Low-Degree Polynomials: Overview and Recent Developments

Alex Wein

University of California, Davis

New survey on arXiv, “Computational Complexity of Statistics: New Insights from Low-Degree Polynomials”

arXiv:2506.10748

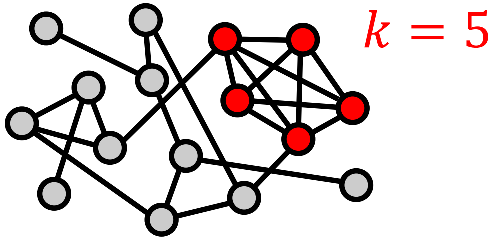
“High-Dimensional Statistics”

“High-Dimensional Statistics”

- Planted ***signal*** (e.g. low rank or sparse structure) in high-dimensional random ***noise*** (e.g. large random matrix/graph)

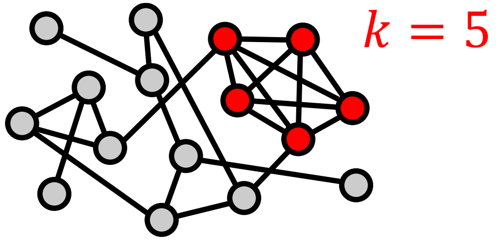
“High-Dimensional Statistics”

- Planted ***signal*** (e.g. low rank or sparse structure) in high-dimensional random ***noise*** (e.g. large random matrix/graph)
- Example: planted clique problem
  - $G(n, 1/2) + \{k\text{-clique}\}$
  - Goal: find the k-clique, w.h.p.

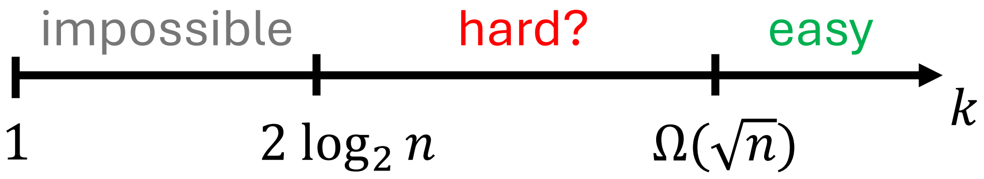


“High-Dimensional Statistics”

- Planted ***signal*** (e.g. low rank or sparse structure) in high-dimensional random ***noise*** (e.g. large random matrix/graph)



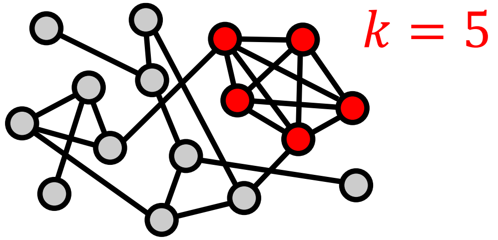
- Example: planted clique problem
  - $G(n, 1/2) + \{k\text{-clique}\}$
  - Goal: find the k-clique, w.h.p.

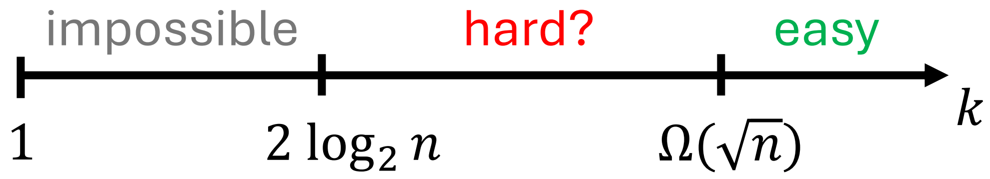


- Statistical-computational gap
  - Impossible: Any estimator fails [Arias-Castro, Verzelen ‘14]
  - Easy: Poly-time algorithm exists [Alon, Krivelevich, Sudakov ‘98]
  - Hard (?): Possible by “brute force” but no poly-time algorithm known

“High-Dimensional Statistics”

- Planted ***signal*** (e.g. low rank or sparse structure) in high-dimensional random ***noise*** (e.g. large random matrix/graph)
- Example: planted clique problem
  - $G(n, 1/2) + \{k\text{-clique}\}$
  - Goal: find the k-clique, w.h.p.

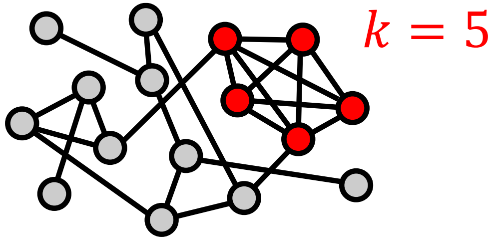




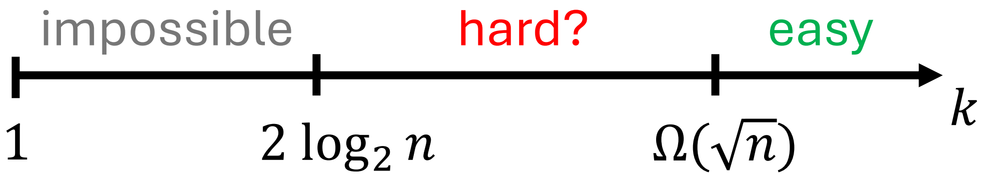
- Statistical-computational gap
  - Impossible: Any estimator fails [Arias-Castro, Verzelen ‘14]
  - Easy: Poly-time algorithm exists [Alon, Krivelevich, Sudakov ‘98]
  - Hard (?): Possible by “brute force” but no poly-time algorithm known
  - How to prove the hardness is fundamental…

"High-Dimensional Statistics"

- Planted ***signal*** (e.g. low rank or sparse structure) in high-dimensional random ***noise*** (e.g. large random matrix/graph)
- Example: planted clique problem
  - $G(n, 1/2) + \{k\text{-clique}\}$
  - Goal: find the k-clique, w.h.p.



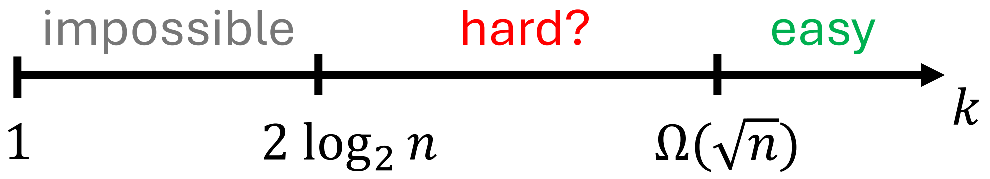
- Statistical-computational gap



  - Impossible: Any estimator fails [Arias-Castro, Verzelen '14]
  - Easy: Poly-time algorithm exists [Alon, Krivelevich, Sudakov '98]
  - Hard (?): Possible by "brute force" but no poly-time algorithm known
  - How to prove the hardness is fundamental…
- Other examples: sparse PCA, community detection, clustering, …

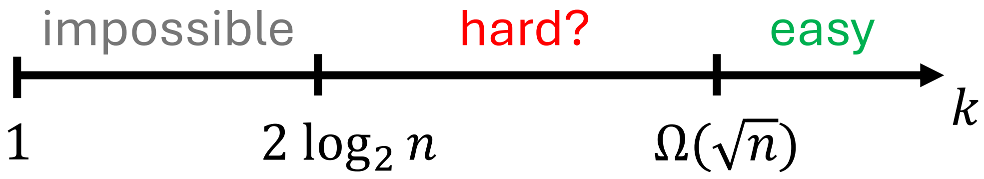
Hard Regime

Hard Regime



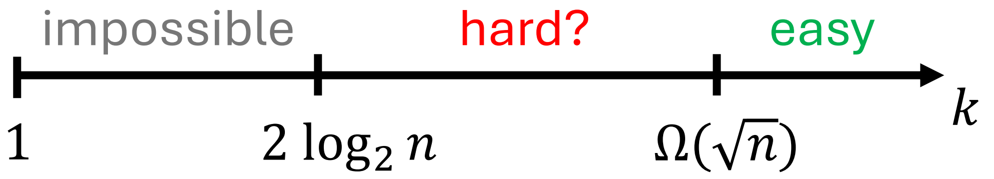
- In what sense can we prove the “hard” regime is hard?

Hard Regime



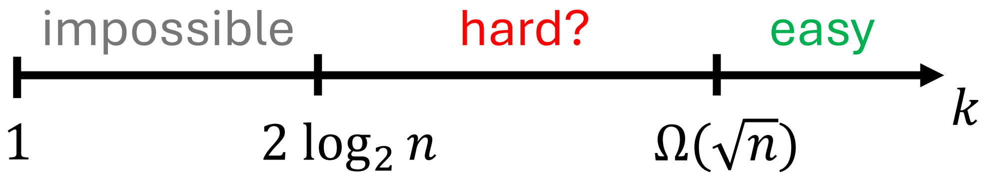
- In what sense can we prove the “hard” regime is hard?
- This is *average-case*, so (worst-case) NP-hardness does not apply

Hard Regime



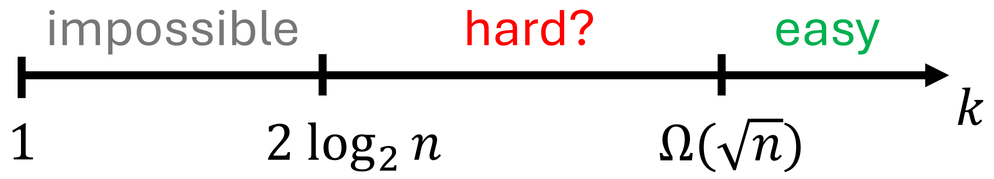
- In what sense can we prove the "hard" regime is hard?
- This is *average-case*, so (worst-case) NP-hardness does not apply
  - Max clique is NP-hard but can still solve planted clique for some k

Hard Regime



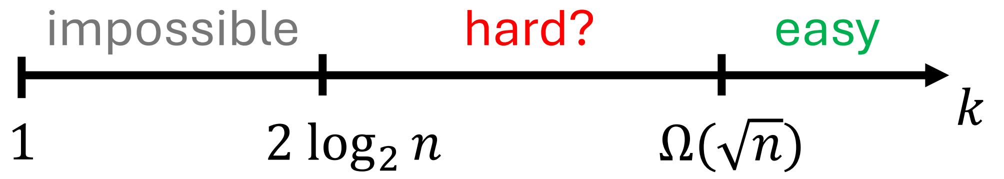
- In what sense can we prove the "hard" regime is hard?
- This is *average-case*, so (worst-case) NP-hardness does not apply
  - Max clique is NP-hard but can still solve planted clique for some k
- Instead, various approaches to average-case hardness:

Hard Regime



- In what sense can we prove the “hard” regime is hard?
- This is *average-case*, so (worst-case) NP-hardness does not apply
  - Max clique is NP-hard but can still solve planted clique for some k
- Instead, various approaches to average-case hardness:
  - Conditional hardness via reductions
    - Popular starting assumptions: planted clique conjecture, shortest vector on lattices, …

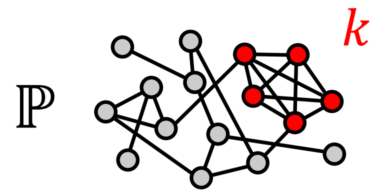
Hard Regime



- In what sense can we prove the "hard" regime is hard?
- This is *average-case*, so (worst-case) NP-hardness does not apply
  - Max clique is NP-hard but can still solve planted clique for some k
- Instead, various approaches to average-case hardness:
  - Conditional hardness via reductions
    - Popular starting assumptions: planted clique conjecture, shortest vector on lattices, …
  - Unconditional failure of restricted classes of algorithms
    - Sum-of-squares hierarchy (SOS)
    - Statistical query model (SQ)
    - Approximate message passing (AMP)
    - Overlap gap property (OGP)
    - …
    - Low-degree polynomials (main focus)

Objectives: Detection vs Recovery

Objectives: Detection vs Recovery

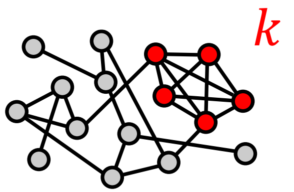


- Planted distribution $\mathbb{P}$: $G(n, 1/2)\ +\ \{\text{random } k\text{-clique}\}$

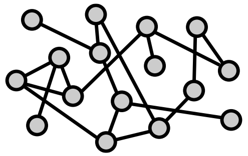
Objectives: Detection vs Recovery

- Planted distribution $\mathbb{P}$: $G(n, 1/2) \; + \; \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$

$\mathbb{P}$



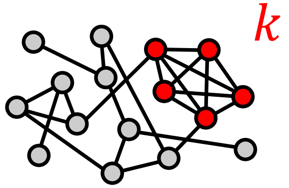
$\mathbb{Q}$



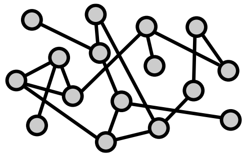
Objectives: Detection vs Recovery

- Planted distribution $\mathbb{P}$: $G(n, 1/2) \; + \; \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$
- <u>Recovery</u>: Under $\mathbb{P}$, find the planted clique

$\mathbb{P}$

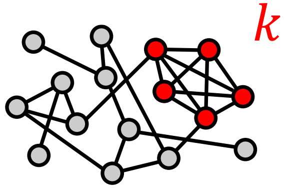


$\mathbb{Q}$

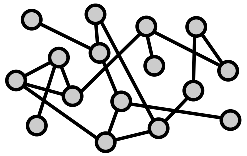


Objectives: Detection vs Recovery

$\mathbb{P}$

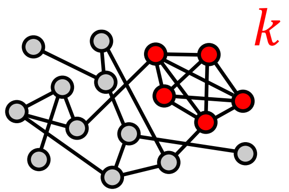


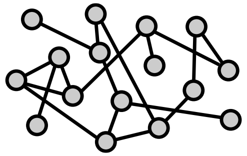
$\mathbb{Q}$



- Planted distribution $\mathbb{P}$: $G(n, 1/2)\ +\ \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$
- <u>Recovery</u>: Under $\mathbb{P}$, find the planted clique
  - *Exact recovery*: Recover the k vertices exactly, w.h.p. $1 - o(1)$ as $n \to \infty$

Objectives: Detection vs Recovery

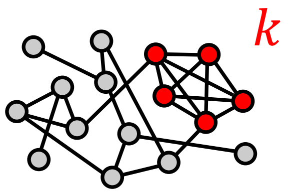
$\mathbb{P}$


$\mathbb{Q}$


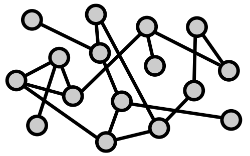
- Planted distribution $\mathbb{P}$: $G(n, 1/2)\ +\ \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$
- Recovery: Under $\mathbb{P}$, find the planted clique
  - *Exact recovery*: Recover the k vertices exactly, w.h.p. $1 - o(1)$ as $n \to \infty$
- Detection: Distinguish $\mathbb{P}$ vs $\mathbb{Q}$ (hypothesis testing)

Objectives: Detection vs Recovery

$\mathbb{P}$



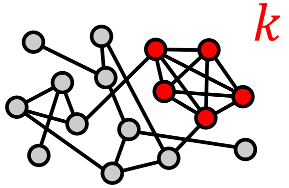
$\mathbb{Q}$



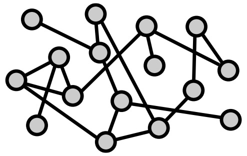
- Planted distribution $\mathbb{P}$: $G(n, 1/2) \; + \; \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$
- <u>Recovery</u>: Under $\mathbb{P}$, find the planted clique
  - *Exact recovery*: Recover the k vertices exactly, w.h.p. $1 - o(1)$ as $n \to \infty$
- <u>Detection</u>: Distinguish $\mathbb{P}$ vs $\mathbb{Q}$ (hypothesis testing)
  - *Strong detection*: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$, w.h.p.
  - *Weak detection*: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$, w.p. $\geq \frac{1}{2} + \Omega(1)$

Objectives: Detection vs Recovery

$\mathbb{P}$

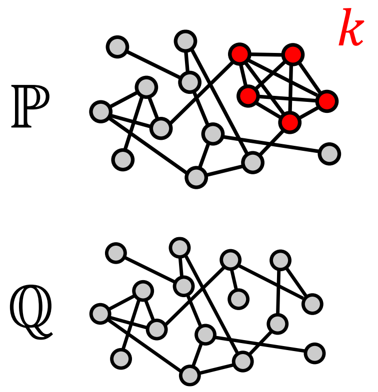


$\mathbb{Q}$

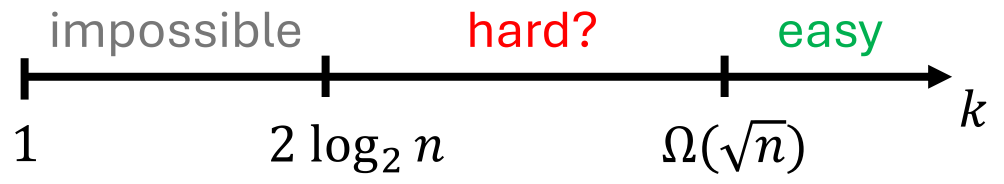


- Planted distribution $\mathbb{P}$: $G(n, 1/2) + \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$
- Recovery: Under $\mathbb{P}$, find the planted clique
  - *Exact recovery*: Recover the k vertices exactly, w.h.p. $1 - o(1)$ as $n \to \infty$
- Detection: Distinguish $\mathbb{P}$ vs $\mathbb{Q}$ (hypothesis testing)
  - *Strong detection*: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$, w.h.p.
  - *Weak detection*: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$, w.p. $\geq \frac{1}{2} + \Omega(1)$
- Generally, recovery is more difficult than detection

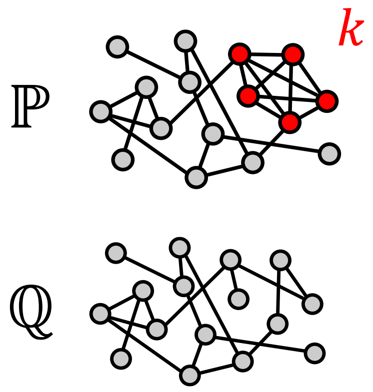
Objectives: Detection vs Recovery



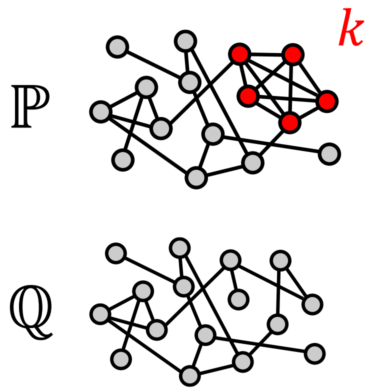
- Planted distribution $\mathbb{P}$: $G(n, 1/2)\ +\ \{\text{random } k\text{-clique}\}$
- Null distribution $\mathbb{Q}$: $G(n, 1/2)$
- <u>Recovery</u>: Under $\mathbb{P}$, find the planted clique
  - *Exact recovery*: Recover the k vertices exactly, w.h.p. $1 - o(1)$ as $n \to \infty$
- <u>Detection</u>: Distinguish $\mathbb{P}$ vs $\mathbb{Q}$ (hypothesis testing)
  - *Strong detection*: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$, w.h.p.
  - *Weak detection*: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$, w.p. $\geq \frac{1}{2} + \Omega(1)$
- Generally, recovery is more difficult than detection
- For planted clique, both have the same thresholds



What Are the Best Known Algorithms?

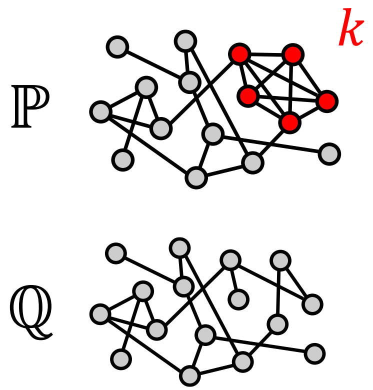


What Are the Best Known Algorithms?



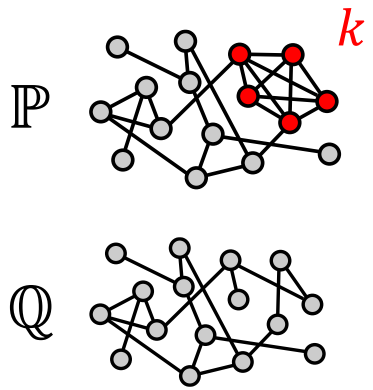
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$

What Are the Best Known Algorithms?



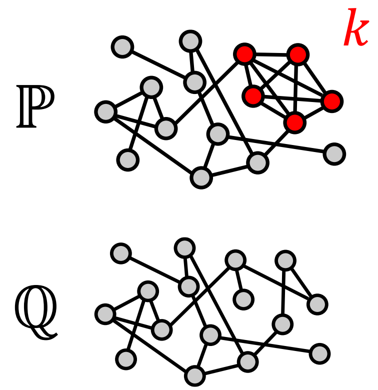
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$

What Are the Best Known Algorithms?



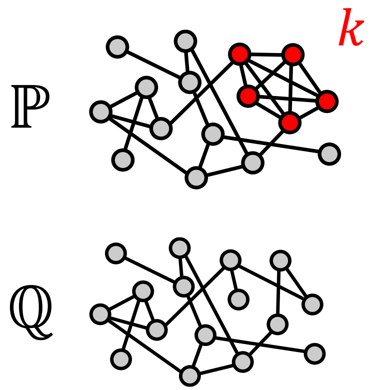
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)

What Are the Best Known Algorithms?



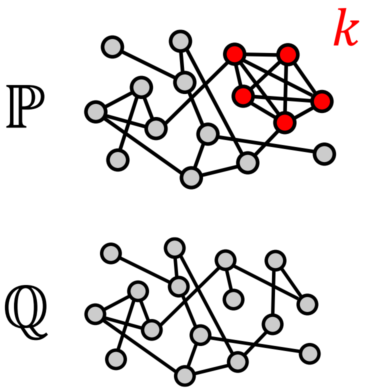
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)
- Recovery: Max-degree vertices works when $k \geq c\sqrt{n \log n}$

What Are the Best Known Algorithms?



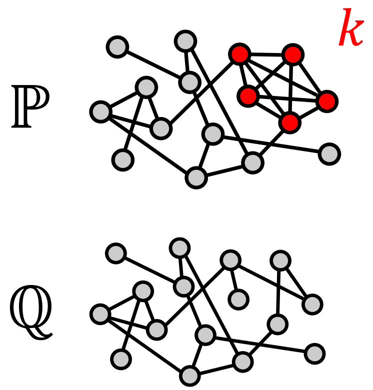
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)
- Recovery: Max-degree vertices works when $k \geq c\sqrt{n \log n}$
  - Degree of vertex i: $\sum_j Y_{ij}$ (degree-1 polynomial)

What Are the Best Known Algorithms?



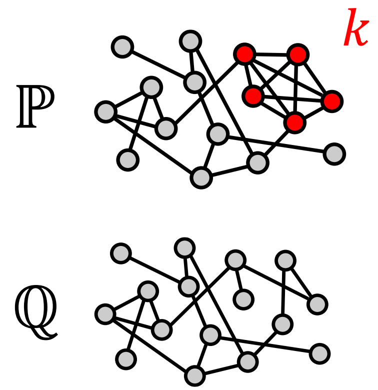
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)
- Recovery: Max-degree vertices works when $k \geq c\sqrt{n \log n}$
  - Degree of vertex i: $\sum_j Y_{ij}$ (degree-1 polynomial)
- To get $k \geq c\sqrt{n}$, use leading eigenvalue/eigenvector of signed adjacency matrix $A = (A_{ij})$ where $A_{ij} = 2Y_{ij} - 1 \in \{\pm 1\}$

What Are the Best Known Algorithms?



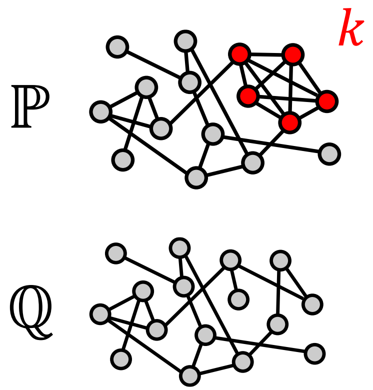
- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)
- Recovery: Max-degree vertices works when $k \geq c\sqrt{n \log n}$
  - Degree of vertex i: $\sum_j Y_{ij}$ (degree-1 polynomial)
- To get $k \geq c\sqrt{n}$, use leading eigenvalue/eigenvector of signed adjacency matrix $A = \left(A_{ij}\right)$ where $A_{ij} = 2Y_{ij} - 1 \in \{\pm 1\}$
  - $\mathrm{Tr}(A^{2p}) = \sum_i \lambda_i^{2p} \approx \lambda_{\max}^{2p}$ for $p = \Theta(\log n)$ (degree-$2p$ polynomial)

What Are the Best Known Algorithms?



- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)
- Recovery: Max-degree vertices works when $k \geq c\sqrt{n \log n}$
  - Degree of vertex i: $\sum_j Y_{ij}$ (degree-1 polynomial)
- To get $k \geq c\sqrt{n}$, use leading eigenvalue/eigenvector of signed adjacency matrix $A = (A_{ij})$ where $A_{ij} = 2Y_{ij} - 1 \in \{\pm 1\}$
  - $\mathrm{Tr}(A^{2p}) = \sum_i \lambda_i^{2p} \approx \lambda_{\max}^{2p}$ for $p = \Theta(\log n)$ (degree-$2p$ polynomial)
- Polynomials of “low” degree $O(\log n)$ capture various algorithms…

What Are the Best Known Algorithms?



- Input variables: $\binom{n}{2}$ edge-indicators $Y_{ij} \in \{0,1\}$ for $i < j$
- Detection: Total edge count works for $k = \omega(\sqrt{n})$
  - $f(Y) = \sum_{i<j} Y_{ij}$ (degree-1 polynomial)
- Recovery: Max-degree vertices works when $k \geq c\sqrt{n \log n}$
  - Degree of vertex i: $\sum_j Y_{ij}$ (degree-1 polynomial)
- To get $k \geq c\sqrt{n}$, use leading eigenvalue/eigenvector of signed adjacency matrix $A = (A_{ij})$ where $A_{ij} = 2Y_{ij} - 1 \in \{\pm 1\}$
  - $\mathrm{Tr}(A^{2p}) = \sum_i \lambda_i^{2p} \approx \lambda_{\max}^{2p}$ for $p = \Theta(\log n)$ (degree-$2p$ polynomial)
- Polynomials of "low" degree $O(\log n)$ capture various algorithms…
  - Spectral methods; AMP; subgraph counts, e.g. triangle count $\sum_{i<j<\ell} Y_{ij} Y_{i\ell} Y_{j\ell}$

Low-Degree Hardness

Low-Degree Hardness

- Prove failure of degree-$O(\log n)$ polynomials in the “hard” regime, as a concrete form of hardness

Low-Degree Hardness

- Prove failure of degree-$O(\log n)$ polynomials in the “hard” regime, as a concrete form of hardness
- Idea arose from sum-of-squares, but can also be motivated directly
  [Barak, Hopkins, Kelner, Kothari, Moitra, Potechin ‘16]
  [Hopkins, Steurer ‘17]
  [Hopkins, Kothari, Potechin, Raghavendra, Schramm, Steurer ‘17]
  [Hopkins ‘18]

Low-Degree Hardness

- Prove failure of degree-$O(\log n)$ polynomials in the “hard” regime, as a concrete form of hardness
- Idea arose from sum-of-squares, but can also be motivated directly
  [Barak, Hopkins, Kelner, Kothari, Moitra, Potechin ‘16]
  [Hopkins, Steurer ‘17]
  [Hopkins, Kothari, Potechin, Raghavendra, Schramm, Steurer ‘17]
  [Hopkins ‘18]
- **Objection:** *Are low-degree polynomials efficiently computable?*

Low-Degree Hardness

- Prove failure of degree-$O(\log n)$ polynomials in the “hard” regime, as a concrete form of hardness
- Idea arose from sum-of-squares, but can also be motivated directly
  [Barak, Hopkins, Kelner, Kothari, Moitra, Potechin ‘16]
  [Hopkins, Steurer ‘17]
  [Hopkins, Kothari, Potechin, Raghavendra, Schramm, Steurer ‘17]
  [Hopkins ‘18]
- **Objection:** *Are low-degree polynomials efficiently computable?*
  - A degree-D polynomial in $n^{\Theta(1)}$ variables has $n^{O(D)}$ terms

Low-Degree Hardness

- Prove failure of degree-$O(\log n)$ polynomials in the "hard" regime, as a concrete form of hardness
- Idea arose from sum-of-squares, but can also be motivated directly
  [Barak, Hopkins, Kelner, Kothari, Moitra, Potechin '16]
  [Hopkins, Steurer '17]
  [Hopkins, Kothari, Potechin, Raghavendra, Schramm, Steurer '17]
  [Hopkins '18]
- **Objection:** *Are low-degree polynomials efficiently computable?*
  - A degree-D polynomial in $n^{\Theta(1)}$ variables has $n^{O(D)}$ terms
  - So a degree-$O(1)$ polynomial can be computed in poly time, but a degree-$O(\log n)$ polynomial cannot in general

Low-Degree Hardness

- Prove failure of degree-$O(\log n)$ polynomials in the "hard" regime, as a concrete form of hardness
- Idea arose from sum-of-squares, but can also be motivated directly
  [Barak, Hopkins, Kelner, Kothari, Moitra, Potechin '16]
  [Hopkins, Steurer '17]
  [Hopkins, Kothari, Potechin, Raghavendra, Schramm, Steurer '17]
  [Hopkins '18]
- **Objection:** *Are low-degree polynomials efficiently computable?*
  - A degree-D polynomial in $n^{\Theta(1)}$ variables has $n^{O(D)}$ terms
  - So a degree-$O(1)$ polynomial can be computed in poly time, but a degree-$O(\log n)$ polynomial cannot in general
  - The point is: degree-$O(\log n)$ polynomials capture important classes of poly-time algorithms, so if they fail, this rules out various approaches

Low-Degree Hardness (Recovery)

Low-Degree Hardness (Recovery)

- Let’s prove that all low-degree polynomials fail in the “hard” regime

Low-Degree Hardness (Recovery)

- Let’s prove that all low-degree polynomials fail in the “hard” regime
- Define “success” for low-degree polynomials, starting with <u>recovery</u>:

Low-Degree Hardness (Recovery)

- Let's prove that all low-degree polynomials fail in the "hard" regime
- Define "success" for low-degree polynomials, starting with recovery:
  - Aim to estimate a scalar value, $x \coloneqq \mathbb{1}_{1 \in \text{clique}}$ (is vertex 1 in the clique?)

Low-Degree Hardness (Recovery)

- Let's prove that all low-degree polynomials fail in the "hard" regime
- Define "success" for low-degree polynomials, starting with <u>recovery</u>:
  - Aim to estimate a scalar value, $x \coloneqq \mathbb{1}_{1 \in \text{clique}}$ (is vertex 1 in the clique?)
  - Low-degree estimator: polynomial $f : \{0,1\}^{\binom{n}{2}} \to \mathbb{R}$ of degree $\leq D = D_n$

Low-Degree Hardness (Recovery)

- Let's prove that all low-degree polynomials fail in the "hard" regime
- Define "success" for low-degree polynomials, starting with recovery:
  - Aim to estimate a scalar value, $x := \mathbb{1}_{1 \in \text{clique}}$ (is vertex 1 in the clique?)
  - Low-degree estimator: polynomial $f: \{0,1\}^{\binom{n}{2}} \to \mathbb{R}$ of degree $\leq D = D_n$
  - $\text{MMSE}_{\leq D} := \inf_{f \text{ deg } D} \mathrm{E}_{\mathbb{P}}[(f(Y) - x)^2]$

Low-Degree Hardness (Recovery)

- Let's prove that all low-degree polynomials fail in the "hard" regime
- Define "success" for low-degree polynomials, starting with <u>recovery</u>:
  - Aim to estimate a scalar value, $x := \mathbb{1}_{1 \in \text{clique}}$ (is vertex 1 in the clique?)
  - Low-degree estimator: polynomial $f: \{0,1\}^{\binom{n}{2}} \to \mathbb{R}$ of degree $\le D = D_n$
  - $\text{MMSE}_{\le D} := \inf_{f \text{ deg } D} \mathrm{E}_{\mathbb{P}}[(f(Y) - x)^2]$

Theorem [Schramm, W '20] In the planted clique model,

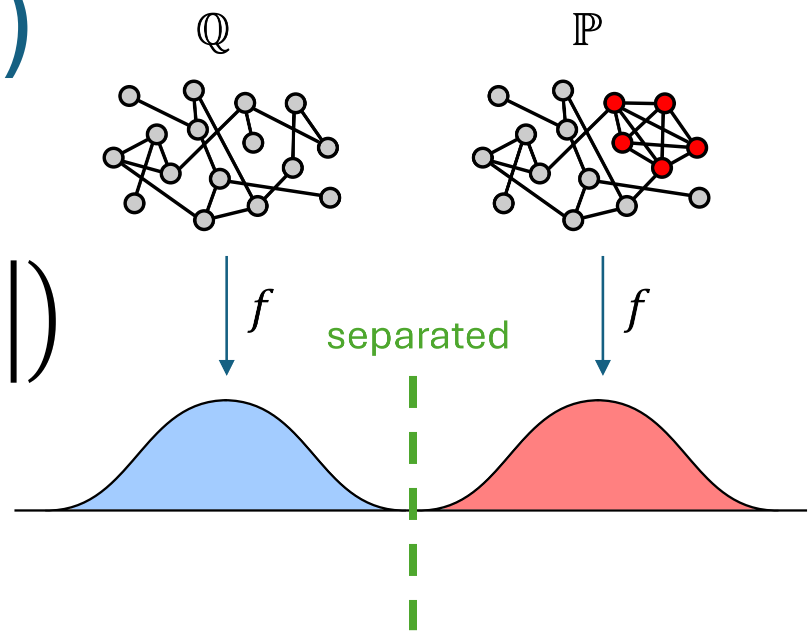
- (Hard) If $k \le n^{1/2-\Omega(1)}$ then $\text{MMSE}_{\le O(\log n)} = (1 - o(1))\text{Var}(x)$
  - No better than the trivial degree-0 estimator $f(Y) = \mathrm{E}[x]$
- (Easy) If $k \ge n^{1/2+\Omega(1)}$ then $\text{MMSE}_{\le O(1)} = o(1/n)$
  - Small enough to guarantee exact recovery

Low-Degree Hardness (Detection)

Low-Degree Hardness (Detection)

- **Definition:** f *strongly separates* $\mathbb{P}$ and $\mathbb{Q}$ if

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

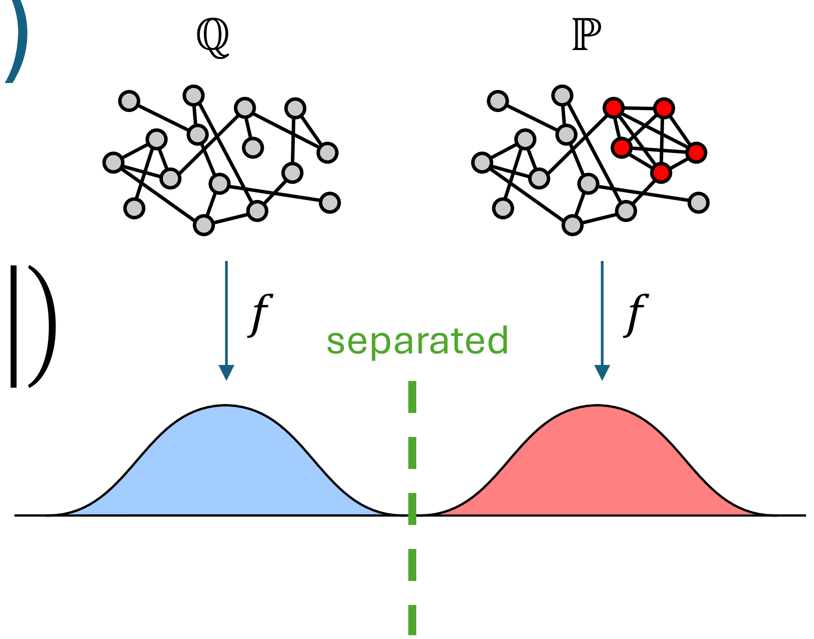


Low-Degree Hardness (Detection)

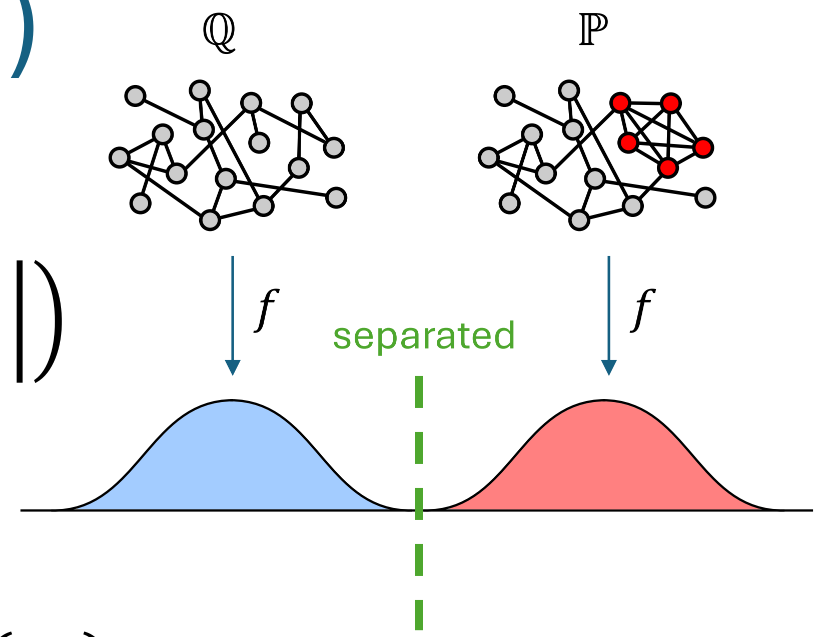
- **Definition:** f *strongly separates* $\mathbb{P}$ and $\mathbb{Q}$ if

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

- Strong separation $\Rightarrow$ strong detection
  - Proof: Chebyshev

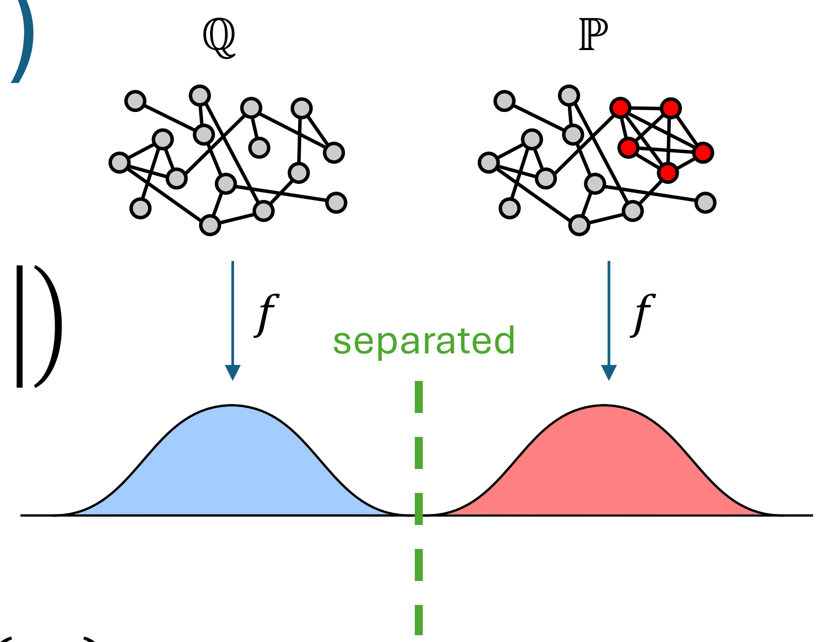


Low-Degree Hardness (Detection)



- **Definition:** f *strongly separates* $\mathbb{P}$ and $\mathbb{Q}$ if
$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$
- Strong separation $\Rightarrow$ strong detection
  - Proof: Chebyshev
- Similarly, *weak separation -- replace* $o(\dots)$ *by* $O(\dots)$

Low-Degree Hardness (Detection)



- **Definition:** f *strongly separates* $\mathbb{P}$ and $\mathbb{Q}$ if

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

- Strong separation $\Rightarrow$ strong detection
    - Proof: Chebyshev
- Similarly, *weak separation -- replace* $o(\ldots)$ *by* $O(\ldots)$

Theorem [BHKKMP '16] In the planted clique model,

- (Hard) If $k \le n^{1/2-\Omega(1)}$, no degree-$O(\log n)$ polynomial strongly (or even weakly) separates $\mathbb{P}$ and $\mathbb{Q}$
- (Easy) If $k \ge n^{1/2+\Omega(1)}$, some degree-1 polynomial (edge count) strongly separates $\mathbb{P}$ and $\mathbb{Q}$

Why “Separation”?

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
    - Right, we’ve only ruled out the “natural analysis” via Chebyshev

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
  - Right, we’ve only ruled out the “natural analysis” via Chebyshev
  - Open: Rule out thresholding (PTF tests)

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
    - Right, we’ve only ruled out the “natural analysis” via Chebyshev
    - Open: Rule out thresholding (PTF tests)
    - Or even better, rule out *any* post-processing to a low-degree polynomial

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
  - Right, we’ve only ruled out the “natural analysis” via Chebyshev
  - Open: Rule out thresholding (PTF tests)
  - Or even better, rule out *any* post-processing to a low-degree polynomial
- Fair point, but I also want to defend “separation”…

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
  - Right, we’ve only ruled out the “natural analysis” via Chebyshev
  - Open: Rule out thresholding (PTF tests)
  - Or even better, rule out *any* post-processing to a low-degree polynomial
- Fair point, but I also want to defend “separation”…
  - Captures known upper bounds in the “easy” regime

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
    - Right, we’ve only ruled out the “natural analysis” via Chebyshev
    - Open: Rule out thresholding (PTF tests)
    - Or even better, rule out *any* post-processing to a low-degree polynomial
- Fair point, but I also want to defend “separation”…
    - Captures known upper bounds in the “easy” regime
    - We have widely-applicable tools to rule it out in the “hard” regime

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
    - Right, we’ve only ruled out the “natural analysis” via Chebyshev
    - Open: Rule out thresholding (PTF tests)
    - Or even better, rule out *any* post-processing to a low-degree polynomial
- Fair point, but I also want to defend “separation”…
    - Captures known upper bounds in the “easy” regime
    - We have widely-applicable tools to rule it out in the “hard” regime
    - Separation is analogous to $\mathrm{MMSE}_{\leq D}$

Why "Separation"?

- **Objection:** *Suppose we've ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
  - Right, we've only ruled out the "natural analysis" via Chebyshev
  - Open: Rule out thresholding (PTF tests)
  - Or even better, rule out *any* post-processing to a low-degree polynomial
- Fair point, but I also want to defend "separation"...
  - Captures known upper bounds in the "easy" regime
  - We have widely-applicable tools to rule it out in the "hard" regime
  - Separation is analogous to $\mathrm{MMSE}_{\le D}$
    - Strong separation is equivalent to $\mathrm{MMSE}_{\le D} = o(1)$ in the following recovery problem: draw $x \sim \{0,1\}$ uniformly, observe Y drawn from $\mathbb{P}$ (if $x = 1$) or $\mathbb{Q}$ (if $x = 0$), estimate x

Why “Separation”?

- **Objection:** *Suppose we’ve ruled out separation. It might still be possible to detect by thresholding a low-degree polynomial.*
  - Right, we’ve only ruled out the “natural analysis” via Chebyshev
  - Open: Rule out thresholding (PTF tests)
  - Or even better, rule out *any* post-processing to a low-degree polynomial
- Fair point, but I also want to defend “separation”…
  - Captures known upper bounds in the “easy” regime
  - We have widely-applicable tools to rule it out in the “hard” regime
  - Separation is analogous to $\mathrm{MMSE}_{\leq D}$
    - Strong separation is equivalent to $\mathrm{MMSE}_{\leq D} = o(1)$ in the following recovery problem: draw $x \sim \{0,1\}$ uniformly, observe Y drawn from $\mathbb{P}$ (if $x = 1$) or $\mathbb{Q}$ (if $x = 0$), estimate x
    - $\mathrm{MMSE}_{\leq D}$ has the same flaw: even if you prove it’s large, you haven’t ruled out exact recovery by *thresholding* a polynomial

Degree-Runtime Correspondence

Degree-Runtime Correspondence

- One perspective: “Degree complexity” is an intrinsic measure of computational complexity, captures certain algorithms

Degree-Runtime Correspondence

- One perspective: “Degree complexity” is an intrinsic measure of computational complexity, captures certain algorithms
- More ambitious perspective: Polynomial degree is a proxy for runtime

Degree-Runtime Correspondence

- One perspective: "Degree complexity" is an intrinsic measure of computational complexity, captures certain algorithms
- More ambitious perspective: Polynomial degree is a proxy for runtime
- Heuristic: degree-$O(1)$ $\subseteq$ polynomial time $\subseteq$ degree-$O(\log n)$

Degree-Runtime Correspondence

- One perspective: "Degree complexity" is an intrinsic measure of computational complexity, captures certain algorithms
- More ambitious perspective: Polynomial degree is a proxy for runtime
- Heuristic: degree-$O(1)$ $\subseteq$ polynomial time $\subseteq$ degree-$O(\log n)$
- Some high-degree polynomials can be computed quickly…

Degree-Runtime Correspondence

- One perspective: "Degree complexity" is an intrinsic measure of computational complexity, captures certain algorithms
- More ambitious perspective: Polynomial degree is a proxy for runtime
- Heuristic: degree-$O(1)$ ⊆ polynomial time ⊆ degree-$O(\log n)$
- Some high-degree polynomials can be computed quickly…
- So, to argue that strong detection can't be done in polynomial time, we should prove that strong separation can't be done by degree-D polynomials, for some $D = \omega(\log n)$ or (ideally) higher

Degree-Runtime Correspondence

- One perspective: "Degree complexity" is an intrinsic measure of computational complexity, captures certain algorithms
- More ambitious perspective: Polynomial degree is a proxy for runtime
- Heuristic: degree-$O(1)$ $\subseteq$ polynomial time $\subseteq$ degree-$O(\log n)$
- Some high-degree polynomials can be computed quickly...
- So, to argue that strong detection can't be done in polynomial time, we should prove that strong separation can't be done by degree-D polynomials, for some $D = \omega(\log n)$ or (ideally) higher
- Ideally, also prove that some degree-$O(\log n)$ polynomial achieves strong separation in the "easy" regime

Degree-Runtime Correspondence

- One perspective: "Degree complexity" is an intrinsic measure of computational complexity, captures certain algorithms
- More ambitious perspective: Polynomial degree is a proxy for runtime
- Heuristic: degree-$O(1)$ $\subseteq$ polynomial time $\subseteq$ degree-$O(\log n)$
- Some high-degree polynomials can be computed quickly…
- So, to argue that strong detection can't be done in polynomial time, we should prove that strong separation can't be done by degree-D polynomials, for some $D = \omega(\log n)$ or (ideally) higher
- Ideally, also prove that some degree-$O(\log n)$ polynomial achieves strong separation in the "easy" regime
- Heuristic for higher runtimes: degree-$n^\delta \approx$ time-$\exp(n^{\delta \pm o(1)})$

Does Degree Really Track Runtime?

Does Degree Really Track Runtime?

- Yes, in many examples…
  - planted dense subgraph, community detection, graph matching, geometric graphs, …
  - sparse PCA, spiked Wigner/Wishart matrix, planted submatrix, group synchronization, …
  - tensor PCA, tensor decomposition, planted dense subhypergraph, …
  - sparse linear regression, non-gaussian component analysis, gaussian mixture models, …
  - … and more

Does Degree Really Track Runtime?

- Yes, in many examples…
  - planted dense subgraph, community detection, graph matching, geometric graphs, …
  - sparse PCA, spiked Wigner/Wishart matrix, planted submatrix, group synchronization, …
  - tensor PCA, tensor decomposition, planted dense subhypergraph, …
  - sparse linear regression, non-gaussian component analysis, gaussian mixture models, …
  - … and more

- But not in others…
  - XOR-SAT (gaussian elimination)
  - certain “noiseless” problems (LLL lattice basis reduction)
  - error-correcting codes
  - heavy-tailed noise
  - broadcasting on trees

Does Degree Really Track Runtime?

- Yes, in many examples…
  - planted dense subgraph, community detection, graph matching, geometric graphs, …
  - sparse PCA, spiked Wigner/Wishart matrix, planted submatrix, group synchronization, …
  - tensor PCA, tensor decomposition, planted dense subhypergraph, …
  - sparse linear regression, non-gaussian component analysis, gaussian mixture models, …
  - … and more

"success stories"

"counterexamples"

- But not in others…
  - XOR-SAT (gaussian elimination)
  - certain "noiseless" problems (LLL lattice basis reduction)
  - error-correcting codes
  - heavy-tailed noise
  - broadcasting on trees

Does Degree Really Track Runtime?

- Yes, in many examples…
  - planted dense subgraph, community detection, graph matching, geometric graphs, …
  - sparse PCA, spiked Wigner/Wishart matrix, planted submatrix, group synchronization, …
  - tensor PCA, tensor decomposition, planted dense subhypergraph, …
  - sparse linear regression, non-gaussian component analysis, gaussian mixture models, …
  - … and more

"success stories"

"counterexamples"

?

- But not in others…
  - XOR-SAT (gaussian elimination)
  - certain "noiseless" problems (LLL lattice basis reduction)
  - error-correcting codes
  - heavy-tailed noise
  - broadcasting on trees

Low-Degree Conjecture

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …
- Attempt by [Hopkins '18] recently refuted* [Buhai, Hsieh, Jain, Kothari '25]

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …
- Attempt by [Hopkins '18] recently refuted* [Buhai, Hsieh, Jain, Kothari '25]
  - And even if true, this conjecture covered a limited range of problems

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …
- Attempt by [Hopkins '18] recently refuted* [Buhai, Hsieh, Jain, Kothari '25]
  - And even if true, this conjecture covered a limited range of problems
- How to know if a new problem "counts"?

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …
- Attempt by [Hopkins '18] recently refuted* [Buhai, Hsieh, Jain, Kothari '25]
  - And even if true, this conjecture covered a limited range of problems
- How to know if a new problem "counts"?
  - More of an art than a science

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …
- Attempt by [Hopkins '18] recently refuted* [Buhai, Hsieh, Jain, Kothari '25]
  - And even if true, this conjecture covered a limited range of problems
- How to know if a new problem "counts"?
  - More of an art than a science
  - We learn more as we find more counterexamples

Low-Degree Conjecture

- The "low-degree conjecture" is the informal belief that: for some class of "natural statistical problems," if degree-$O(\log n)$ polynomials fail then so do all poly-time algorithms
- To make this formal, we need to specify the class of problems…
- Should be "noise-robust", "highly-symmetric", …
- Attempt by [Hopkins '18] recently refuted* [Buhai, Hsieh, Jain, Kothari '25]
  - And even if true, this conjecture covered a limited range of problems
- How to know if a new problem "counts"?
  - More of an art than a science
  - We learn more as we find more counterexamples
  - Should be cautious about conjectures

Proof Ideas (Detection)

Proof Ideas (Detection)

- How to rule out strong separation by a degree-D polynomial:

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

Proof Ideas (Detection)

- How to rule out strong separation by a degree-D polynomial:

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

- Sufficient:

$$\sup_{f \ \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]}{\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}}} = O(1)$$

Proof Ideas (Detection)

- How to rule out strong separation by a degree-D polynomial:

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

- Sufficient:

$$\sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]}{\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}}} = O(1)$$

- Sufficient:

$$\mathrm{Adv}_{\leq D} \coloneqq \sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = O(1)$$

Proof Ideas (Detection)

- How to rule out strong separation by a degree-D polynomial:

$$\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}} = o\left(\left|\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]\right|\right)$$

- Sufficient:

$$\sup_{f \text{ deg } D} \frac{\mathrm{E}_{\mathbb{P}}[f] - \mathrm{E}_{\mathbb{Q}}[f]}{\sqrt{\max\{\mathrm{Var}_{\mathbb{P}}(f), \mathrm{Var}_{\mathbb{Q}}(f)\}}} = O(1)$$

- Sufficient:

$$\mathrm{Adv}_{\leq D} := \sup_{f \text{ deg } D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = O(1)$$

- *"Advantage"* a.k.a. *"norm of the low-degree likelihood ratio"* $\|L^{\leq D}\|$

Bounding the “Advantage”

Bounding the "Advantage"

Fact: $\mathrm{Adv}_{\leq D} \coloneqq \sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = \sqrt{\sum_i \mathrm{E}_{\mathbb{P}}[h_i]^2}$ where $\{h_i\}$ is a basis of *orthonormal polynomials* for $\mathbb{R}[Y]_{\leq D}$ under $\mathbb{Q}$: $\mathrm{E}_{Y\sim\mathbb{Q}}\left[h_i(Y)h_j(Y)\right] = \delta_{ij}$

Bounding the "Advantage"

Fact: $\mathrm{Adv}_{\leq D} \coloneqq \sup_{f \text{ deg } D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = \sqrt{\sum_i \mathrm{E}_{\mathbb{P}}[h_i]^2}$ where $\{h_i\}$ is a basis of *orthonormal polynomials* for $\mathbb{R}[Y]_{\leq D}$ under $\mathbb{Q}$: $\mathrm{E}_{Y\sim\mathbb{Q}}\left[h_i(Y)h_j(Y)\right] = \delta_{ij}$

- E.g. If $Y \sim \mathbb{Q}$ has i.i.d. entries $Y_1, \ldots, Y_N \sim \mathrm{Unif}(\pm 1)$, orthonormal polynomials are monomials $1, Y_1, Y_2, Y_1Y_2, Y_1Y_3, Y_1Y_2Y_3, \ldots$ up to deg D

Bounding the "Advantage"

Fact: $\mathrm{Adv}_{\leq D} := \sup_{f \text{ deg } D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = \sqrt{\sum_i \mathrm{E}_{\mathbb{P}}[h_i]^2}$ where $\{h_i\}$ is a basis of *orthonormal polynomials* for $\mathbb{R}[Y]_{\leq D}$ under $\mathbb{Q}$: $\mathrm{E}_{Y\sim\mathbb{Q}}\left[h_i(Y)h_j(Y)\right] = \delta_{ij}$

- E.g. If $Y \sim \mathbb{Q}$ has i.i.d. entries $Y_1, \ldots, Y_N \sim \mathrm{Unif}(\pm 1)$, orthonormal polynomials are monomials $1, Y_1, Y_2, Y_1Y_2, Y_1Y_3, Y_1Y_2Y_3, \ldots$ up to deg D
- Generally, can construct orthogonal polynomials when $\mathbb{Q}$ is a product measure (independent coordinates) $\mathbb{Q}$

Bounding the "Advantage"

Fact: $\mathrm{Adv}_{\leq D} \coloneqq \sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = \sqrt{\sum_i \mathrm{E}_{\mathbb{P}}[h_i]^2}$ where $\{h_i\}$ is a basis of *orthonormal polynomials* for $\mathbb{R}[Y]_{\leq D}$ under $\mathbb{Q}$: $\mathrm{E}_{Y\sim\mathbb{Q}}\left[h_i(Y)h_j(Y)\right] = \delta_{ij}$

- E.g. If $Y \sim \mathbb{Q}$ has i.i.d. entries $Y_1, \ldots, Y_N \sim \mathrm{Unif}(\pm 1)$, orthonormal polynomials are monomials $1, Y_1, Y_2, Y_1 Y_2, Y_1 Y_3, Y_1 Y_2 Y_3, \ldots$ up to deg D
- Generally, can construct orthogonal polynomials when $\mathbb{Q}$ is a product measure (independent coordinates) $\mathbb{Q}$
    - First construct orthogonal polynomials for each coordinate…

Bounding the "Advantage"

Fact: $\mathrm{Adv}_{\leq D} \coloneqq \sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f]}{\sqrt{\mathrm{E}_{\mathbb{Q}}[f^2]}} = \sqrt{\sum_i \mathrm{E}_{\mathbb{P}}[h_i]^2}$ where $\{h_i\}$ is a basis of *orthonormal polynomials* for $\mathbb{R}[Y]_{\leq D}$ under $\mathbb{Q}$: $\mathrm{E}_{Y\sim\mathbb{Q}}\left[h_i(Y)h_j(Y)\right] = \delta_{ij}$

- E.g. If $Y \sim \mathbb{Q}$ has i.i.d. entries $Y_1, \ldots, Y_N \sim \mathrm{Unif}(\pm 1)$, orthonormal polynomials are monomials $1, Y_1, Y_2, Y_1Y_2, Y_1Y_3, Y_1Y_2Y_3, \ldots$ up to deg D
- Generally, can construct orthogonal polynomials when $\mathbb{Q}$ is a product measure (independent coordinates) $\mathbb{Q}$
    - First construct orthogonal polynomials for each coordinate…
- *Proof (Fact)*: Write $f(Y) = \sum_i \hat{f}_i\, h_i(Y)$ so $\mathrm{E}_{\mathbb{Q}}[f^2] = \sum_i \hat{f}_i^2$ …

Proof Ideas (Summary)

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, …]

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks...

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks...
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks...
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks...
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one
    - Not clear how to explicitly construct orthogonal polynomials for $\mathbb{Q}$

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, …]
- Other cases are more difficult, but we do have tools…
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks…
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one
    - Not clear how to explicitly construct orthogonal polynomials for $\mathbb{Q}$
  - Recovery: bound $\mathrm{MMSE}_{\leq D}$ [Schramm, W '20]

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, …]
- Other cases are more difficult, but we do have tools…
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks…
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one
    - Not clear how to explicitly construct orthogonal polynomials for $\mathbb{Q}$
  - Recovery: bound $\mathrm{MMSE}_{\leq D}$ [Schramm, W '20]
    - Sufficient: bound $\mathrm{Corr}_{\leq D} := \sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f \cdot x]}{\sqrt{\mathrm{E}_{\mathbb{P}}[f^2]}}$

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks...
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one
    - Not clear how to explicitly construct orthogonal polynomials for $\mathbb{Q}$
  - Recovery: bound $\mathrm{MMSE}_{\leq D}$ [Schramm, W '20]
    - Sufficient: bound $\mathrm{Corr}_{\leq D} \coloneqq \sup_{f \deg D} \frac{\mathrm{E}_{\mathbb{P}}[f \cdot x]}{\sqrt{\mathrm{E}_{\mathbb{P}}[f^2]}}$
    - Difficult for similar reason: would like orthogonal polynomials for $\mathbb{P}$...

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, …]
- Other cases are more difficult, but we do have tools…
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks…
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one
    - Not clear how to explicitly construct orthogonal polynomials for $\mathbb{Q}$
  - Recovery: bound $\mathrm{MMSE}_{\leq D}$ [Schramm, W '20]
    - Sufficient: bound $\mathrm{Corr}_{\leq D} \coloneqq \sup_{f \text{ deg } D} \frac{\mathrm{E}_{\mathbb{P}}[f \cdot x]}{\sqrt{\mathrm{E}_{\mathbb{P}}[f^2]}}$
    - Difficult for similar reason: would like orthogonal polynomials for $\mathbb{P}$…
  - Solution: use orthogonal polynomials in the underlying (non-observable) independent variables [Sohn, W '25]

Proof Ideas (Summary)

- The previous strategy led to an initial wave of success for low-degree hardness of testing "planted" vs (i.i.d.) "null" [HS '17, HKPRSS '17, ...]
- Other cases are more difficult, but we do have tools...
  - Sometimes $\mathrm{Adv}_{\leq D} = \omega(1)$ in the hard regime! But *separation* still tracks...
  - $\mathbb{Q}$ not a product measure [Rush, Skerman, W, Yang '22]
    - E.g. $\mathbb{P}$ has two planted cliques while $\mathbb{Q}$ has one
    - Not clear how to explicitly construct orthogonal polynomials for $\mathbb{Q}$
  - Recovery: bound $\mathrm{MMSE}_{\leq D}$ [Schramm, W '20]
    - Sufficient: bound $\mathrm{Corr}_{\leq D} \coloneqq \sup_{f \text{ deg } D} \frac{\mathrm{E}_{\mathbb{P}}[f \cdot x]}{\sqrt{\mathrm{E}_{\mathbb{P}}[f^2]}}$
    - Difficult for similar reason: would like orthogonal polynomials for $\mathbb{P}$...
  - Solution: use orthogonal polynomials in the underlying (non-observable) independent variables [Sohn, W '25]
    - For planted clique: $Z_{ij} \sim \mathrm{Ber}(1/2)$ for each $i < j$ and $x_i \sim \mathrm{Ber}(k/n)$ for each vertex

Sharp Thresholds in Estimation

Sharp Thresholds in Estimation

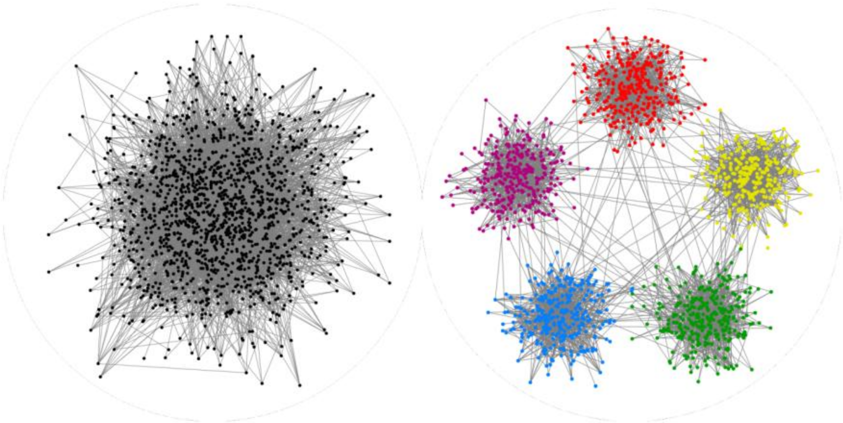


Image credit: Abbe '17

- Stochastic block model (community detection)

Sharp Thresholds in Estimation

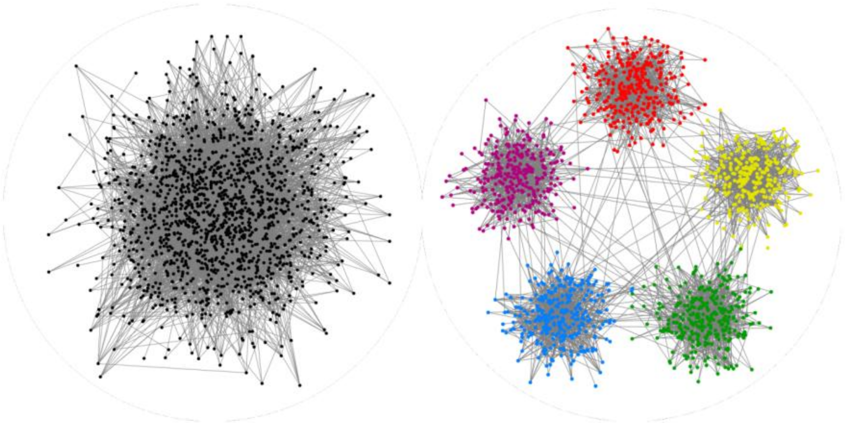


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$

Sharp Thresholds in Estimation

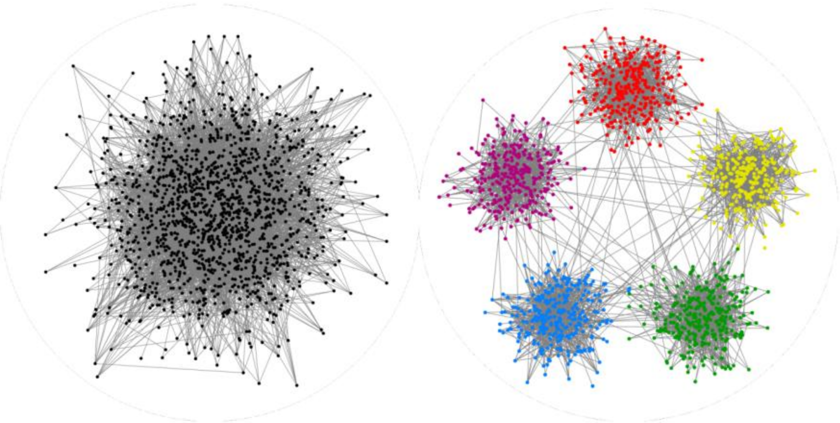


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery

Sharp Thresholds in Estimation

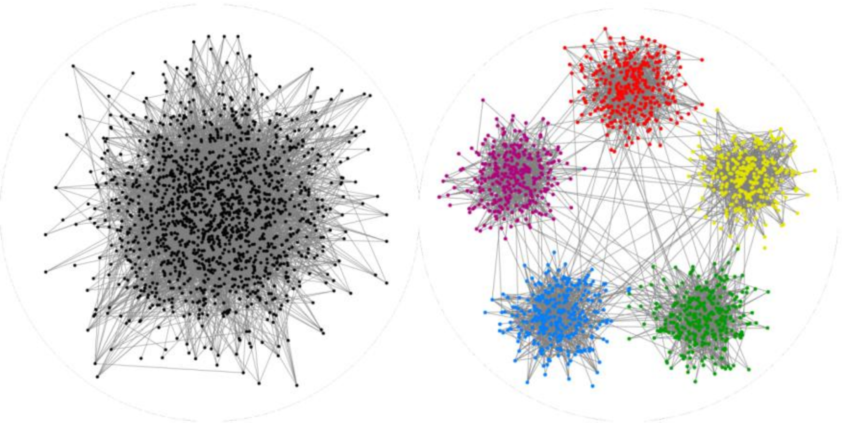


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]

Sharp Thresholds in Estimation

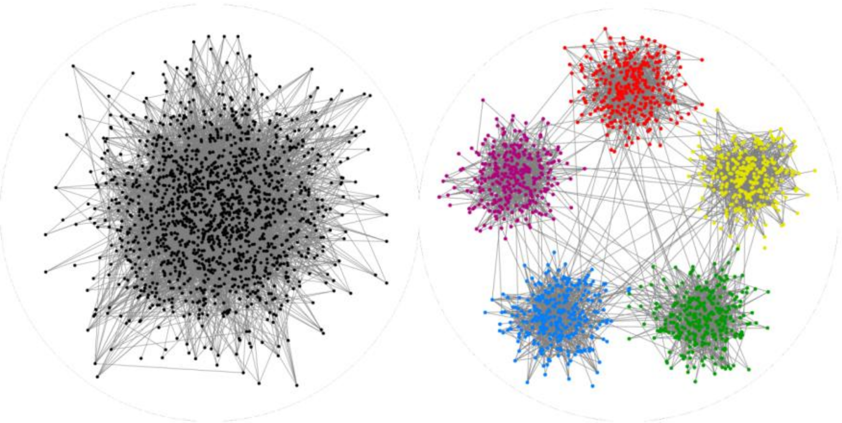


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]
- Low-degree recovery matches KS [Sohn, W '25; Ding, Hua, Slot, Steurer '25]

Sharp Thresholds in Estimation

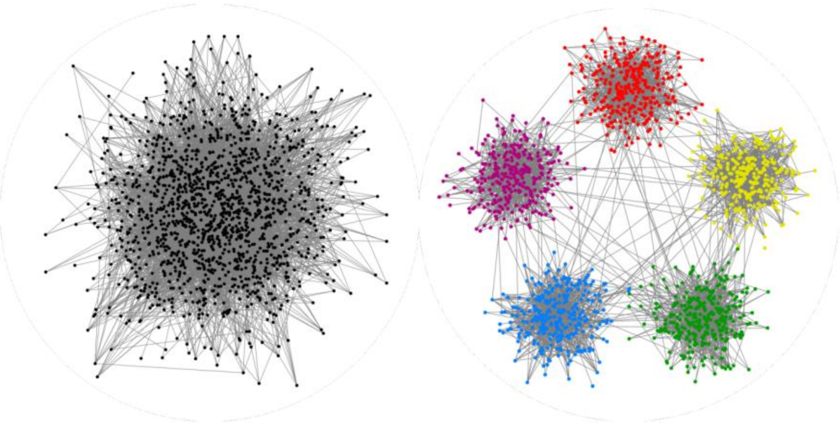


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]
- Low-degree recovery matches KS [Sohn, W '25; Ding, Hua, Slot, Steurer '25]
- Now take q growing with n...

Sharp Thresholds in Estimation

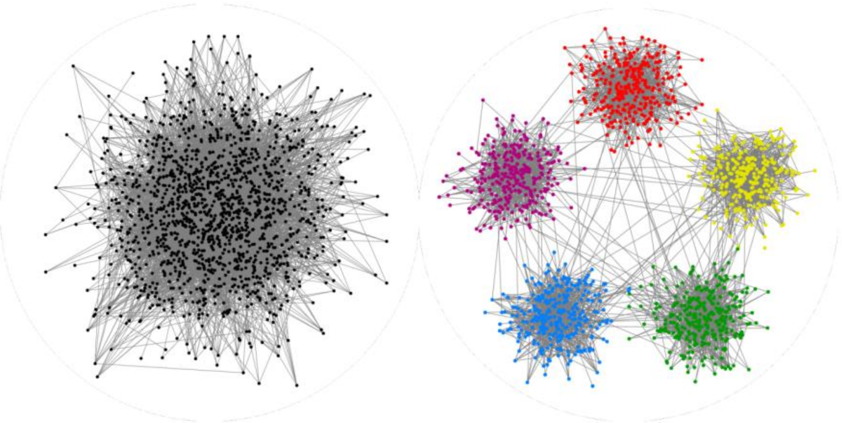


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]
- Low-degree recovery matches KS [Sohn, W '25; Ding, Hua, Slot, Steurer '25]
- Now take q growing with n...
  - Low-degree recovery matches KS for $q \ll \sqrt{n}$... [Chin, Mossel, Sohn, W '25]

Sharp Thresholds in Estimation

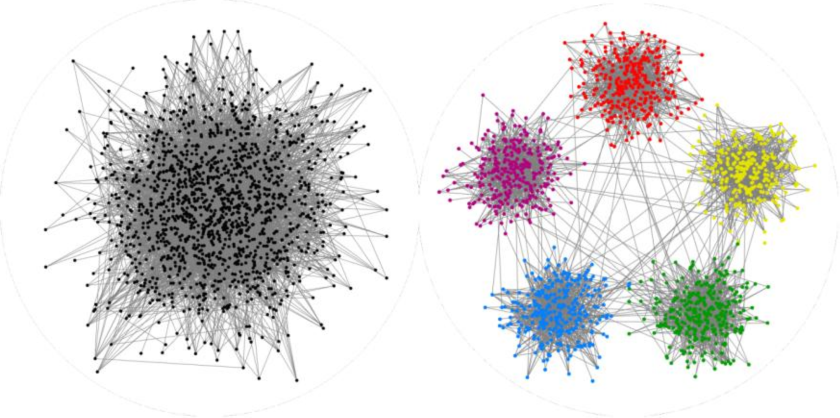


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]
- Low-degree recovery matches KS [Sohn, W '25; Ding, Hua, Slot, Steurer '25]
- Now take q growing with n...
  - Low-degree recovery matches KS for $q \ll \sqrt{n}$... [Chin, Mossel, Sohn, W '25]
  - ... but you can beat KS when $q \gg \sqrt{n}$

Sharp Thresholds in Estimation

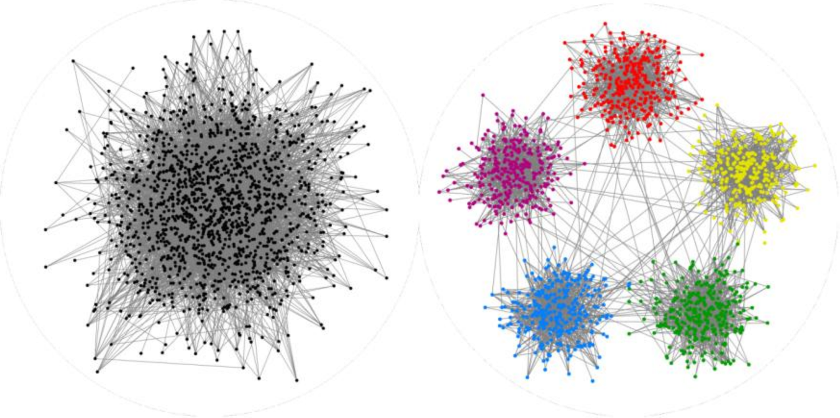


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]
- Low-degree recovery matches KS [Sohn, W '25; Ding, Hua, Slot, Steurer '25]
- Now take q growing with n…
  - Low-degree recovery matches KS for $q \ll \sqrt{n}$… [Chin, Mossel, Sohn, W '25]
  - … but you can beat KS when $q \gg \sqrt{n}$
  - Detection-recovery gap

Sharp Thresholds in Estimation

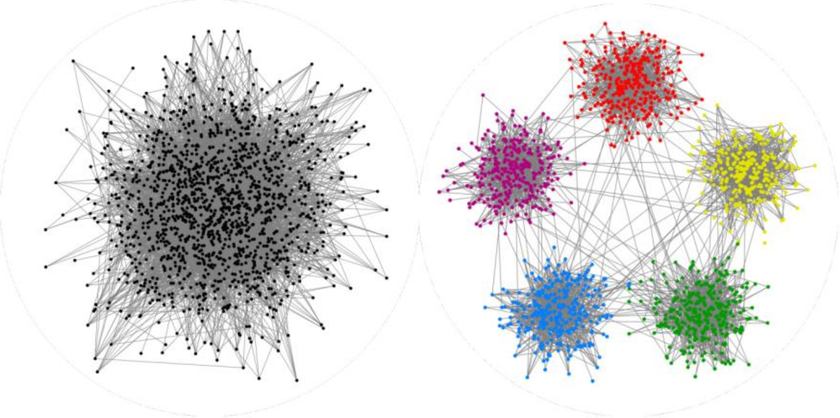


Image credit: Abbe '17

- Stochastic block model (community detection)
  - Average degree d; SNR λ; q communities -- all $\Theta(1)$
- Sharp "Kesten-Stigum" threshold: $d\lambda^2 = 1$
  - Conjectured computational threshold for strong detection & weak recovery
- Low-degree detection matches KS [Hopkins, Steurer '17]
- Low-degree recovery matches KS [Sohn, W '25; Ding, Hua, Slot, Steurer '25]
- Now take q growing with n...
  - Low-degree recovery matches KS for $q \ll \sqrt{n}$... [Chin, Mossel, Sohn, W '25]
  - ... but you can beat KS when $q \gg \sqrt{n}$
  - Detection-recovery gap
  - No other frameworks apply here (?)

Connection to Other Heuristics

Connection to Other Heuristics

- AMP (approximate message passing)
- OGP (overlap gap property)
- SOS (sum-of-squares hierarchy)
- SQ (statistical query model)
- LD (low-degree polynomials)

Connection to Other Heuristics

- AMP (approximate message passing)
- OGP (overlap gap property)
- SOS (sum-of-squares hierarchy)
- SQ (statistical query model)
- LD (low-degree polynomials)

- “Unify” these: Can we prove they all make the same predictions?

Connection to Other Heuristics

- AMP (approximate message passing)
- OGP (overlap gap property)
- SOS (sum-of-squares hierarchy)
- SQ (statistical query model)
- LD (low-degree polynomials)

- "Unify" these: Can we prove they all make the same predictions?

- Two issues:

Connection to Other Heuristics

- AMP (approximate message passing)
- OGP (overlap gap property)
- SOS (sum-of-squares hierarchy)
- SQ (statistical query model)
- LD (low-degree polynomials)

- “Unify” these: Can we prove they all make the same predictions?

- Two issues:
  1. Sometimes they are NOT equivalent…

Connection to Other Heuristics

- AMP (approximate message passing)
- OGP (overlap gap property)
- SOS (sum-of-squares hierarchy)
- SQ (statistical query model)
- LD (low-degree polynomials)

- "Unify" these: Can we prove they all make the same predictions?

- Two issues:
    1. Sometimes they are NOT equivalent...
    2. Often they are not even answering the same question...

Tensor PCA

Tensor PCA

- “Rank-1 tensor plus noise” -- $T \in \mathbb{R}^{n \times n \times n}$

Tensor PCA

- "Rank-1 tensor plus noise" -- $T \in \mathbb{R}^{n \times n \times n}$
- AMP gets the "wrong" (sub-optimal) threshold [Montanari, Richard '14]

Tensor PCA

- "Rank-1 tensor plus noise" -- $T \in \mathbb{R}^{n \times n \times n}$
- AMP gets the "wrong" (sub-optimal) threshold [Montanari, Richard '14]
- OGP and "local search methods" also predict the same wrong threshold [Ben Arous, Gheissari, Jagannath '18; Chen, Sheehan, Zadik '24]

Tensor PCA

- “Rank-1 tensor plus noise” -- $T \in \mathbb{R}^{n \times n \times n}$
- AMP gets the “wrong” (sub-optimal) threshold [Montanari, Richard ‘14]
- OGP and “local search methods” also predict the same wrong threshold [Ben Arous, Gheissari, Jagannath ’18; Chen, Sheehan, Zadik ‘24]
- SQ also gets a different wrong threshold [Dudeja, Hsu ‘20]

Tensor PCA

- "Rank-1 tensor plus noise" -- $T \in \mathbb{R}^{n \times n \times n}$
- AMP gets the "wrong" (sub-optimal) threshold [Montanari, Richard '14]
- OGP and "local search methods" also predict the same wrong threshold [Ben Arous, Gheissari, Jagannath '18; Chen, Sheehan, Zadik '24]
- SQ also gets a different wrong threshold [Dudeja, Hsu '20]
- SOS, LD get the "correct" threshold [Hopkins, Shi, Steurer '15; HKPRSS '17]

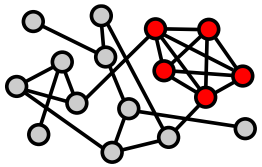
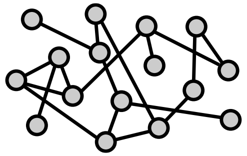
Tensor PCA

- "Rank-1 tensor plus noise" -- $T \in \mathbb{R}^{n \times n \times n}$
- AMP gets the "wrong" (sub-optimal) threshold [Montanari, Richard '14]
- OGP and "local search methods" also predict the same wrong threshold [Ben Arous, Gheissari, Jagannath '18; Chen, Sheehan, Zadik '24]
- SQ also gets a different wrong threshold [Dudeja, Hsu '20]
- SOS, LD get the "correct" threshold [Hopkins, Shi, Steurer '15; HKPRSS '17]

- "Redemption"
  - Kikuchi hierarchy [W, Alaoui, Moore '19]
  - Averaged gradient descent [Biroli, Cammarota, Ricci-Tersenghi '19]
  - Modified MCMC [Lovig, Sheehan, Tsirkas, Zadik '25]
  - ... but somewhat problem-specific (?)

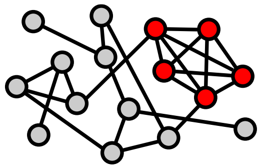
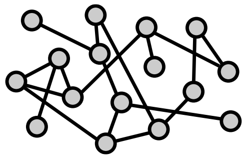
Tasks

Tasks

$\mathbb{P}$  $\mathbb{Q}$ 

- Using planted clique as a running example…

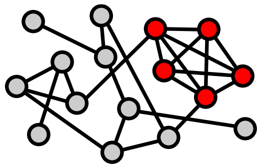
Tasks

$\mathbb{P}$  $\mathbb{Q}$ 

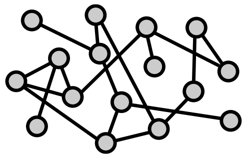
- Using planted clique as a running example…
- **Detection (a.k.a. Testing)**: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$

Tasks

$\mathbb{P}$

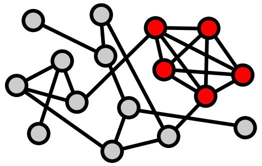
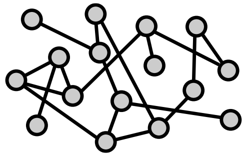


$\mathbb{Q}$



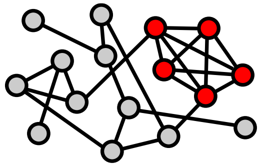
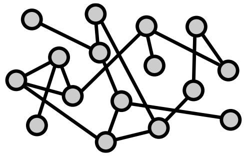
- Using planted clique as a running example…
- **Detection (a.k.a. Testing)**: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$
- **Recovery (a.k.a. Estimation)**: Given $G \sim \mathbb{P}$, find the planted clique

Tasks

$\mathbb{P}$  $\mathbb{Q}$ 

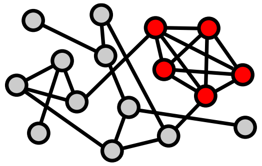
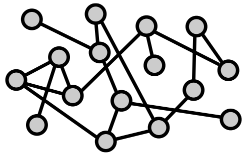
- Using planted clique as a running example…
- **Detection (a.k.a. Testing)**: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$
- **Recovery (a.k.a. Estimation)**: Given $G \sim \mathbb{P}$, find the planted clique
- **(Non-planted) Optimization**: Given $G \sim \mathbb{Q}$, find any k-clique

Tasks

$\mathbb{P}$  $\mathbb{Q}$ 

- Using planted clique as a running example…
- **Detection (a.k.a. Testing)**: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$
- **Recovery (a.k.a. Estimation)**: Given $G \sim \mathbb{P}$, find the planted clique
- **(Non-planted) Optimization**: Given $G \sim \mathbb{Q}$, find any k-clique
- **Refutation (or Certification)**: Given $G \sim \mathbb{Q}$, prove there's no k-clique

Tasks

$\mathbb{P}$  $\mathbb{Q}$ 

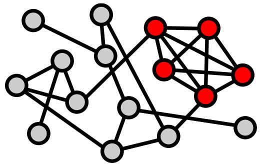
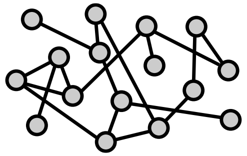
- Using planted clique as a running example…
- **Detection (a.k.a. Testing)**: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$
- **Recovery (a.k.a. Estimation)**: Given $G \sim \mathbb{P}$, find <u>the</u> planted clique
- **(Non-planted) Optimization**: Given $G \sim \mathbb{Q}$, find <u>any</u> k-clique
- **Refutation (or Certification)**: Given $G \sim \mathbb{Q}$, prove there's no k-clique

Detection ($\mathbb{P}$ vs $\mathbb{Q}$)

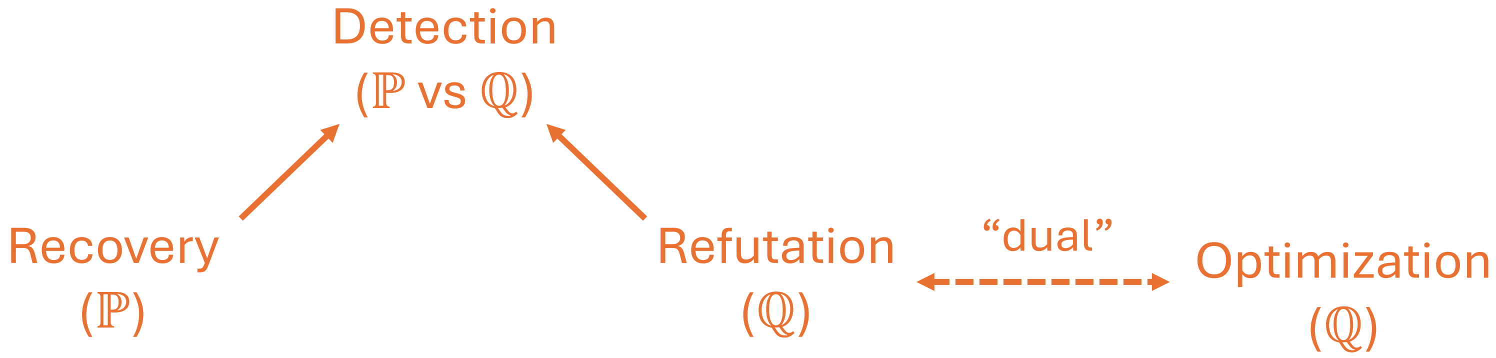
Recovery ($\mathbb{P}$)

Refutation ($\mathbb{Q}$) ← "dual" → Optimization ($\mathbb{Q}$)

Tasks

$\mathbb{P}$  $\mathbb{Q}$ 

- Using planted clique as a running example...
- **Detection (a.k.a. Testing)**: Decide if a given graph came from $\mathbb{P}$ or $\mathbb{Q}$
- **Recovery (a.k.a. Estimation)**: Given $G \sim \mathbb{P}$, find <u>the</u> planted clique
- **(Non-planted) Optimization**: Given $G \sim \mathbb{Q}$, find <u>any</u> k-clique
- **Refutation (or Certification)**: Given $G \sim \mathbb{Q}$, prove there's no k-clique



- These tasks can all have different thresholds in general

Frameworks vs Tasks

Which frameworks can give **hardness results** for which tasks?

	AMP	OGP	SOS	SQ	LD
Detection			✓	✓	✓
Recovery	✓	✓		✓	✓
Optimization	✓	✓			✓
Refutation			✓		✓

Known Connections

Known Connections

Despite **many** caveats, some known connections among frameworks

Known Connections

Despite **many** caveats, some known connections among frameworks

“Statistical Physics” / “Geometric”

Belief Propagation (BP)

Approximate Message Passing (AMP)

Overlap Gap Property (OGP)

Free Energy Barriers (Franz-Parisi Potential)

Known Connections

Despite **many** caveats, some known connections among frameworks

"Statistical Physics" / "Geometric"

Belief Propagation (BP)

Approximate Message Passing (AMP)

Overlap Gap Property (OGP)

Free Energy Barriers (Franz-Parisi Potential)

"Computer Science" / "Algebraic"

Sum-of-Squares (SOS)

Spectral Methods

Low-Degree Polynomials

Statistical Query (SQ)

Known Connections

Despite **many** caveats, some known connections among frameworks

"Statistical Physics" / "Geometric"

Belief Propagation (BP)

Approximate Message Passing (AMP)

[GJ'19]

Overlap Gap Property (OGP)

Free Energy Barriers (Franz-Parisi Potential)

Gamarnik, Jagannath '19

"Computer Science" / "Algebraic"

Sum-of-Squares (SOS)

Spectral Methods

Low-Degree Polynomials

Statistical Query (SQ)

Known Connections

Despite **many** caveats, some known connections among frameworks

"Statistical Physics" / "Geometric"

Belief Propagation (BP)

↓

Approximate Message Passing (AMP)

↑ [GJ'19]

Overlap Gap Property (OGP)

Free Energy Barriers (Franz-Parisi Potential)

"Computer Science" / "Algebraic"

Sum-of-Squares (SOS)

↓ [HKPRSS'17]

Spectral Methods

↓ Power Iter

Low-Degree Polynomials

Statistical Query (SQ)

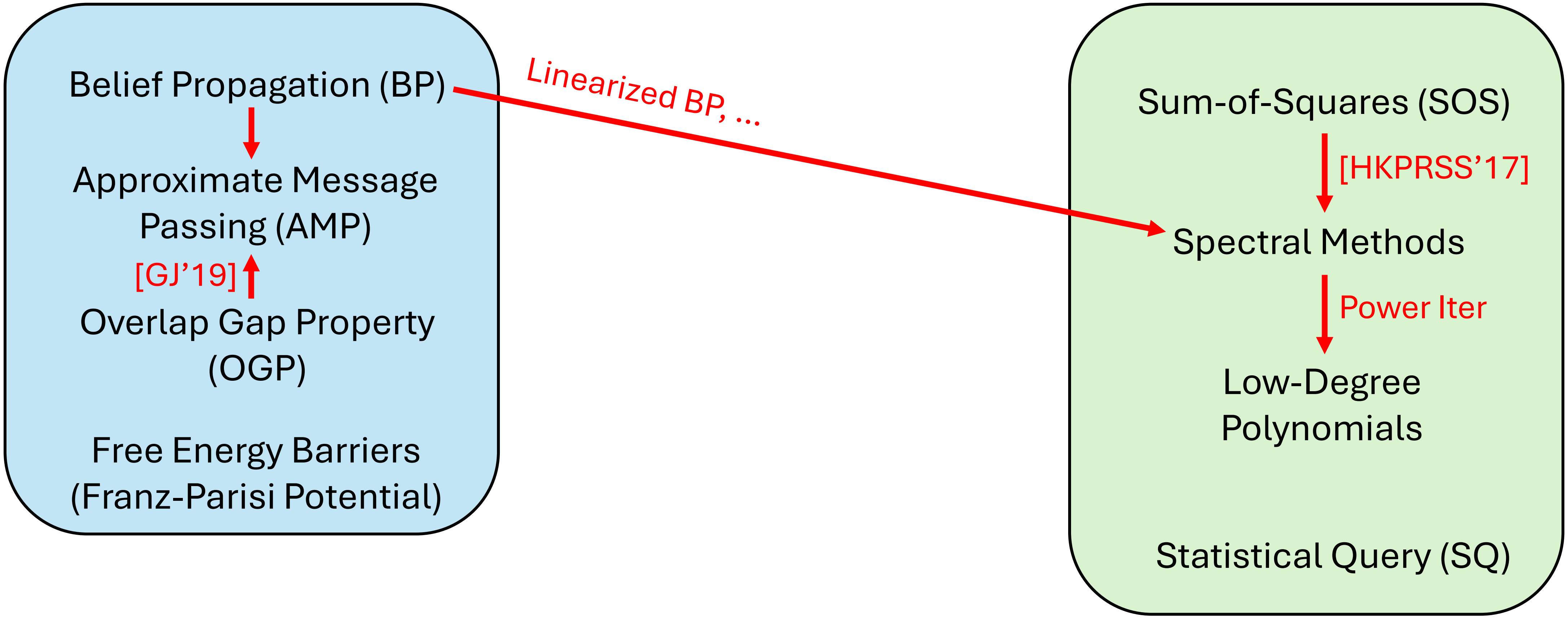
Hopkins, Kothari, Potechin, Raghavendra, Schramm, Steurer '17

Known Connections

Despite **many** caveats, some known connections among frameworks

“Statistical Physics” / “Geometric”

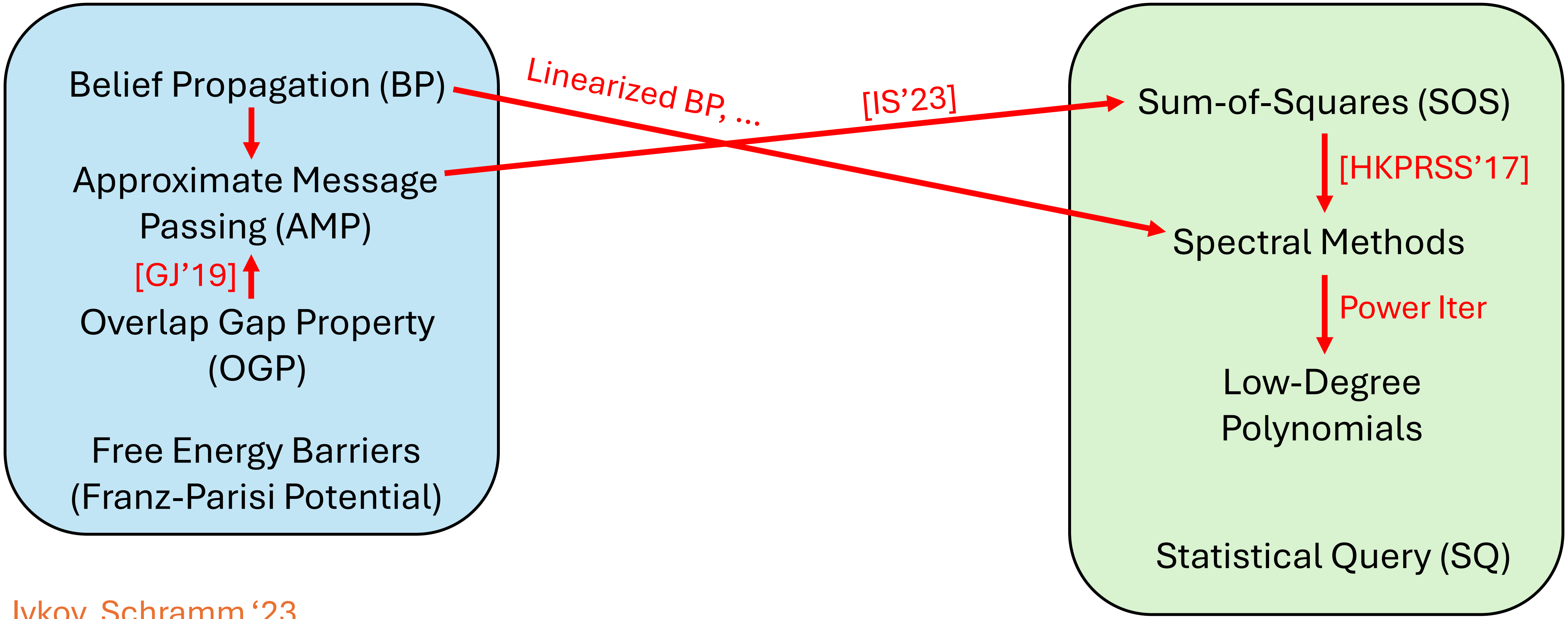
“Computer Science” / “Algebraic”



Known Connections

Despite **many** caveats, some known connections among frameworks

“Statistical Physics” / “Geometric” **“Computer Science” / “Algebraic”**

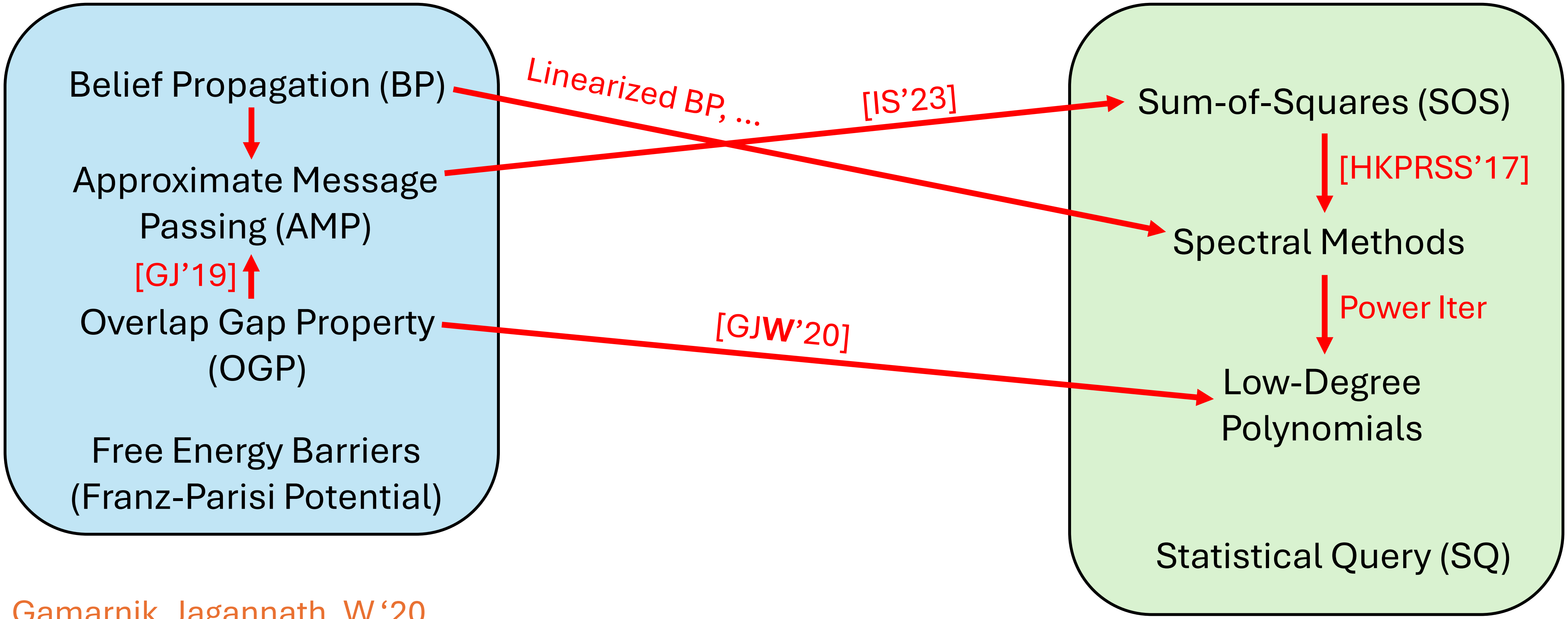


Ivkov, Schramm ‘23

Known Connections

Despite **many** caveats, some known connections among frameworks

"Statistical Physics" / "Geometric" **"Computer Science" / "Algebraic"**

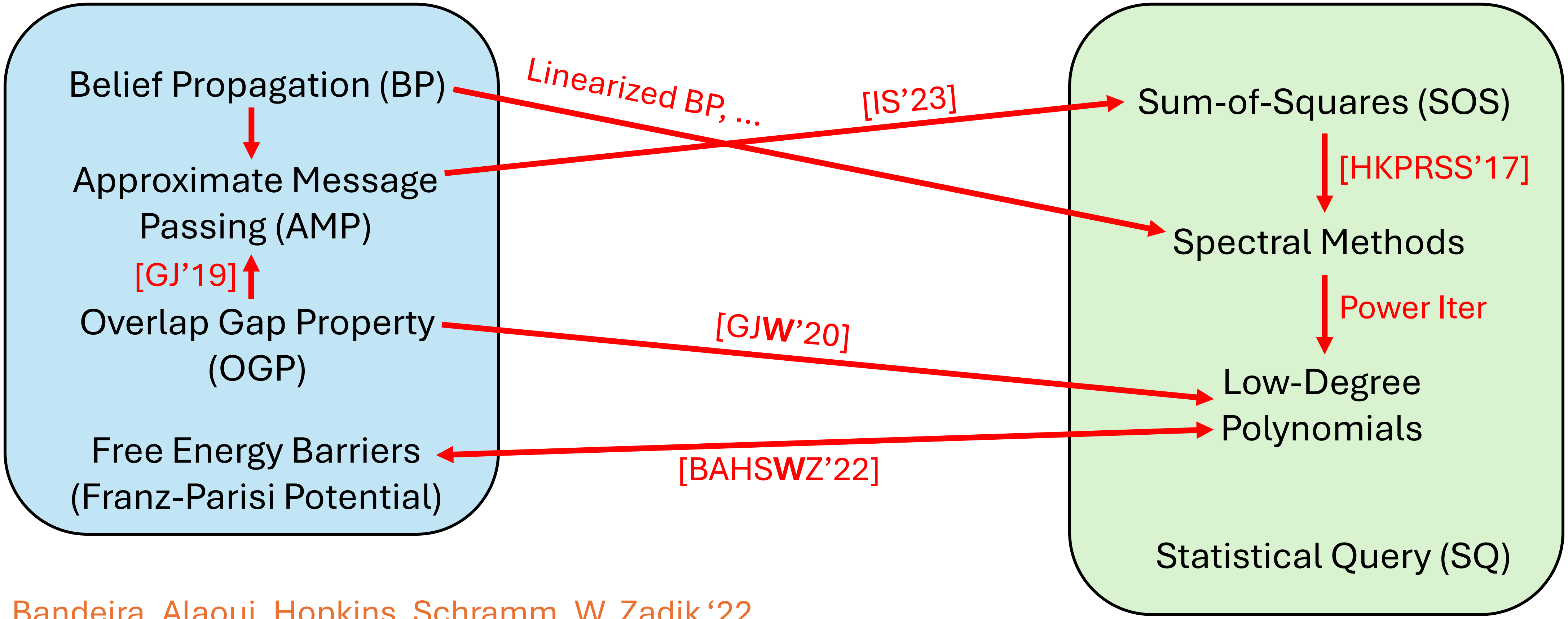


Gamarnik, Jagannath, W '20

Known Connections

Despite **many** caveats, some known connections among frameworks

“Statistical Physics” / “Geometric” **“Computer Science” / “Algebraic”**

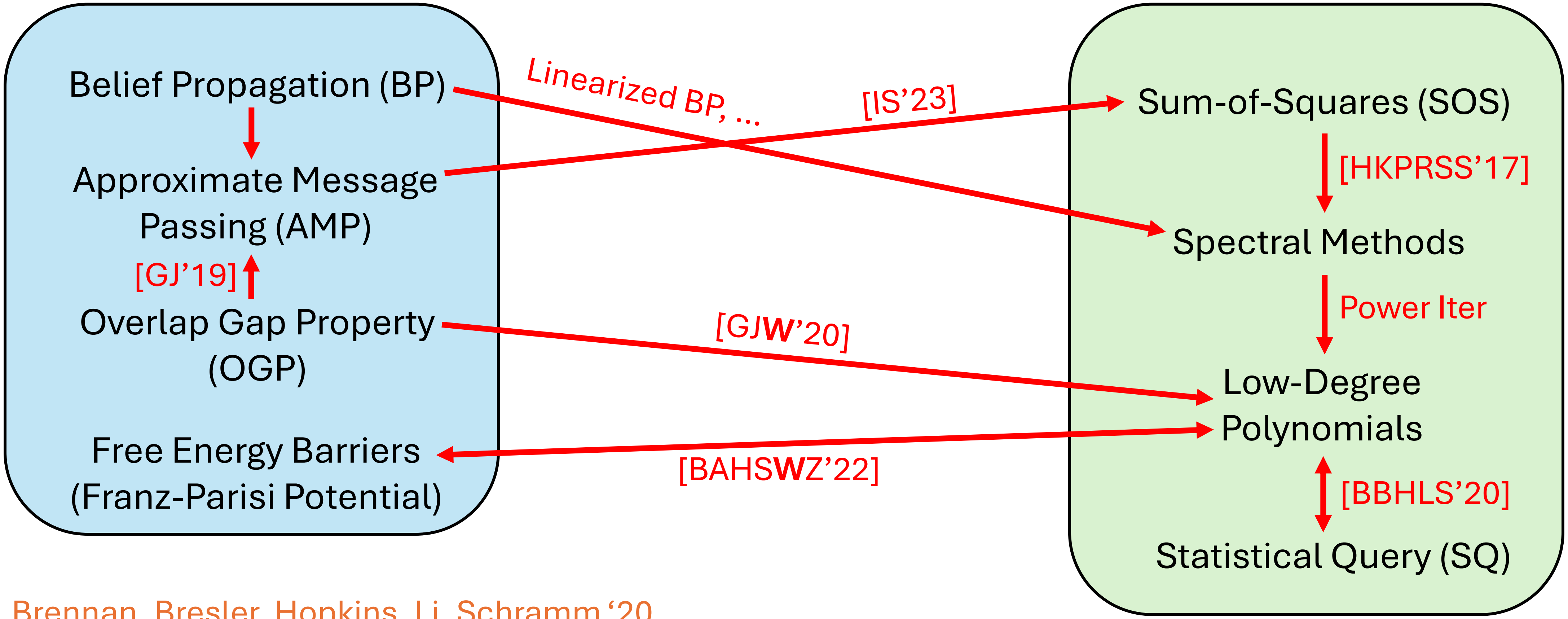


Bandeira, Alaoui, Hopkins, Schramm, W, Zadik ‘22

Known Connections

Despite **many** caveats, some known connections among frameworks

"Statistical Physics" / "Geometric" **"Computer Science" / "Algebraic"**

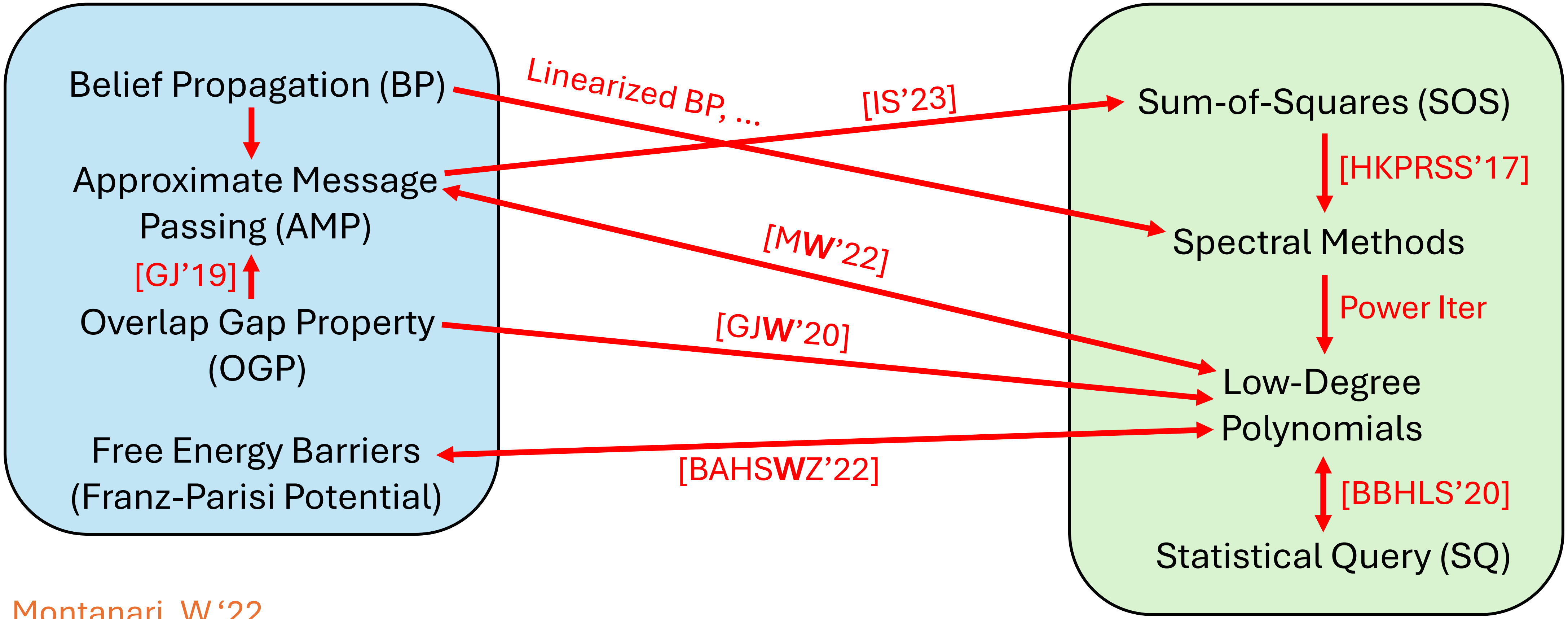


Brennan, Bresler, Hopkins, Li, Schramm '20

Known Connections

Despite **many** caveats, some known connections among frameworks

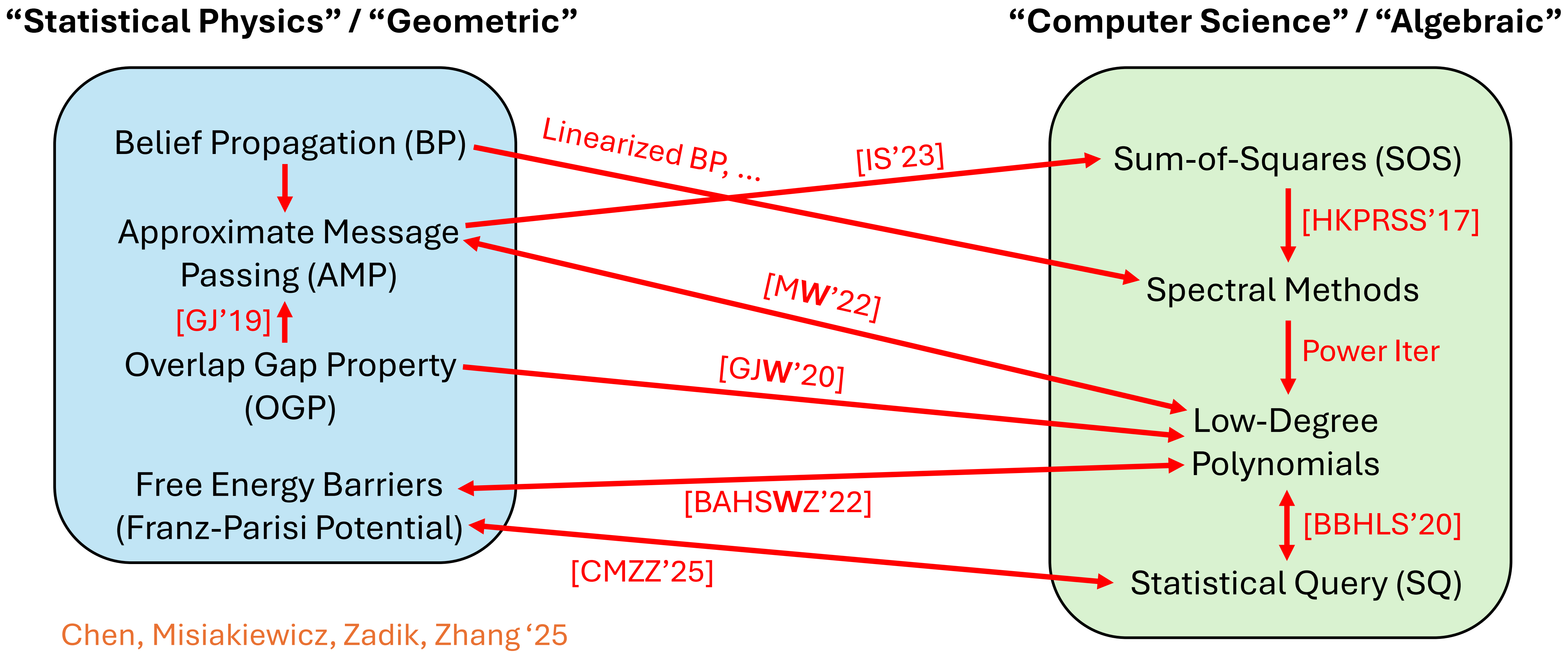
“Statistical Physics” / “Geometric” **“Computer Science” / “Algebraic”**



Montanari, W ‘22

Known Connections

Despite **many** caveats, some known connections among frameworks



Chen, Misiakiewicz, Zadik, Zhang ‘25

Known Connections

Despite **many** caveats, some known connections among frameworks