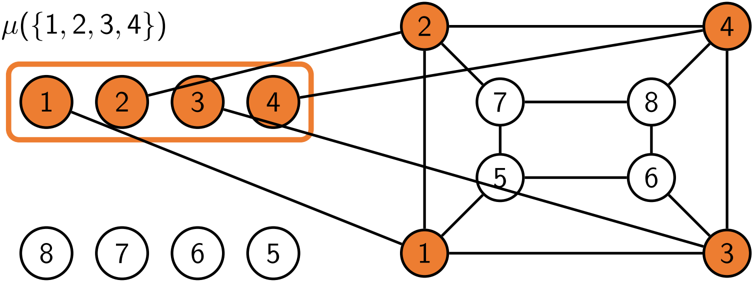
We can bound mixing time of multi-step down-up walk by proving that the hypergraph associated with μ is a high-dimensional expander.

High-dimensional expander

View distribution as weighted hypergraphs

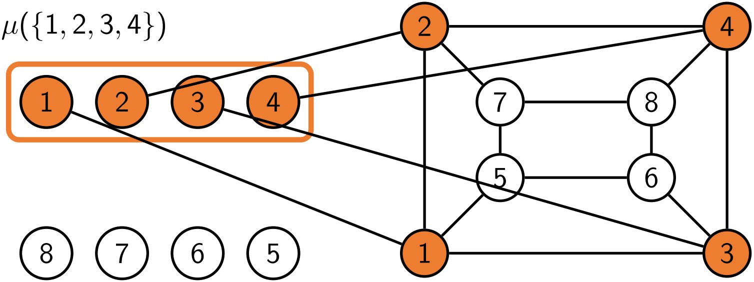
High-dimensional expander

View distribution $\mu : \binom{[n]}{k} \to \mathbb{R}_{\geq 0}$ as weighted k-uniform hypergraphs



High-dimensional expander

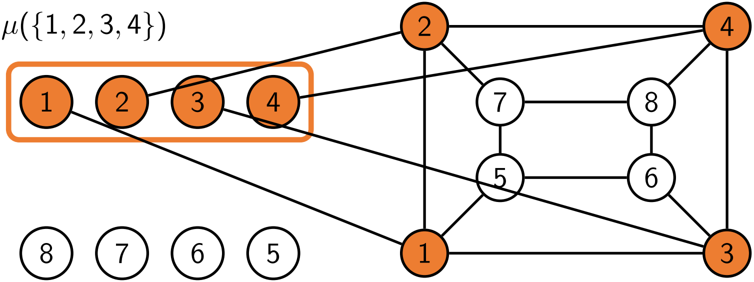
View distribution $\mu : \binom{[n]}{k} \to \mathbb{R}_{\geq 0}$ as weighted k-uniform hypergraphs



High-dimensional expander

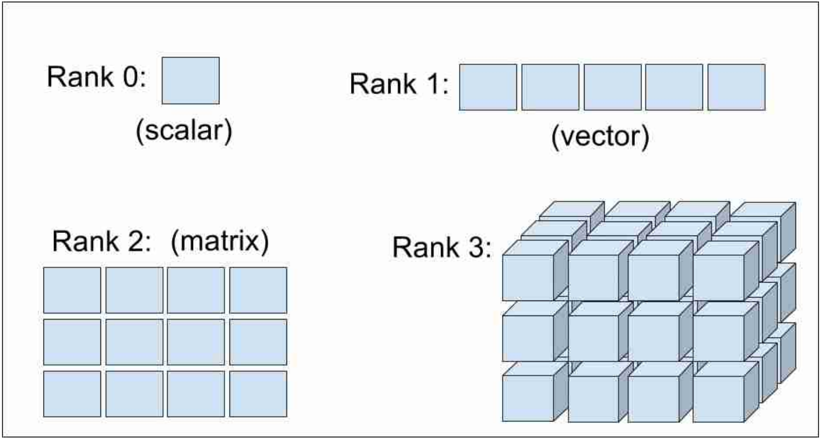
View distribution $\mu : \binom{[n]}{k} \to \mathbb{R}_{\geq 0}$ as weighted k-uniform hypergraphs

High-dimensional expansion (HDX): measuring connected-ness of hypergraph



High-dimensional expander (HDX): tensor view

Tensor T_μ : $T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.



High-dimensional expander (HDX): tensor view

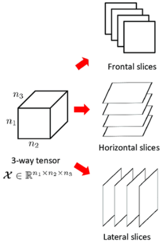
Tensor T_μ : $T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.

- $k = 2$: (graph) expansion $\equiv$ spectral properties of the adjacency matrix

High-dimensional expander (HDX): tensor view

Tensor T_μ : $T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.

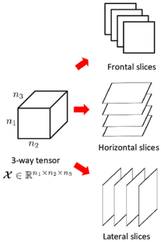
- $k = 2$: (graph) expansion $\equiv$ spectral properties of the adjacency matrix
- $k > 2$: HDX $\equiv$ spectral properties of all dimension 2 slices & averages of slices of tensor.



High-dimensional expander (HDX): tensor view

Tensor T_μ : $T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.

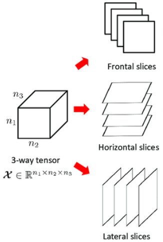
- $k = 2$: (graph) expansion $\equiv$ spectral properties of the adjacency matrix
- $k > 2$: HDX $\equiv$ spectral properties of all dimension 2 slices & averages of slices of tensor. slice (link): $\langle T_\mu, e_{i_1} \otimes e_{i_2} \otimes \cdots \otimes e_{i_{k-2}} \rangle$ average of slice (1-skeleton): $\langle T_\mu, (\mathbb{1}/n) \otimes (\mathbb{1}/n) \otimes \cdots \otimes (\mathbb{1}/n) \rangle$



High-dimensional expander (HDX): tensor view

Tensor T_μ : $T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.

- $k = 2$: (graph) expansion $\equiv$ spectral properties of the adjacency matrix
- $k > 2$: HDX $\equiv$ spectral properties of all dimension 2 slices & averages of slices of tensor.



Mixing time $\equiv$ spectral properties of transition matrix M

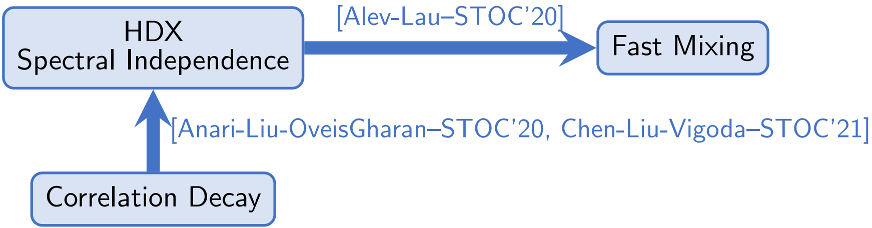
think M as "unpacking" of T_μ.

Proof of fast mixing: outline

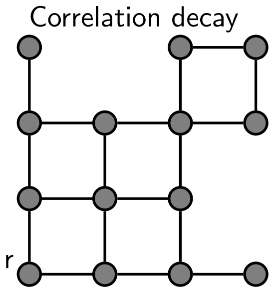
HDX
Spectral Independence

Proof of fast mixing: outline

Proof of fast mixing: outline

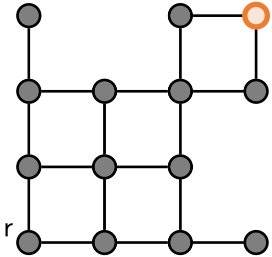


Proof of fast mixing: outline



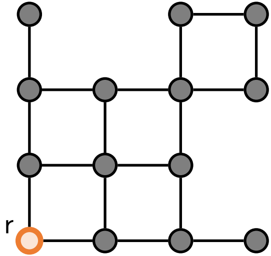
Proof of fast mixing: outline

Correlation decay

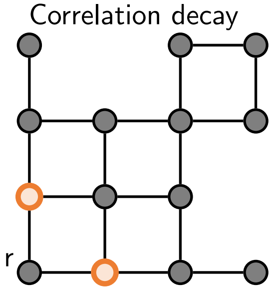


Proof of fast mixing: outline

Correlation decay

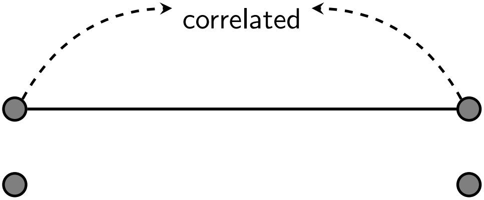


Proof of fast mixing: outline

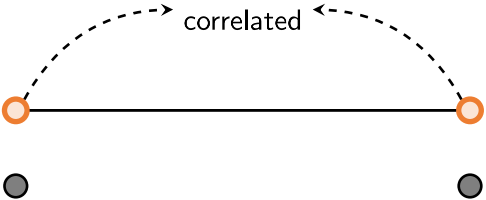


r is only highly correlated with a few "nearby" vertices

Proof of fast mixing: outline

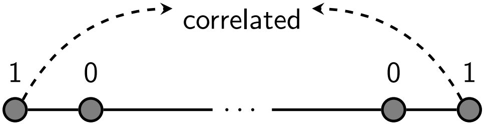


Proof of fast mixing: outline

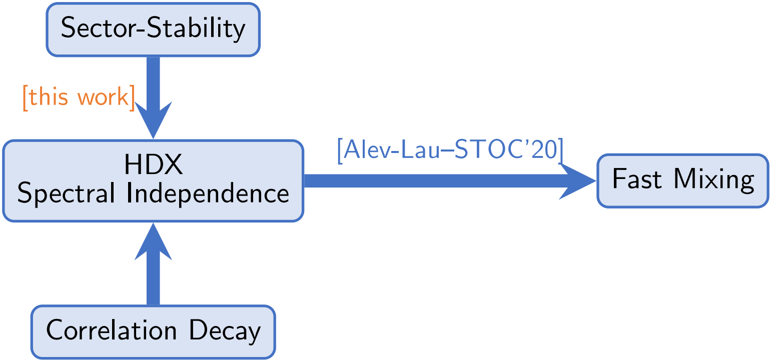


Either both are endpoints (of a matching) or neither are
$\Rightarrow$ *positively* correlated.

Proof of fast mixing: outline



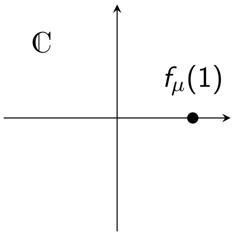
Proof of fast mixing: outline



From root-free-ness to fast algorithm: intuition

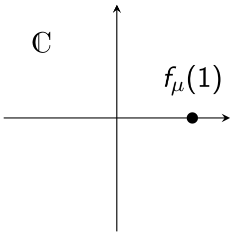
- $\mu \sim \binom{[n]}{k} \overset{\text{encode}}{\longleftrightarrow} f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) \prod_{i \in S} z_i$

From root-free-ness to fast algorithm: intuition



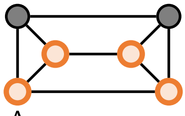
- $\mu \sim \binom{[n]}{k} \overset{\text{encode}}{\longleftrightarrow} f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) \prod_{i \in S} z_i$
- Counting $\equiv$ Compute $f_\mu(\mathbf{1})$

From root-free-ness to fast algorithm: intuition



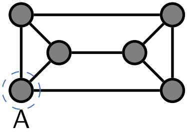
- $\mu \sim \binom{[n]}{k} \overset{\text{encode}}{\longleftrightarrow} f_\mu(z_1, \dots, z_n) = \sum_S \mu(S) \prod_{i \in S} z_i$
- Counting $\equiv$ Compute $f_\mu(\mathbf{1})$
- $f_\mu(\mathbf{z}) = 0 \longleftrightarrow \log f_\mu(\mathbf{z})$ singular

From root-free-ness to fast algorithm: intuition



- $\mu \sim \binom{[n]}{k} \overset{\text{encode}}{\longleftrightarrow} f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) \prod_{i \in S} z_i$
- Counting $\equiv$ Compute $f_\mu(\mathbf{1})$
- $f_\mu(\mathbf{z}) = 0 \longleftrightarrow \log f_\mu(\mathbf{z})$ singular $\Rightarrow$ abrupt change of derivatives of $\log f_\mu(\mathbf{z})$

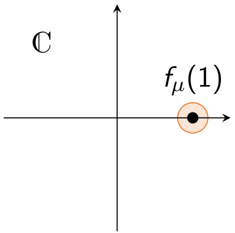
From root-free-ness to fast algorithm: intuition



- $\mu \sim \binom{[n]}{k} \overset{\text{encode}}{\longleftrightarrow} f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) \prod_{i \in S} z_i$
- Counting $\equiv$ Compute $f_\mu(\mathbf{1})$
- $f_\mu(\mathbf{z}) = 0 \longleftrightarrow \log f_\mu(\mathbf{z})$ singular $\Rightarrow$ abrupt change of derivatives of $\log f_\mu(\mathbf{z})$

$$\partial_{z_1} f_\mu(\mathbf{z}) = \mathbb{P}[A \text{ is endpoint}]$$

From root-free-ness to fast algorithm: intuition



- $\mu \sim \binom{[n]}{k} \overset{\text{encode}}{\longleftrightarrow} f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) \prod_{i \in S} z_i$
- Counting $\equiv$ Compute $f_\mu(\mathbf{1})$
- $f_\mu(\mathbf{z}) = 0 \longleftrightarrow \log f_\mu(\mathbf{z})$ singular $\Rightarrow$ abrupt change of derivatives of $\log f_\mu(\mathbf{z})$
- No roots "near" 1 $\Rightarrow$ "easy" to approximate f_μ

How to make it concrete?

Generating polynomial

For distribution $\mu : \binom{[n]}{k} \to \mathbb{R}_{\geq 0}$, let its **generating polynomial** be

$$f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) z^S = \sum_S \mu(S) \prod_{i \in S} z_i = \langle T_\mu, \mathbf{z}^{\otimes k} \rangle$$

where T_μ is the tensor defined by
$T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.

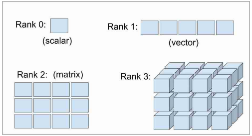
Generating polynomial

For distribution $\mu : \binom{[n]}{k} \to \mathbb{R}_{\geq 0}$, let its **generating polynomial** be

$$f_\mu(z_1, \dots, z_n) = \sum_S \mu(S) z^S = \sum_S \mu(S) \prod_{i \in S} z_i = \langle T_\mu, \mathbf{z}^{\otimes k} \rangle$$

where T_μ is the tensor defined by

$$T_\mu(i_1, \dots, i_k) = \mu(\{i_1, \dots, i_k\})/k!.$$



Generating polynomial

For distribution $\mu : \binom{[n]}{k} \to \mathbb{R}_{\geq 0}$, let its **generating polynomial** be

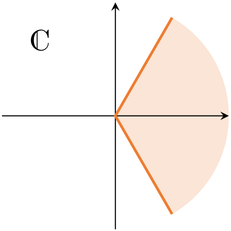
$$f_\mu(z_1, \ldots, z_n) = \sum_S \mu(S) z^S = \sum_S \mu(S) \prod_{i \in S} z_i = \langle T_\mu, \mathbf{z}^{\otimes k} \rangle$$

where T_μ is the tensor defined by
$T_\mu(i_1, \ldots, i_k) = \mu(\{i_1, \ldots, i_k\})/k!$.

Question

f_μ no roots near $\mathbb{R}^n_+ \Rightarrow$ Efficient Sampling from μ?

Sector stable polynomial

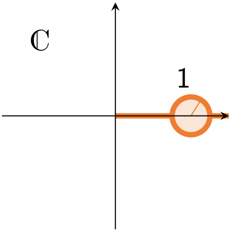


Definition

f is *α-sector-stable* if $f(z) \neq 0$ for z in

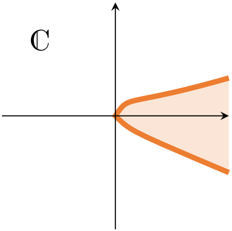
$$S_\alpha = \{z \in \mathbb{C}^* \mid |\arg(z)| < \alpha\pi/2\}$$

Generalization



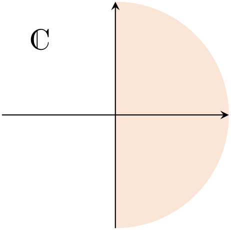
μ is spectral independence
if $f_\mu \neq 0$ on $(\mathbb{D}(1, \epsilon) \cup \mathbb{R}_+)^n$

Generalization



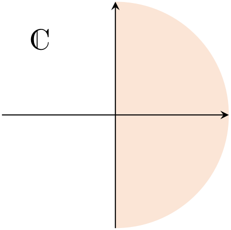
μ is spectral independence [Chen-Liu-Vigoda–FOCS'21]: if $f_\mu \neq 0$ on infinite regions

Example of α-Sector-Stable Distributions



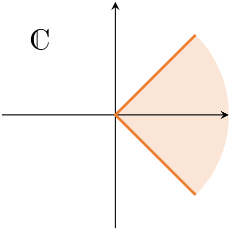
- (Non-homogeneous) endpoint distribution is 1-sector-stable (Hurwitz stable) [Hellman-Lieb'72]

Example of α-Sector-Stable Distributions



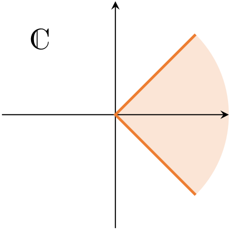
- (Non-homogeneous) endpoint distribution is 1-sector-stable (Hurwitz stable) [Hellman-Lieb'72]
- 1-sector stable $\equiv$ half-plane stable (circular region): has been studied by [Lee-Yang'52,Borcea-Branden'09]

Example of α-Sector-Stable Distributions



- (Non-homogeneous) endpoint distribution is 1-sector-stable (Hurwitz stable) [Hellman-Lieb'72]
- 1-sector stable $\equiv$ half-plane stable (circular region): has been studied by [Lee-Yang'52,Borcea-Branden'09]
- Homogenegous endpoint distribution is 1/2-sector-stable [this work]

Example of α-Sector-Stable Distributions



- (Non-homogeneous) endpoint distribution is 1-sector-stable (Hurwitz stable) [Hellman-Lieb'72]
- 1-sector stable $\equiv$ half-plane stable (circular region): has been studied by [Lee-Yang'52,Borcea-Branden'09]
- Homogenegous endpoint distribution is 1/2-sector-stable [this work]
- Many things to explore

Sector stable $\Rightarrow$ Spectral Independence

Ψ^{cor} : correlation matrix

$$\Psi_{\mu}^{\text{cor}}(i,j) = \mathbb{P}_{S\sim\mu}[j \in S \mid i \in S]$$

Sector stable $\Rightarrow$ Spectral Independence

Ψ^{cor} : correlation matrix

$$\Psi_\mu^{\text{cor}}(i,j) = \mathbb{P}_{S\sim\mu}[j \in S \mid i \in S]$$

Theorem (Main technical)

If μ is α-sector stable, then $\forall \lambda \in \mathbb{R}^n_{\geq 0}$, $\|\Psi_\mu^{cor}\|_1 \leq 2/\alpha$.

Sector stable $\Rightarrow$ Spectral Independence

Ψ^{cor} : correlation matrix

$$\Psi^{\text{cor}}_{\mu}(i,j) = \mathbb{P}_{S\sim\mu}[j \in S \mid i \in S]$$

Theorem (Main technical)

If μ is α-sector stable, then $\forall \lambda \in \mathbb{R}^n_{\geq 0}$, $\underbrace{\|\Psi^{cor}_{\lambda*\mu}\| \leq 2/\alpha.}_{\frac{\alpha}{2}-FLC}$

Spectral independence $\equiv \{\|\Psi^{\text{cor}}_{\mu(|S)}\|_2 \leq O(1)\forall S\}$

If μ is sector stable, then $\lambda * \mu$ (defined by $\lambda * \mu(S) \propto \mu(S)\prod_{i\in S}\lambda_i$) is sector stable

Fractional log-concave (FLC) $\equiv \lambda * \mu$ spectrally independent for all external field $\lambda \in \mathbb{R}^n_{\geq 0}$.

Sector stable $\Rightarrow$ Spectral Independence

Ψ^{cor} : correlation matrix

$$\Psi^{\text{cor}}_{\mu}(i,j) = \mathbb{P}_{S\sim\mu}[j \in S \mid i \in S]$$

Theorem (Main technical)

If μ is α-sector stable, then $\forall\lambda \in \mathbb{R}^n_{\geq 0}$,

Spectral independence $\equiv \{\|\Psi^{\text{cor}}_{\mu(|S)}\|_2 \leq O(1)\forall S\}$

If μ is sector stable, then $\lambda * \mu$ (defined by $\lambda * \mu(S) \propto \mu(S)\prod_{i\in S}\lambda_i$) is sector stable

Fractional log-concave (FLC) $\equiv \lambda * \mu$ spectrally independent for all external field $\lambda \in \mathbb{R}^n_{\geq 0}$.

Geometry of polynomial view of fractional log-concavity (FLC)

μ is α-fractionally log concave $\approx$

$$f_\mu(z_1, \ldots, z_n)^{\frac{1}{k\alpha}} \leq \sum_{i=1}^{n} p_i z_i^{1/\alpha}$$

with $p_i = \frac{1}{k}\frac{\partial f}{\partial z_i}(\vec{1})$

Geometry of polynomial view of fractional log-concavity (FLC)

μ is α-fractionally log concave $\approx$

$$\langle T_\mu, \mathbf{z}^{\otimes k}\rangle^{\frac{1}{k\alpha}} = f_\mu(z_1, \ldots, z_n)^{\frac{1}{k\alpha}} \leq \sum_{i=1}^{n} p_i z_i^{1/\alpha} \approx \|z\|_{1/\alpha}^{1/\alpha}$$

with $p_i = \frac{1}{k}\frac{\partial f}{\partial z_i}(\vec{1})$

Proof of main technical theorem

- Complex Analysis

$$\bigcirc \underset{\phi(0)=0}{\overset{\phi}{\rightsquigarrow}} \bigcirc$$

Proof of main technical theorem

- Complex Analysis

$$\bigcirc \overset{\phi}{\rightsquigarrow} \bigcirc$$

$$\phi(0) = 0$$

$$\Rightarrow |\phi'(0)| \leq 1$$

Proof of main technical theorem

- Complex Analysis
- Write $\|\Psi_\mu^{\text{cor}}(i, \cdot)\|_1 = \phi'(0)$ for ϕ' holomorphic

Proof of main technical theorem

- Complex Analysis
- Write $\|\Psi_{\mu}^{\text{cor}}(i,\cdot)\|_1 = \phi'(0)$ for ϕ' holomorphic
- Use Schwarz's lemma to bound $\phi'(0)$.

Overview

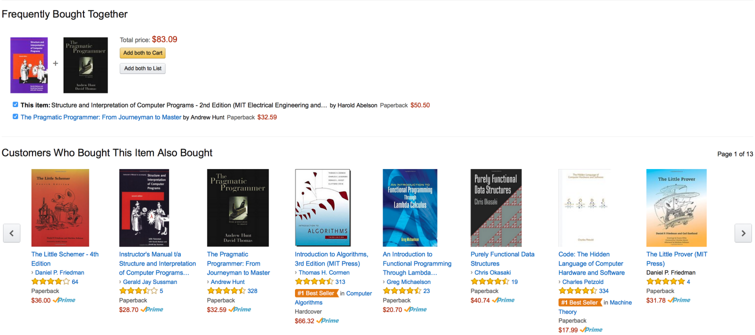
- Local Markov chains to sample from determinantal point processes (DPP).

- Local Markov chains to sample from determinantal point processes (DPP).

$$\mu(S) = \det(L_{S,S}) \forall S \subseteq [n].$$

$$\begin{pmatrix} & & 1 & & 1 & 0 \\ & & 5 & & 2 & 4 \\ 4 & 8 & 9 & 5 & 3 & 3 \\ & & 9 & & 2 & 3 \\ 3 & 7 & 9 & 5 & 3 & 3 \\ 4 & 8 & 6 & 1 & 3 & 0 \end{pmatrix}$$

- Local Markov chains to sample from determinantal point processes (DPP).



- Local Markov chains to sample from nonsymmetric determinantal point processes (DPP) [Gartrell-Brunel'20] with kernel L when $L + L^T$ PSD.

- Local Markov chains to sample from nonsymmetric determinantal point processes (DPP).

- Local Markov chains to sample from nonsymmetric determinantal point processes (DPP).

- Local Markov chains to sample from nonsymmetric determinantal point processes (DPP).
- Efficient algorithm for counting/sampling DPP intersects with partition constraints

- Local Markov chains to sample from nonsymmetric determinantal point processes (DPP).
- Efficient algorithm for counting/sampling DPP intersects with partition constraints
  Generally: the intersection of Rayleigh matroid and partition matroid of constantly many partitions.

- Local Markov chains to sample from nonsymmetric determinantal point processes (DPP).
- Efficient algorithm for counting/sampling DPP intersects with partition constraints
- Fast mixing $\Rightarrow$ MAP-inference via local search [Anari-**V.**'21]