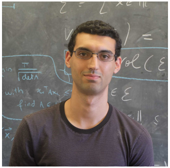
The Kikuchi Hierarchy and Tensor PCA

Alex Wein
Courant Institute, New York University

Joint work with:



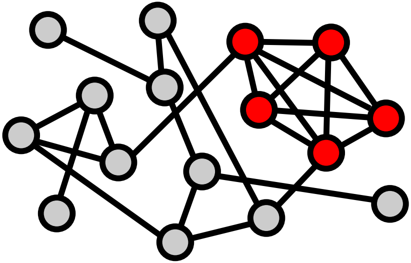
Ahmed El Alaoui
Cornell

Cris Moore
Santa Fe Institute

High-Dimensional Statistics

Example: planted clique problem

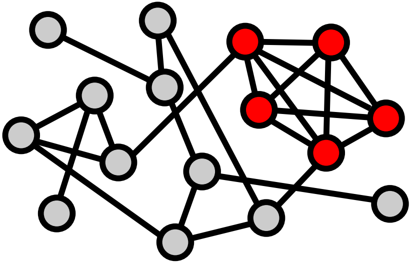
- Random graph $G(n, 1/2)$ with a planted k-clique
- Goal: find the clique



High-Dimensional Statistics

Example: planted clique problem

- Random graph $G(n, 1/2)$ with a planted k-clique
- Goal: find the clique

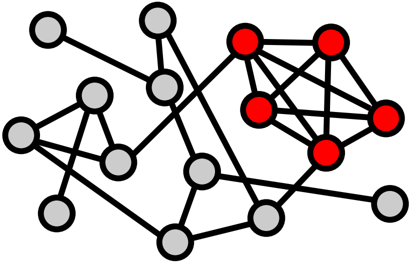


Believed to have a statistical-computational gap

High-Dimensional Statistics

Example: planted clique problem

- Random graph $G(n, 1/2)$ with a planted k-clique
- Goal: find the clique



Believed to have a statistical-computational gap

What makes statistical problems easy vs hard?

Statistical Physics of Inference

- **High-dimensional inference problems:** planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...

Statistical Physics of Inference

- High-dimensional inference problems: planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...
- Connection to statistical physics: posterior distribution is a Gibbs/Boltzmann distribution

Statistical Physics of Inference

- High-dimensional inference problems: planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...
- Connection to statistical physics: posterior distribution is a Gibbs/Boltzmann distribution
- Algorithms: belief propagation (BP) [Pearl '82, Mézard-Parisi-Virasoro '86], approximate message passing (AMP) [Donoho-Maleki-Montanari '09]

Statistical Physics of Inference

- High-dimensional inference problems: planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...
- Connection to statistical physics: posterior distribution is a Gibbs/Boltzmann distribution
- Algorithms: belief propagation (BP) [Pearl '82, Mézard-Parisi-Virasoro '86], approximate message passing (AMP) [Donoho-Maleki-Montanari '09]
  - Sharp results: exact MMSE, phase transitions

Statistical Physics of Inference

- High-dimensional inference problems: planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...
- Connection to statistical physics: posterior distribution is a Gibbs/Boltzmann distribution
- Algorithms: belief propagation (BP) [Pearl '82, Mézard-Parisi-Virasoro '86], approximate message passing (AMP) [Donoho-Maleki-Montanari '09]
  - Sharp results: exact MMSE, phase transitions
  - Known to be statistically optimal in many settings

Statistical Physics of Inference

- High-dimensional inference problems: planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...
- Connection to statistical physics: posterior distribution is a Gibbs/Boltzmann distribution
- Algorithms: belief propagation (BP) [Pearl '82, Mézard-Parisi-Virasoro '86], approximate message passing (AMP) [Donoho-Maleki-Montanari '09]
  - Sharp results: exact MMSE, phase transitions
  - Known to be statistically optimal in many settings
- Evidence for computational hardness: failure of BP/AMP, free energy barriers [Decelle-Krzakala-Moore-Zdeborová '11, Lesieur-Krzakala-Zdeborová '15]

Statistical Physics of Inference

- High-dimensional inference problems: planted clique, community detection, compressed sensing, phase retrieval, sparse PCA, spiked Wigner/Wishart matrix, constraint satisfaction, ...
- Connection to statistical physics: posterior distribution is a Gibbs/Boltzmann distribution
- Algorithms: belief propagation (BP) [Pearl '82, Mézard-Parisi-Virasoro '86], approximate message passing (AMP) [Donoho-Maleki-Montanari '09]
  - Sharp results: exact MMSE, phase transitions
  - Known to be statistically optimal in many settings
- Evidence for computational hardness: failure of BP/AMP, free energy barriers [Decelle-Krzakala-Moore-Zdeborová '11, Lesieur-Krzakala-Zdeborová '15]

This theory has been hugely successful at precisely understanding statistical and computational limits of many problems.

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems
- Degree-d relaxation can be solved in $n^{O(d)}$-time

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems
- Degree-d relaxation can be solved in $n^{O(d)}$-time
- Higher degree gives more powerful algorithms

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems
- Degree-d relaxation can be solved in $n^{O(d)}$-time
- Higher degree gives more powerful algorithms
- State-of-the-art algorithms for many statistical problems: tensor decomposition, tensor completion, planted sparse vector, dictionary learning, refuting random CSPs, mixtures of Gaussians, ...

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems
- Degree-d relaxation can be solved in $n^{O(d)}$-time
- Higher degree gives more powerful algorithms
- State-of-the-art algorithms for many statistical problems: tensor decomposition, tensor completion, planted sparse vector, dictionary learning, refuting random CSPs, mixtures of Gaussians, ...
- Evidence for computational hardness: SoS lower bounds

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems
- Degree-d relaxation can be solved in $n^{O(d)}$-time
- Higher degree gives more powerful algorithms
- State-of-the-art algorithms for many statistical problems:
  tensor decomposition, tensor completion, planted sparse vector, dictionary learning, refuting random CSPs, mixtures of Gaussians, ...
- Evidence for computational hardness: SoS lower bounds

Meta-question: unify the statistical physics and SoS approaches?

Sum-of-Squares (SoS) Hierarchy

A competing theory: sum-of-squares hierarchy [Parrilo '00, Lasserre '01]

- Systematic way to obtain convex relaxations of polynomial optimization problems
- Degree-d relaxation can be solved in $n^{O(d)}$-time
- Higher degree gives more powerful algorithms
- State-of-the-art algorithms for many statistical problems: tensor decomposition, tensor completion, planted sparse vector, dictionary learning, refuting random CSPs, mixtures of Gaussians, ...
- Evidence for computational hardness: SoS lower bounds

Meta-question: unify the statistical physics and SoS approaches?

This talk: case study on tensor PCA – a problem where statistical physics and SoS disagree (!!!)

Tensor PCA (Principal Component Analysis)

Definition (Spiked Tensor Model [Richard-Montanari '14])

$x \in \{\pm 1\}^n$ – signal
$p \in \{2, 3, 4, \dots\}$ – tensor order
For each subset $U \subseteq [n]$ of size $|U| = p$, observe

$$Y_U = \lambda \prod_{i \in U} x_i + \mathcal{N}(0, 1)$$

$\lambda \geq 0$ – signal-to-noise parameter
Goal: given $\{Y_U\}$, recover x

- "For every p variables, get a noisy observation of their parity"
- In tensor notation: $Y = \lambda x^{\otimes p} + Z$ where Z is symmetric noise
- Case $p = 2$ is the spiked Wigner matrix model $Y = \lambda x x^\top + Z$

Algorithms for Tensor PCA

Maximum likelihood estimation (MLE):

$$\Pr[x|Y] \propto \exp\left(\sum_{|U|=p} \lambda Y_U \prod_{i\in U} x_i\right) = \exp\left(\frac{\lambda}{p}\langle Y, x^{\otimes p}\rangle\right)$$

$$\text{MLE: } \hat{x} = \underset{v\in\{\pm 1\}^n}{\operatorname{argmax}} \langle Y, v^{\otimes p}\rangle$$

Algorithms for Tensor PCA

Maximum likelihood estimation (MLE):

$$\Pr[x|Y] \propto \exp\left(\sum_{|U|=p} \lambda Y_U \prod_{i \in U} x_i\right) = \exp\left(\frac{\lambda}{p}\langle Y, x^{\otimes p}\rangle\right)$$

$$\text{MLE: } \hat{x} = \underset{v \in \{\pm 1\}^n}{\operatorname{argmax}} \langle Y, v^{\otimes p}\rangle$$

- Succeeds when $\lambda \gtrsim n^{(1-p)/2}$ [Richard-Montanari '14]

Algorithms for Tensor PCA

Maximum likelihood estimation (MLE):

$$\Pr[x|Y] \propto \exp\left(\sum_{|U|=p} \lambda Y_U \prod_{i \in U} x_i\right) = \exp\left(\frac{\lambda}{p}\langle Y, x^{\otimes p}\rangle\right)$$

$$\text{MLE: } \hat{x} = \underset{v \in \{\pm 1\}^n}{\operatorname{argmax}} \langle Y, v^{\otimes p}\rangle$$

- Succeeds when $\lambda \gtrsim n^{(1-p)/2}$ [Richard-Montanari '14]
- Statistically optimal (up to constant factors in λ)

Algorithms for Tensor PCA

Maximum likelihood estimation (MLE):

$$\Pr[x|Y] \propto \exp\left(\sum_{|U|=p} \lambda Y_U \prod_{i \in U} x_i\right) = \exp\left(\frac{\lambda}{p}\langle Y, x^{\otimes p}\rangle\right)$$

$$\text{MLE: } \hat{x} = \operatorname*{argmax}_{v \in \{\pm 1\}^n} \langle Y, v^{\otimes p}\rangle$$

- Succeeds when $\lambda \gtrsim n^{(1-p)/2}$ [Richard-Montanari '14]
- Statistically optimal (up to constant factors in λ)
- Problem: requires exponential time 2^n

Algorithms for Tensor PCA

Local algorithms: keep track of a "guess" $v \in \mathbb{R}^n$ and locally maximize the log-likelihood $\mathcal{L}(v) = \langle Y, v^{\otimes p} \rangle$

Algorithms for Tensor PCA

Local algorithms: keep track of a "guess" $v \in \mathbb{R}^n$ and locally maximize the log-likelihood $\mathcal{L}(v) = \langle Y, v^{\otimes p} \rangle$

- Gradient descent [Ben Arous-Gheissari-Jagannath '18]

Algorithms for Tensor PCA

Local algorithms: keep track of a "guess" $v \in \mathbb{R}^n$ and locally maximize the log-likelihood $\mathcal{L}(v) = \langle Y, v^{\otimes p} \rangle$

- Gradient descent [Ben Arous-Gheissari-Jagannath '18]
- Tensor power iteration [Richard-Montanari '14]

Algorithms for Tensor PCA

Local algorithms: keep track of a "guess" $v \in \mathbb{R}^n$ and locally maximize the log-likelihood $\mathcal{L}(v) = \langle Y, v^{\otimes p} \rangle$

- Gradient descent [Ben Arous-Gheissari-Jagannath '18]
- Tensor power iteration [Richard-Montanari '14]
- Langevin dynamics [Ben Arous-Gheissari-Jagannath '18]

Algorithms for Tensor PCA

Local algorithms: keep track of a "guess" $v \in \mathbb{R}^n$ and locally maximize the log-likelihood $\mathcal{L}(v) = \langle Y, v^{\otimes p} \rangle$

- Gradient descent [Ben Arous-Gheissari-Jagannath '18]
- Tensor power iteration [Richard-Montanari '14]
- Langevin dynamics [Ben Arous-Gheissari-Jagannath '18]
- Approximate message passing (AMP) [Richard-Montanari '14]

Algorithms for Tensor PCA

Local algorithms: keep track of a "guess" $v \in \mathbb{R}^n$ and locally maximize the log-likelihood $\mathcal{L}(v) = \langle Y, v^{\otimes p} \rangle$

- Gradient descent [Ben Arous-Gheissari-Jagannath '18]
- Tensor power iteration [Richard-Montanari '14]
- Langevin dynamics [Ben Arous-Gheissari-Jagannath '18]
- Approximate message passing (AMP) [Richard-Montanari '14]

These only succeed when $\lambda \gg n^{-1/2}$ (worse than MLE)

Algorithms for Tensor PCA

Sum-of-squares (SoS) and spectral methods:

Algorithms for Tensor PCA

Sum-of-squares (SoS) and spectral methods:

- SoS semidefinite program [Hopkins-Shi-Steurer '15]

Algorithms for Tensor PCA

Sum-of-squares (SoS) and spectral methods:

- SoS semidefinite program [Hopkins-Shi-Steurer '15]
- Spectral SoS [Hopkins-Shi-Steurer '15, Hopkins-Schramm-Shi-Steurer '15]

Algorithms for Tensor PCA

Sum-of-squares (SoS) and spectral methods:

- SoS semidefinite program [Hopkins-Shi-Steurer '15]
- Spectral SoS [Hopkins-Shi-Steurer '15, Hopkins-Schramm-Shi-Steurer '15]
- Tensor unfolding [Richard-Montanari '14, Hopkins-Shi-Steurer '15]

Algorithms for Tensor PCA

Sum-of-squares (SoS) and spectral methods:

- SoS semidefinite program [Hopkins-Shi-Steurer '15]
- Spectral SoS [Hopkins-Shi-Steurer '15, Hopkins-Schramm-Shi-Steurer '15]
- Tensor unfolding [Richard-Montanari '14, Hopkins-Shi-Steurer '15]

These are poly-time and succeed when $\lambda \gg n^{-p/4}$

Algorithms for Tensor PCA

Sum-of-squares (SoS) and spectral methods:

- SoS semidefinite program [Hopkins-Shi-Steurer '15]
- Spectral SoS [Hopkins-Shi-Steurer '15, Hopkins-Schramm-Shi-Steurer '15]
- Tensor unfolding [Richard-Montanari '14, Hopkins-Shi-Steurer '15]

These are poly-time and succeed when $\lambda \gg n^{-p/4}$

SoS lower bounds suggest no poly-time algorithm when $\lambda \ll n^{-p/4}$

[Hopkins-Shi-Steurer '15, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17]

Algorithms for Tensor PCA

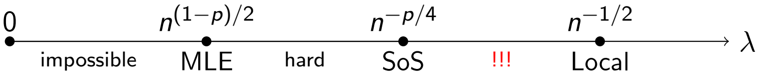
Sum-of-squares (SoS) and spectral methods:

- SoS semidefinite program [Hopkins-Shi-Steurer '15]
- Spectral SoS [Hopkins-Shi-Steurer '15, Hopkins-Schramm-Shi-Steurer '15]
- Tensor unfolding [Richard-Montanari '14, Hopkins-Shi-Steurer '15]

These are poly-time and succeed when $\lambda \gg n^{-p/4}$

SoS lower bounds suggest no poly-time algorithm when $\lambda \ll n^{-p/4}$

[Hopkins-Shi-Steurer '15, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17]



Local algorithms (gradient descent, AMP, ...) are suboptimal when $p \geq 3$

Subexponential-Time Algorithms

Subexponential-time: 2^{n^δ} for $\delta \in (0, 1)$

Subexponential-Time Algorithms

Subexponential-time: 2^{n^δ} for $\delta \in (0,1)$

Tensor PCA has a smooth tradeoff between runtime and statistical power: for $\delta \in (0,1)$,

there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

[Raghavendra-Rao-Schramm '16, Bhattiprolu-Guruswami-Lee '16]

Subexponential-Time Algorithms

Subexponential-time: 2^{n^δ} for $\delta \in (0,1)$

Tensor PCA has a smooth tradeoff between runtime and statistical power: for $\delta \in (0,1)$,

there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

[Raghavendra-Rao-Schramm '16, Bhattiprolu-Guruswami-Lee '16]

Interpolates between SoS and MLE:

- $\delta = 0 \quad \Rightarrow \quad$ poly-time algorithm for $\lambda \sim n^{-p/4}$
- $\delta = 1 \quad \Rightarrow \quad$ 2^n-time algorithm for $\lambda \sim n^{(1-p)/2}$

Subexponential-Time Algorithms

Subexponential-time: 2^{n^δ} for $\delta \in (0,1)$

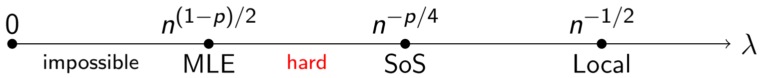
Tensor PCA has a smooth tradeoff between runtime and statistical power: for $\delta \in (0,1)$,

there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

[Raghavendra-Rao-Schramm '16, Bhattiprolu-Guruswami-Lee '16]

Interpolates between SoS and MLE:

- $\delta = 0 \quad \Rightarrow \quad$ poly-time algorithm for $\lambda \sim n^{-p/4}$
- $\delta = 1 \quad \Rightarrow \quad$ 2^n-time algorithm for $\lambda \sim n^{(1-p)/2}$



Subexponential-Time Algorithms

Subexponential-time: 2^{n^δ} for $\delta \in (0,1)$

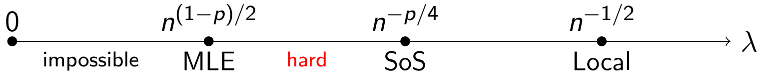
Tensor PCA has a smooth tradeoff between runtime and statistical power: for $\delta \in (0,1)$,

there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

[Raghavendra-Rao-Schramm '16, Bhattiprolu-Guruswami-Lee '16]

Interpolates between SoS and MLE:

- $\delta = 0 \quad \Rightarrow \quad$ poly-time algorithm for $\lambda \sim n^{-p/4}$
- $\delta = 1 \quad \Rightarrow \quad$ 2^n-time algorithm for $\lambda \sim n^{(1-p)/2}$



In contrast, some problems have a sharp threshold

- E.g., $\lambda > 1$ is nearly-linear time; $\lambda < 1$ needs time 2^n

Subexponential-Time Algorithms

Subexponential-time: 2^{n^δ} for $\delta \in (0,1)$

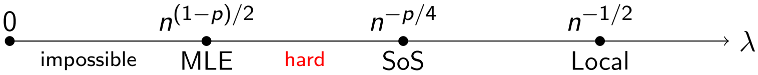
Tensor PCA has a smooth tradeoff between runtime and statistical power: for $\delta \in (0,1)$,

there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

[Raghavendra-Rao-Schramm '16, Bhattiprolu-Guruswami-Lee '16]

Interpolates between SoS and MLE:

- $\delta = 0 \quad \Rightarrow \quad$ poly-time algorithm for $\lambda \sim n^{-p/4}$
- $\delta = 1 \quad \Rightarrow \quad$ 2^n-time algorithm for $\lambda \sim n^{(1-p)/2}$



In contrast, some problems have a sharp threshold

- E.g., $\lambda > 1$ is nearly-linear time; $\lambda < 1$ needs time 2^n

For "soft" thresholds (like tensor PCA): BP/AMP can't be optimal

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

- A relatively simple calculation that predicts the computational complexity of high-dimensional inference problems

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

- A relatively simple calculation that predicts the computational complexity of high-dimensional inference problems
- Arose from the study of SoS lower bounds, pseudo-calibration
  [Barak-Hopkins-Kelner-Kothari-Moitra-Potechin '16, Hopkins-Steurer '17, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17, Hopkins PhD thesis '18]

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

- A relatively simple calculation that predicts the computational complexity of high-dimensional inference problems
- Arose from the study of SoS lower bounds, pseudo-calibration
  [Barak-Hopkins-Kelner-Kothari-Moitra-Potechin '16, Hopkins-Steurer '17, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17, Hopkins PhD thesis '18]
- Idea: look for a low-degree polynomial (of Y) that distinguishes $\mathbb{P}$ (spiked tensor) and $\mathbb{Q}$ (pure noise)

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

- A relatively simple calculation that predicts the computational complexity of high-dimensional inference problems
- Arose from the study of SoS lower bounds, pseudo-calibration
  [Barak-Hopkins-Kelner-Kothari-Moitra-Potechin '16, Hopkins-Steurer '17, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17, Hopkins PhD thesis '18]
- Idea: look for a low-degree polynomial (of Y) that distinguishes $\mathbb{P}$ (spiked tensor) and $\mathbb{Q}$ (pure noise)

$$\max_{f \text{ degree } \leq D} \frac{\mathbb{E}_{Y\sim\mathbb{P}}[f(Y)]}{\sqrt{\mathbb{E}_{Y\sim\mathbb{Q}}[f(Y)^2]}} \overset{?}{=} \begin{cases} O(1) & \Rightarrow \quad \text{"hard"} \\ \omega(1) & \Rightarrow \quad \text{"easy"} \end{cases}$$

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

- A relatively simple calculation that predicts the computational complexity of high-dimensional inference problems
- Arose from the study of SoS lower bounds, pseudo-calibration
  [Barak-Hopkins-Kelner-Kothari-Moitra-Potechin '16, Hopkins-Steurer '17, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17, Hopkins PhD thesis '18]
- Idea: look for a low-degree polynomial (of Y) that distinguishes $\mathbb{P}$ (spiked tensor) and $\mathbb{Q}$ (pure noise)

$$\max_{f \text{ degree } \le D} \frac{\mathbb{E}_{Y\sim\mathbb{P}}[f(Y)]}{\sqrt{\mathbb{E}_{Y\sim\mathbb{Q}}[f(Y)^2]}} \stackrel{?}{=} \begin{cases} O(1) & \Rightarrow & \text{"hard"} \\ \omega(1) & \Rightarrow & \text{"easy"} \end{cases}$$

- Take deg-D polynomials as a proxy for $n^{\tilde{\Theta}(D)}$-time algorithms

Aside: Low-Degree Likelihood Ratio

Recall: there is a 2^{n^δ}-time algorithm for $\lambda \sim n^{-p/4+\delta(1/2-p/4)}$

Evidence that this tradeoff is optimal: low-degree likelihood ratio

- A relatively simple calculation that predicts the computational complexity of high-dimensional inference problems
- Arose from the study of SoS lower bounds, pseudo-calibration
  [Barak-Hopkins-Kelner-Kothari-Moitra-Potechin '16, Hopkins-Steurer '17, Hopkins-Kothari-Potechin-Raghavendra-Schramm-Steurer '17, Hopkins PhD thesis '18]
- Idea: look for a low-degree polynomial (of Y) that distinguishes $\mathbb{P}$ (spiked tensor) and $\mathbb{Q}$ (pure noise)

$$\max_{f \text{ degree } \le D} \frac{\mathbb{E}_{Y\sim\mathbb{P}}[f(Y)]}{\sqrt{\mathbb{E}_{Y\sim\mathbb{Q}}[f(Y)^2]}} \stackrel{?}{=} \begin{cases} O(1) & \Rightarrow \quad \text{“hard”} \\ \omega(1) & \Rightarrow \quad \text{“easy”} \end{cases}$$

- Take deg-D polynomials as a proxy for $n^{\tilde{\Theta}(D)}$-time algorithms

For more, see the survey Kunisky-W.-Bandeira, “Notes on Computational Hardness of Hypothesis Testing: Predictions using the Low-Degree Likelihood Ratio”, arXiv:1907.11636

Our Contributions

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]
  - Based on Kikuchi free energy [Kikuchi '51]

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]
  - Based on Kikuchi free energy [Kikuchi '51]
- We prove that these algorithms match the performance of SoS
  - Both for poly-time and for subexponential-time tradeoff

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]
  - Based on Kikuchi free energy [Kikuchi '51]
- We prove that these algorithms match the performance of SoS
  - Both for poly-time and for subexponential-time tradeoff
- This refines and "redeems" the statistical physics approach to algorithm design

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]
  - Based on Kikuchi free energy [Kikuchi '51]
- We prove that these algorithms match the performance of SoS
  - Both for poly-time and for subexponential-time tradeoff
- This refines and "redeems" the statistical physics approach to algorithm design
- Our algorithms and analysis are simpler than prior work

Our Contributions

- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]
  - Based on Kikuchi free energy [Kikuchi '51]
- We prove that these algorithms match the performance of SoS
  - Both for poly-time and for subexponential-time tradeoff
- This refines and "redeems" the statistical physics approach to algorithm design
- Our algorithms and analysis are simpler than prior work
- This talk: even-order tensors only

Our Contributions

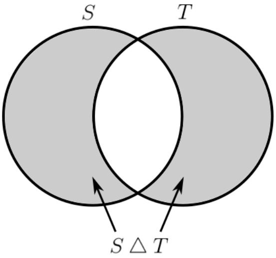
- For tensor PCA, we give a hierarchy of increasingly powerful BP/AMP-type algorithms: level ℓ "reasons about" ℓ-way interactions and requires $n^{O(\ell)}$ time
  - Specifically: Kikuchi Hessian spectral method
  - Generalization of Bethe Hessian [Saade, Krzakala, Zdeborová '14]
  - Based on Kikuchi free energy [Kikuchi '51]
- We prove that these algorithms match the performance of SoS
  - Both for poly-time and for subexponential-time tradeoff
- This refines and "redeems" the statistical physics approach to algorithm design
- Our algorithms and analysis are simpler than prior work
- This talk: even-order tensors only
- Similar results for odd-order tensors and refuting random k-XOR formulas (with k even)

The Algorithm

Definition (Symmetric Difference Matrix)

Input: an order-p tensor $Y = (Y_U)_{|U|=p}$ (with p even) and an integer ℓ in the range $p/2 \le \ell \le n - p/2$. Define the $\binom{n}{\ell} \times \binom{n}{\ell}$ matrix (indexed by ℓ-subsets of $[n]$)

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases}$$



The Algorithm

Definition (Symmetric Difference Matrix)

Input: an order-p tensor $Y = (Y_U)_{|U|=p}$ (with p even) and an integer ℓ in the range $p/2 \le \ell \le n - p/2$. Define the $\binom{n}{\ell} \times \binom{n}{\ell}$ matrix (indexed by ℓ-subsets of $[n]$)

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases}$$

- Algorithm: compute leading eigenvalue/eigenvector of M

The Algorithm

Definition (Symmetric Difference Matrix)

Input: an order-p tensor $Y = (Y_U)_{|U|=p}$ (with p even) and an integer ℓ in the range $p/2 \leq \ell \leq n - p/2$. Define the $\binom{n}{\ell} \times \binom{n}{\ell}$ matrix (indexed by ℓ-subsets of $[n]$)

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases}$$

- Algorithm: compute leading eigenvalue/eigenvector of M
- Runtime: $n^{O(\ell)}$

The Algorithm

Definition (Symmetric Difference Matrix)

Input: an order-p tensor $Y = (Y_U)_{|U|=p}$ (with p even) and an integer ℓ in the range $p/2 \le \ell \le n - p/2$. Define the $\binom{n}{\ell} \times \binom{n}{\ell}$ matrix (indexed by ℓ-subsets of $[n]$)

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases}$$

- Algorithm: compute leading eigenvalue/eigenvector of M
- Runtime: $n^{O(\ell)}$
- The case $\ell = p/2$ is "tensor unfolding," which is poly-time and succeeds up to the SoS threshold

The Algorithm

Definition (Symmetric Difference Matrix)

Input: an order-p tensor $Y = (Y_U)_{|U|=p}$ (with p even) and an integer ℓ in the range $p/2 \le \ell \le n - p/2$. Define the $\binom{n}{\ell} \times \binom{n}{\ell}$ matrix (indexed by ℓ-subsets of $[n]$)

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases}$$

- Algorithm: compute leading eigenvalue/eigenvector of M
- Runtime: $n^{O(\ell)}$
- The case $\ell = p/2$ is "tensor unfolding," which is poly-time and succeeds up to the SoS threshold
- $\ell = n^\delta$ gives an algorithm of runtime $n^{O(n^\ell)} = 2^{n^{\delta+o(1)}}$

The Algorithm

Definition (Symmetric Difference Matrix)

Input: an order-p tensor $Y = (Y_U)_{|U|=p}$ (with p even) and an integer ℓ in the range $p/2 \le \ell \le n - p/2$. Define the $\binom{n}{\ell} \times \binom{n}{\ell}$ matrix (indexed by ℓ-subsets of $[n]$)

$$M_{S,T} = \begin{cases} Y_{S \triangle T} & \text{if } |S \triangle T| = p, \\ 0 & \text{otherwise.} \end{cases}$$

- Algorithm: compute leading eigenvalue/eigenvector of M
- Runtime: $n^{O(\ell)}$
- The case $\ell = p/2$ is "tensor unfolding," which is poly-time and succeeds up to the SoS threshold
- $\ell = n^\delta$ gives an algorithm of runtime $n^{O(n^\ell)} = 2^{n^{\delta+o(1)}}$
- Algorithm derived from Kikuchi Hessian

Intuition for Symmetric Difference Matrix

Recall: $M_{S,T} = \mathbb{1}_{|S \triangle T| = p} Y_{S \triangle T}$ where $|S| = |T| = \ell$

Intuition for Symmetric Difference Matrix

Recall: $M_{S,T} = \mathbb{1}_{|S \triangle T|=p} Y_{S \triangle T}$ where $|S| = |T| = \ell$

Compute top eigenvector via power iteration: $v \leftarrow Mv$

- $v \in \mathbb{R}^{\binom{n}{\ell}}$ where v_S is an estimate of $x^S := \prod_{i \in S} x_i$

Intuition for Symmetric Difference Matrix

Recall: $M_{S,T} = \mathbb{1}_{|S \triangle T|=p} Y_{S \triangle T}$ where $|S| = |T| = \ell$

Compute top eigenvector via power iteration: $v \leftarrow Mv$

- $v \in \mathbb{R}^{\binom{n}{\ell}}$ where v_S is an estimate of $x^S := \prod_{i \in S} x_i$

Expand formula $v \leftarrow Mv$:

$$v_S \leftarrow \sum_{T : |S \triangle T| = p} Y_{S \triangle T} \, v_T$$

- Recall: $Y_{S \triangle T}$ is a noisy measurement of $x^{S \triangle T}$
- So $Y_{S \triangle T} \, v_T$ is T's opinion about x^S (since $x^S = x^{S \triangle T} x^T$)

Intuition for Symmetric Difference Matrix

Recall: $M_{S,T} = \mathbb{1}_{|S \triangle T|=p} Y_{S \triangle T}$ where $|S| = |T| = \ell$

Compute top eigenvector via power iteration: $v \leftarrow Mv$

- $v \in \mathbb{R}^{\binom{n}{\ell}}$ where v_S is an estimate of $x^S := \prod_{i \in S} x_i$

Expand formula $v \leftarrow Mv$:

$$v_S \leftarrow \sum_{T:|S \triangle T|=p} Y_{S \triangle T}\, v_T$$

- Recall: $Y_{S \triangle T}$ is a noisy measurement of $x^{S \triangle T}$
- So $Y_{S \triangle T}\, v_T$ is T's opinion about x^S (since $x^S = x^{S \triangle T} x^T$)

This is a message-passing algorithm among sets of size ℓ

Analysis

Simplest statistical task: detection

- Distinguish between $\lambda = \bar{\lambda}$ (spiked tensor) and $\lambda = 0$ (noise)

Analysis

Simplest statistical task: detection

- Distinguish between $\lambda = \bar{\lambda}$ (spiked tensor) and $\lambda = 0$ (noise)

Algorithm: given Y, build matrix $M_{S,T} = \mathbb{1}_{|S \triangle T| = p} Y_{S \triangle T}$, threshold maximum eigenvalue

Analysis

Simplest statistical task: detection

- Distinguish between $\lambda = \bar{\lambda}$ (spiked tensor) and $\lambda = 0$ (noise)

Algorithm: given Y, build matrix $M_{S,T} = \mathbb{1}_{|S \triangle T|=p} Y_{S \triangle T}$, threshold maximum eigenvalue

Key step: bound spectral norm $\|M\|$ when $Y \sim$ i.i.d. $\mathcal{N}(0,1)$

Analysis

Simplest statistical task: detection

- Distinguish between $\lambda = \bar{\lambda}$ (spiked tensor) and $\lambda = 0$ (noise)

Algorithm: given Y, build matrix $M_{S,T} = \mathbb{1}_{|S \triangle T| = p} Y_{S \triangle T}$, threshold maximum eigenvalue

Key step: bound spectral norm $\|M\|$ when $Y \sim$ i.i.d. $\mathcal{N}(0,1)$

Theorem (Matrix Chernoff Bound [Oliveira '10, Tropp '10])

Let $M = \sum_i z_i A_i$ where $z_i \sim \mathcal{N}(0,1)$ independently and $\{A_i\}$ is a finite sequence of fixed symmetric $d \times d$ matrices. Then, for all $t \geq 0$,

$$\mathbb{P}\left(\|M\| \geq t\right) \leq 2de^{-t^2/2\sigma^2} \quad \textit{where} \quad \sigma^2 = \left\| \sum_i (A_i)^2 \right\|.$$

Analysis

Simplest statistical task: detection

- Distinguish between $\lambda = \bar{\lambda}$ (spiked tensor) and $\lambda = 0$ (noise)

Algorithm: given Y, build matrix $M_{S,T} = \mathbb{1}_{|S \triangle T| = p} Y_{S \triangle T}$, threshold maximum eigenvalue

Key step: bound spectral norm $\|M\|$ when $Y \sim$ i.i.d. $\mathcal{N}(0,1)$

Theorem (Matrix Chernoff Bound [Oliveira '10, Tropp '10])

Let $M = \sum_i z_i A_i$ where $z_i \sim \mathcal{N}(0,1)$ independently and $\{A_i\}$ is a finite sequence of fixed symmetric $d \times d$ matrices. Then, for all $t \geq 0$,

$$\mathbb{P}\left(\|M\| \geq t\right) \leq 2de^{-t^2/2\sigma^2} \quad \textit{where} \quad \sigma^2 = \left\|\sum_i (A_i)^2\right\|.$$

In our case, $\sum_i (A_i)^2$ is a multiple of the identity

Related Work

Related Work

- [Yedidia-Freeman-Weiss '01, "Understanding Belief Propagation and its Generalizations"]
  - Similar higher-order message passing algorithm

Related Work

- [Yedidia-Freeman-Weiss '01, "Understanding Belief Propagation and its Generalizations"]
  - Similar higher-order message passing algorithm
- [Hastings '19, "Classical and Quantum Algorithms for Tensor PCA"]
  - Similar construction (symmetric difference matrix) with different motivation: quantum
  - Hamiltonian of system of bosons

Related Work

- [Yedidia-Freeman-Weiss '01, "Understanding Belief Propagation and its Generalizations"]
  - Similar higher-order message passing algorithm
- [Hastings '19, "Classical and Quantum Algorithms for Tensor PCA"]
  - Similar construction (symmetric difference matrix) with different motivation: quantum
  - Hamiltonian of system of bosons
- [Biroli, Cammarota, Ricci-Tersenghi '19, "How to iron out rough landscapes and get optimal performances"]
  - A different form of "redemption" for local algorithms
  - Replicated gradient descent

Summary

Summary

- Local algorithms are suboptimal for tensor PCA

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations
  - Matches SoS (conjectured optimal)

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations
  - Matches SoS (conjectured optimal)
  - Proof is much simpler than prior work

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations
  - Matches SoS (conjectured optimal)
  - Proof is much simpler than prior work
- Future directions

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations
  - Matches SoS (conjectured optimal)
  - Proof is much simpler than prior work
- Future directions
  - Unify statistical physics and SoS?

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations
  - Matches SoS (conjectured optimal)
  - Proof is much simpler than prior work
- Future directions
  - Unify statistical physics and SoS?
  - Systematically obtain optimal spectral methods in general?

Summary

- Local algorithms are suboptimal for tensor PCA
  - E.g. gradient descent, AMP
  - Keep track of an n-dimensional state
  - Nearly-linear runtime
- "Redemption" for local algorithms and AMP
  - Hierarchy of message-passing algorithms: symm. diff. matrices
  - Keep track of beliefs about higher-order correlations
  - Matches SoS (conjectured optimal)
  - Proof is much simpler than prior work
- Future directions
  - Unify statistical physics and SoS?
  - Systematically obtain optimal spectral methods in general?

Thanks!