Computational lower bounds for Tensor PCA

Daniel Hsu
Columbia University

Joint work with Rishabh Dudeja (Columbia → Harvard)

May 20, 2021
IPAM Workshop: "Efficient Tensor Representations for Learning and Computational Complexity"

Tensor PCA [Montanari & Richard, 2014]

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda \, \theta^{\otimes k} + \mathbf{Z}_i$$

- $\theta \in \Theta = S^{d-1}$: parameter vector to estimate (up to sign)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Tensor PCA [Montanari & Richard, 2014]

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda \, \theta^{\otimes k} + \mathbf{Z}_i$$

- $\theta \in \Theta = S^{d-1}$: parameter vector to estimate (up to sign)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Motivations:

- $k = 2$: spiked Wigner model, for studying (matrix) PCA
- $k \geq 2$: stylized model for studying tensor-based method-of-moments [e.g., AGHKT'14]

Tensor PCA [Montanari & Richard, 2014]

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda\, \theta^{\otimes k} + \mathbf{Z}_i$$

- $\theta \in \Theta = S^{d-1}$: parameter vector to estimate (up to sign)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Motivations:

- $k = 2$: spiked Wigner model, for studying (matrix) PCA
- $k \geq 2$: stylized model for studying tensor-based method-of-moments [e.g., AGHKT'14]

Matrix case ($k = 2$):

- Compute top eigenvector (e.g., $\log d$ iterations of power method); works if $N \gtrsim d/\lambda^2$

Tensor PCA [Montanari & Richard, 2014]

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda \, \theta^{\otimes k} + \mathbf{Z}_i$$

- $\theta \in \Theta = S^{d-1}$: parameter vector to estimate (up to sign)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Motivations:

- $k = 2$: spiked Wigner model, for studying (matrix) PCA
- $k \geq 2$: stylized model for studying tensor-based method-of-moments [e.g., AGHKT'14]

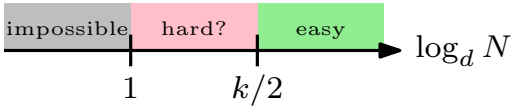
Matrix case ($k = 2$):

- Compute top eigenvector (e.g., $\log d$ iterations of power method); works if $N \gtrsim d/\lambda^2$

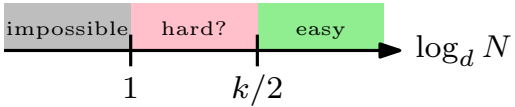
Computational-statistical gap ($k \geq 3$):

- Information-theoretically impossible if $N \lesssim d/\lambda^2$
- Maximum likelihood estimation works if $N \gtrsim d/\lambda^2$, but may need exponential time
- Known efficient algorithms require $N \gtrsim d^{k/2}/\lambda^2$

Computational hardness in intermediate sample size regime?



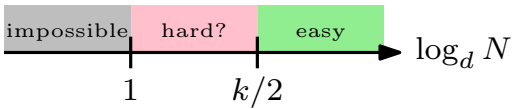
Computational hardness in intermediate sample size regime?



Known efficient algorithms:

- Matricization + matrix SVD [MR'14, HSS'15, ZT'15, HSSS'16, ...]

Computational hardness in intermediate sample size regime?



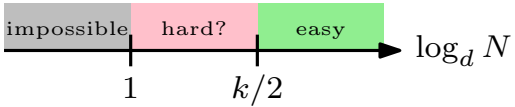
Known efficient algorithms:

- Matricization + matrix SVD [MR'14, HSS'15, ZT'15, HSSS'16, ...]

Failure of specific computational methods:

- Local search methods [MR'14; BAGJ'20]
- Sum-of-Squares (SoS) relaxations [HKPRSS'17]

Computational hardness in intermediate sample size regime?



Known efficient algorithms:

- Matricization + matrix SVD [MR'14, HSS'15, ZT'15, HSSS'16, ...]

Failure of specific computational methods:

- Local search methods [MR'14; BAGJ'20]
- Sum-of-Squares (SoS) relaxations [HKPRSS'17]

Other evidence/suggestions of computational hardness:

- Reduction from Hypergraphic Planted Clique [ZX'18; BB'20]
- Low-degree polynomial heuristic [KWB'19]
- Failure of Statistical Query (SQ) algorithms [DH'20; BBHLS'20]

What we do

Goals:

1. Prove (modest!) lower bounds for arbitrary algorithms under computational resource constraints
2. Gain some insight into role of over-parameterization

What we do

Goals:

1. Prove (modest!) lower bounds for arbitrary algorithms under computational resource constraints
2. Gain some insight into role of over-parameterization

Main theorem (informal). Every algorithm for $\mathrm{TPCA}(d, k, \lambda^2)$ that accurately estimates θ uses

$$\text{memory size} \quad \times \quad \text{iterations} \quad \times \quad \text{sample size} \quad \gtrsim \quad \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2}$$

What we do

Goals:

1. Prove (modest!) lower bounds for arbitrary algorithms under computational resource constraints
2. Gain some insight into role of over-parameterization

Main theorem (informal). Every algorithm for $\mathrm{TPCA}(d, k, \lambda^2)$ that accurately estimates θ uses

$$\text{memory size} \quad \times \quad \text{iterations} \quad \times \quad \text{sample size} \quad \gtrsim \quad \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2}$$

Techniques:

- Reduction from distributed estimation in blackboard model [Shamir, 2014; Dagan & Shamir, 2018]
- New communication lower bounds for Tensor PCA in blackboard model

What we do

Goals:

1. Prove (modest!) lower bounds for arbitrary algorithms under computational resource constraints
2. Gain some insight into role of over-parameterization

Main theorem (informal). Every algorithm for $\mathrm{TPCA}(d, k, \lambda^2)$ that accurately estimates θ uses

$$\text{memory size} \quad \times \quad \text{iterations} \quad \times \quad \text{sample size} \quad \gtrsim \quad \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2}$$

Techniques:

- Reduction from distributed estimation in blackboard model [Shamir, 2014; Dagan & Shamir, 2018]
- New communication lower bounds for Tensor PCA in blackboard model

Also: Similar results for "Asymmetric Tensor PCA" and other related problems

Talk outline

1. Computational model and lower bounds for Tensor PCA
2. Lower bounds for Asymmetric Tensor PCA and benefits of over-parameterization
3. Some comments on proof via communication complexity

Computational model and lower bounds for Tensor PCA

Computational model for memory-bounded algorithms

Algorithm template:

- Initialize memory $\texttt{state} \in \{0,1\}^B$
- For iteration $t = 1, 2, \ldots, T$:
  - For data point $i = 1, 2, \ldots, N$:

    $$\texttt{state} \leftarrow \mathrm{update}_{t,i}(\texttt{state}, \mathbf{X}_i)$$

- Return $\hat{\theta}(\texttt{state})$

($\mathrm{update}_{t,i}(\cdot,\cdot)$ & $\hat{\theta}(\cdot)$ may be arbitrary functions)

Computational model for memory-bounded algorithms

Algorithm template:

- Initialize memory $\texttt{state} \in \{0,1\}^B$
- For iteration $t = 1, 2, \ldots, T$:
  - For data point $i = 1, 2, \ldots, N$:

    $$\texttt{state} \leftarrow \mathrm{update}_{t,i}(\texttt{state}, \mathbf{X}_i)$$

- Return $\hat{\theta}(\texttt{state})$

($\mathrm{update}_{t,i}(\cdot,\cdot)$ & $\hat{\theta}(\cdot)$ may be arbitrary functions)

Example: maximum likelihood estimation

$$\arg\max_{\hat{\theta} \in \Theta} \langle \bar{\mathbf{X}}, \hat{\theta}^{\otimes k} \rangle$$

where $\bar{\mathbf{X}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{X}_i \quad (\Theta = \{\pm \frac{1}{\sqrt{d}}\}^d)$

Computational model for memory-bounded algorithms

Algorithm template:

- Initialize memory $\texttt{state} \in \{0,1\}^B$
- For iteration $t = 1, 2, \ldots, T$:
  - For data point $i = 1, 2, \ldots, N$:

$$\texttt{state} \leftarrow \mathrm{update}_{t,i}(\texttt{state}, \mathbf{X}_i)$$

- Return $\hat{\theta}(\texttt{state})$

($\mathrm{update}_{t,i}(\cdot,\cdot)$ & $\hat{\theta}(\cdot)$ may be arbitrary functions)

Example: maximum likelihood estimation

$$\arg\max_{\hat{\theta} \in \Theta} \langle \bar{\mathbf{X}}, \hat{\theta}^{\otimes k} \rangle$$

where $\bar{\mathbf{X}} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{X}_i \quad (\Theta = \{\pm \frac{1}{\sqrt{d}}\}^d)$

- Linearity: $\langle \bar{\mathbf{X}}, \hat{\theta}^{\otimes k} \rangle = \sum_{i=1}^{N} \langle \frac{1}{N} \mathbf{X}_i, \hat{\theta}^{\otimes k} \rangle$
- state tracks best obj. value and $\hat{\theta}$ so far, and space for running sums ($B = O(d)$)
- Number of iterations: $T = 2^d$

Efficient algorithm for TPCA (even k)

Partial trace algorithm

[Hopkins, Schramm, Shi, Steurer, 2016]

- Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be given by

$$(\mathbf{A})_{i,j} = \sum_{a,b,\ldots \in [d]} (\bar{\mathbf{X}})_{a,a,b,b,\ldots,i,j}$$

- $\hat{\theta}$ = top eigenvector of $\mathbf{A}$
  (via power method)

Efficient algorithm for TPCA (even k)

Partial trace algorithm

[Hopkins, Schramm, Shi, Steurer, 2016]

- Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be given by

$$(\mathbf{A})_{i,j} = \sum_{a,b,\ldots \in [d]} (\bar{\mathbf{X}})_{a,a,b,b,\ldots,i,j}$$

- $\hat{\theta}$ = top eigenvector of $\mathbf{A}$
  (via power method)

- **Recall:**

$$\bar{\mathbf{X}} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes k} + \frac{1}{\sqrt{N}} \mathbf{Z}$$

Efficient algorithm for TPCA (even k)

Partial trace algorithm

[Hopkins, Schramm, Shi, Steurer, 2016]

- Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be given by

$$(\mathbf{A})_{i,j} = \sum_{a,b,\ldots \in [d]} (\bar{\mathbf{X}})_{a,a,b,b,\ldots,i,j}$$

- $\hat{\theta}$ = top eigenvector of $\mathbf{A}$
  (via power method)

- **Recall:**

$$\bar{\mathbf{X}} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes k} + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Partial trace matrix:**

$$\mathbf{A} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes 2} + \frac{1}{\sqrt{N}} \sqrt{d^{\frac{k}{2}-1}}\, \mathbf{Z}'$$

Efficient algorithm for TPCA (even k)

Partial trace algorithm

[Hopkins, Schramm, Shi, Steurer, 2016]

- Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be given by

$$(\mathbf{A})_{i,j} = \sum_{a,b,\ldots \in [d]} (\bar{\mathbf{X}})_{a,a,b,b,\ldots,i,j}$$

- $\hat{\theta}$ = top eigenvector of $\mathbf{A}$
  (via power method)

- **Recall:**

$$\bar{\mathbf{X}} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes k} + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Partial trace matrix:**

$$\mathbf{A} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes 2} + \frac{1}{\sqrt{N}} \sqrt{d^{\frac{k}{2}-1}}\, \mathbf{Z}'$$

- **SNR:** reduces from λ^2 to $\lambda^2 / d^{\frac{k}{2}-1}$

Efficient algorithm for TPCA (even k)

Partial trace algorithm

[Hopkins, Schramm, Shi, Steurer, 2016]

- Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be given by

$$(\mathbf{A})_{i,j} = \sum_{a,b,\ldots \in [d]} (\bar{\mathbf{X}})_{a,a,b,b,\ldots,i,j}$$

- $\hat{\theta}$ = top eigenvector of $\mathbf{A}$ (via power method)

- **Recall:**

$$\bar{\mathbf{X}} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes k} + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Partial trace matrix:**

$$\mathbf{A} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes 2} + \frac{1}{\sqrt{N}} \sqrt{d^{\frac{k}{2}-1}}\, \mathbf{Z}'$$

- **SNR:** reduces from λ^2 to $\lambda^2 / d^{\frac{k}{2}-1}$

$$\mathrm{TPCA}(d, k, \lambda^2) \longrightarrow \mathrm{TPCA}(d, 2, \lambda^2 / d^{\frac{k}{2}-1})$$

Efficient algorithm for TPCA (even k)

Partial trace algorithm

[Hopkins, Schramm, Shi, Steurer, 2016]

- Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be given by

$$(\mathbf{A})_{i,j} = \sum_{a,b,\ldots \in [d]} (\bar{\mathbf{X}})_{a,a,b,b,\ldots,i,j}$$

- $\hat{\theta}$ = top eigenvector of $\mathbf{A}$
  (via power method)

- **Recall:**

$$\bar{\mathbf{X}} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes k} + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Partial trace matrix:**

$$\mathbf{A} \stackrel{\mathrm{d}}{=} \lambda\, \theta^{\otimes 2} + \frac{1}{\sqrt{N}} \sqrt{d^{\frac{k}{2}-1}}\, \mathbf{Z}'$$

- **SNR:** reduces from λ^2 to $\lambda^2 / d^{\frac{k}{2}-1}$

$$\mathrm{TPCA}(d, k, \lambda^2) \longrightarrow \mathrm{TPCA}(d, 2, \lambda^2 / d^{\frac{k}{2}-1})$$

Upshot: Needs sample size

$$N \asymp \frac{d}{\lambda^2 / d^{\frac{k}{2}-1}} = \frac{d^{k/2}}{\lambda^2}$$

but works with $T \asymp \log d$ iterations and $B \asymp d$ bits of memory $\quad (B \times N \times T \asymp (d^{\frac{k}{2}+1} \log d)/\lambda^2)$

Main result #1: lower bounds for TPCA

Theorem 1. Suppose estimate $\hat{\theta}$ is computed by memory-bounded algorithm for $\mathrm{TPCA}(d, k, \lambda^2)$ with

$$B \times T \times N \ll \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2}$$

and $N \gg d/\lambda^2$. Then

$$\inf_{\theta \in \Theta} \mathbb{E}_\theta \left[\langle \theta, \hat{\theta} \rangle^2 \right] \lesssim \frac{\log d}{d}.$$

Main result #1: lower bounds for TPCA

Theorem 1. Suppose estimate $\hat{\theta}$ is computed by memory-bounded algorithm for $\mathrm{TPCA}(d, k, \lambda^2)$ with

$$B \times T \times N \ll \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2}$$

and $N \gg d/\lambda^2$. Then

$$\inf_{\theta \in \Theta} \mathbb{E}_\theta \left[\langle \theta, \hat{\theta} \rangle^2 \right] \lesssim \frac{\log d}{d}.$$

Remarks:

- ($\mathrm{TPCA}(d, k, \lambda^2)$ is information-theoretically impossible when $N \ll d/\lambda^2$)

Main result #1: lower bounds for TPCA

Theorem 1. Suppose estimate $\hat{\theta}$ is computed by memory-bounded algorithm for $\mathrm{TPCA}(d, k, \lambda^2)$ with

$$B \times T \times N \ll \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2}$$

and $N \gg d/\lambda^2$. Then

$$\inf_{\theta \in \Theta} \mathbb{E}_\theta \left[\langle \theta, \hat{\theta} \rangle^2 \right] \lesssim \frac{\log d}{d}.$$

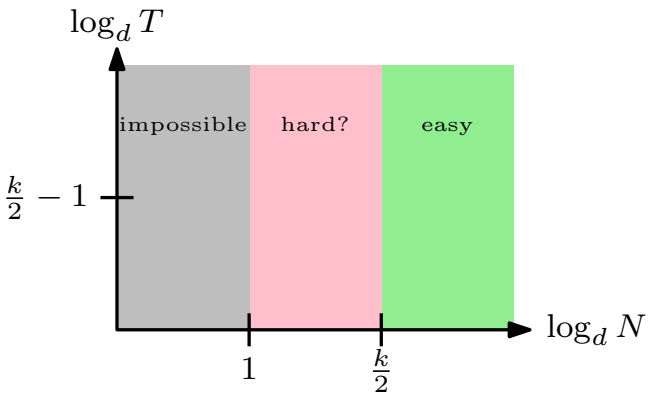
Remarks:

- ($\mathrm{TPCA}(d, k, \lambda^2)$ is information-theoretically impossible when $N \ll d/\lambda^2$)
- Theorem is an unconditional lower bound on computational and information resources:

If algorithm computes an accurate estimate,
then it must use enough resources, as measured by $B \times T \times N$

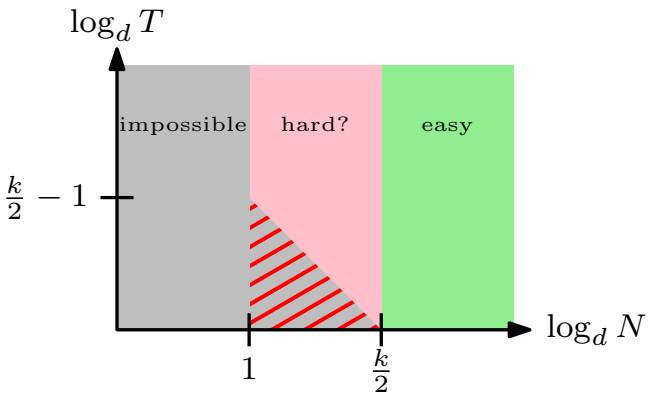
Run-time vs sample size in TPCA (even k)

Linear memory algorithms: $B \asymp d$ bits of memory



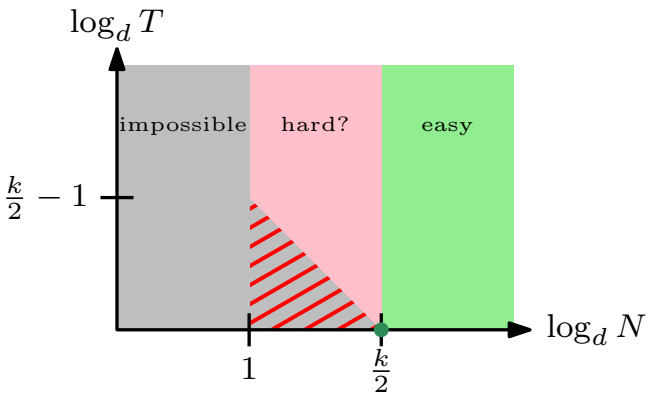
Run-time vs sample size in TPCA (even k)

Linear memory algorithms: $B \asymp d$ bits of memory



Run-time vs sample size in TPCA (even k)

Linear memory algorithms: $B \asymp d$ bits of memory



Cannot reduce sample complexity of partial trace algorithm without increasing memory or run-time

Lower bounds for Asymmetric Tensor PCA

Asymmetric Tensor PCA

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda\, \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_k + \mathbf{Z}_i$$

- $\theta_1, \theta_2, \ldots, \theta_k \in \Theta = S^{d-1}$: parameter vectors to estimate (up to tensor product)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Asymmetric Tensor PCA

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda\, \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_k + \mathbf{Z}_i$$

- $\theta_1, \theta_2, \dots, \theta_k \in \Theta = S^{d-1}$: parameter vectors to estimate (up to tensor product)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Matrix case ($k = 2$):

- Compute top singular vectors (e.g., $\log d$ iterations of power method); works if $N \gtrsim d/\lambda^2$

Asymmetric Tensor PCA

Data model: iid random order-k tensors $\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_N$ in $\bigotimes^k \mathbb{R}^d$

$$\mathbf{X}_i = \lambda\, \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_k + \mathbf{Z}_i$$

- $\theta_1, \theta_2, \ldots, \theta_k \in \Theta = S^{d-1}$: parameter vectors to estimate (up to tensor product)
- $\lambda^2 \in \mathbb{R}_+$: signal-to-noise ratio per data point
- $\mathbf{Z}_i$: order-k tensor of d^k iid standard normal random variables

Matrix case ($k = 2$):

- Compute top singular vectors (e.g., $\log d$ iterations of power method); works if $N \gtrsim d/\lambda^2$

Computational-statistical gap ($k \geq 3$):

- Information-theoretically impossible if $N \lesssim d/\lambda^2$
- Maximum likelihood estimation works if $N \gtrsim d/\lambda^2$, but may need exponential time
- Known efficient algorithms require $N \gtrsim d^{k/2}/\lambda^2$

Efficient algorithm for ATPCA (even k)

Matricization algorithm

[Montanari & Richard, 2014]

- Let $\mathbf{A} = \mathrm{mat}(\bar{\mathbf{X}}) \in \mathbb{R}^{d^{k/2} \times d^{k/2}}$
- $\tilde{\mathbf{A}} =$ rank-1 SVD of $\mathbf{A}$
  (via power method)
- $(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_k) = \mathrm{mat}^{-1}(\tilde{\mathbf{A}})$

Efficient algorithm for ATPCA (even k)

Matricization algorithm

[Montanari & Richard, 2014]

- Let $\mathbf{A} = \mathrm{mat}(\bar{\mathbf{X}}) \in \mathbb{R}^{d^{k/2} \times d^{k/2}}$
- $\tilde{\mathbf{A}} =$ rank-1 SVD of $\mathbf{A}$ (via power method)
- $(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_k) = \mathrm{mat}^{-1}(\tilde{\mathbf{A}})$

- **Recall:**

$$\bar{\mathbf{X}} \stackrel{\mathrm{d}}{=} \lambda\, \theta_1 \otimes \cdots \otimes \theta_k + \frac{1}{\sqrt{N}} \mathbf{Z}$$

Efficient algorithm for ATPCA (even k)

Matricization algorithm
[Montanari & Richard, 2014]

- Let $\mathbf{A} = \mathrm{mat}(\bar{\mathbf{X}}) \in \mathbb{R}^{d^{k/2} \times d^{k/2}}$
- $\tilde{\mathbf{A}} =$ rank-1 SVD of $\mathbf{A}$ (via power method)
- $(\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_k) = \mathrm{mat}^{-1}(\tilde{\mathbf{A}})$

- **Recall:**

$$\bar{\mathbf{X}} \overset{\mathrm{d}}{=} \lambda\, \theta_1 \otimes \cdots \otimes \theta_k + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Matricization:**

$$\mathbf{A} \overset{\mathrm{d}}{=} \lambda\, u \otimes v + \frac{1}{\sqrt{N}} \mathrm{mat}(\mathbf{Z})$$
$$u = \mathrm{vec}(\theta_1 \otimes \cdots \otimes \theta_{k/2})$$
$$v = \mathrm{vec}(\theta_{k/2+1} \otimes \cdots \otimes \theta_k)$$

Efficient algorithm for ATPCA (even k)

Matricization algorithm
[Montanari & Richard, 2014]

- Let $\mathbf{A} = \mathrm{mat}(\bar{\mathbf{X}}) \in \mathbb{R}^{d^{k/2} \times d^{k/2}}$
- $\tilde{\mathbf{A}} =$ rank-1 SVD of $\mathbf{A}$ (via power method)
- $(\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_k) = \mathrm{mat}^{-1}(\tilde{\mathbf{A}})$

- **Recall:**

$$\bar{\mathbf{X}} \overset{\mathrm{d}}{=} \lambda\, \theta_1 \otimes \cdots \otimes \theta_k + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Matricization:**

$$\mathbf{A} \overset{\mathrm{d}}{=} \lambda\, u \otimes v + \frac{1}{\sqrt{N}} \mathrm{mat}(\mathbf{Z})$$
$$u = \mathrm{vec}(\theta_1 \otimes \cdots \otimes \theta_{k/2})$$
$$v = \mathrm{vec}(\theta_{k/2+1} \otimes \cdots \otimes \theta_k)$$

$$\mathrm{ATPCA}(d, k, \lambda^2) \longrightarrow \mathrm{ATPCA}(d^{k/2}, 2, \lambda^2)$$

Efficient algorithm for ATPCA (even k)

Matricization algorithm
[Montanari & Richard, 2014]

- Let $\mathbf{A} = \mathrm{mat}(\bar{\mathbf{X}}) \in \mathbb{R}^{d^{k/2} \times d^{k/2}}$
- $\tilde{\mathbf{A}} =$ rank-1 SVD of $\mathbf{A}$ (via power method)
- $(\hat{\theta}_1, \hat{\theta}_2, \dots, \hat{\theta}_k) = \mathrm{mat}^{-1}(\tilde{\mathbf{A}})$

- **Recall:**

$$\bar{\mathbf{X}} \overset{\mathrm{d}}{=} \lambda\, \theta_1 \otimes \cdots \otimes \theta_k + \frac{1}{\sqrt{N}} \mathbf{Z}$$

- **Matricization:**

$$\mathbf{A} \overset{\mathrm{d}}{=} \lambda\, u \otimes v + \frac{1}{\sqrt{N}} \mathrm{mat}(\mathbf{Z})$$
$$u = \mathrm{vec}(\theta_1 \otimes \cdots \otimes \theta_{k/2})$$
$$v = \mathrm{vec}(\theta_{k/2+1} \otimes \cdots \otimes \theta_k)$$

$$\mathrm{ATPCA}(d, k, \lambda^2) \longrightarrow \mathrm{ATPCA}(d^{k/2}, 2, \lambda^2)$$

Upshot: Needs sample size

$$N \asymp \frac{d^{k/2}}{\lambda^2}$$

and $B \asymp d^{k/2}$ bits of memory; works with $T \asymp \log d$ iterations $\quad (B \times T \times N \asymp (d^k \log d)/\lambda^2)$

Main result #2: lower bounds for ATPCA

Theorem 2. Suppose estimate $(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_k)$ is computed by memory-bounded algorithm for $\mathrm{ATPCA}(d, k, \lambda^2)$ with

$$B \times T \times N \ll \frac{d^k}{\lambda^2}$$

and $N \gg d/\lambda^2$. Then

$$\inf_{\theta_1,\theta_2,\ldots,\theta_k \in \Theta} \mathbb{E}_{(\theta_1,\theta_2,\ldots,\theta_k)} \left[\left\| \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_k - \hat{\theta}_1 \otimes \hat{\theta}_2 \otimes \cdots \otimes \hat{\theta}_k \right\|^2 \right] \geq \frac{1}{32}.$$

Main result #2: lower bounds for ATPCA

Theorem 2. Suppose estimate $(\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_k)$ is computed by memory-bounded algorithm for $\mathrm{ATPCA}(d, k, \lambda^2)$ with

$$B \times T \times N \ll \frac{d^k}{\lambda^2}$$

and $N \gg d/\lambda^2$. Then

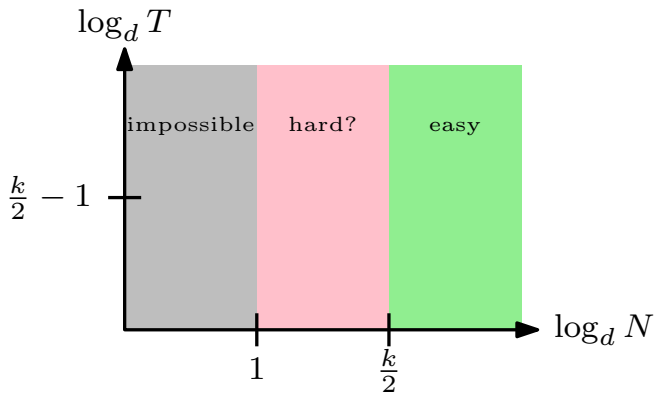
$$\inf_{\theta_1,\theta_2,\ldots,\theta_k \in \Theta} \mathbb{E}_{(\theta_1,\theta_2,\ldots,\theta_k)} \left[\left\| \theta_1 \otimes \theta_2 \otimes \cdots \otimes \theta_k - \hat{\theta}_1 \otimes \hat{\theta}_2 \otimes \cdots \otimes \hat{\theta}_k \right\|^2 \right] \geq \frac{1}{32}.$$

Remarks:

- Implies strictly higher resource requirement than needed for TPCA for $k \geq 3$ (d^k vs $d^{\lceil (k+1)/2 \rceil}$)

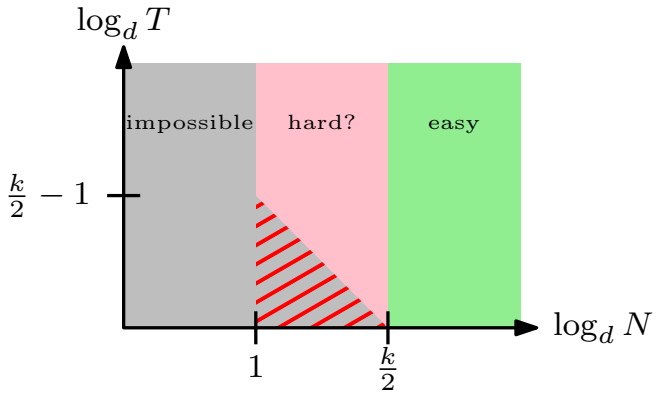
Run-time vs sample size for ATPCA

"Over-parameterized" algorithms: $B \asymp d^{k/2}$ bits of memory



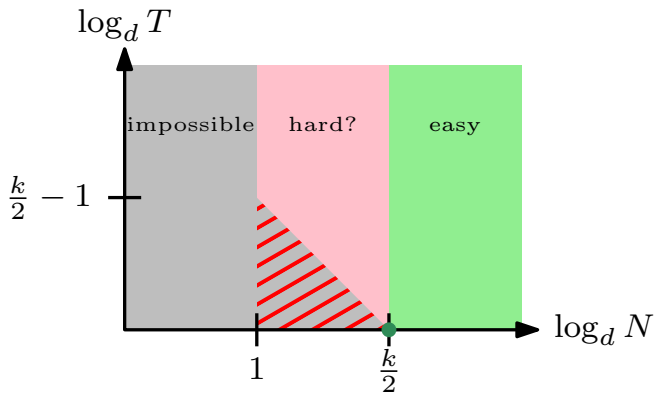
Run-time vs sample size for ATPCA

"Over-parameterized" algorithms: $B \asymp d^{k/2}$ bits of memory



Run-time vs sample size for ATPCA

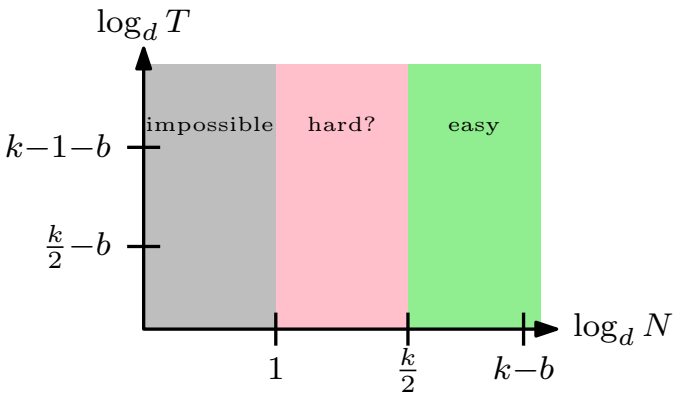
"Over-parameterized" algorithms: $B \asymp d^{k/2}$ bits of memory



Cannot reduce sample complexity of matricization algorithm without increasing memory or run-time

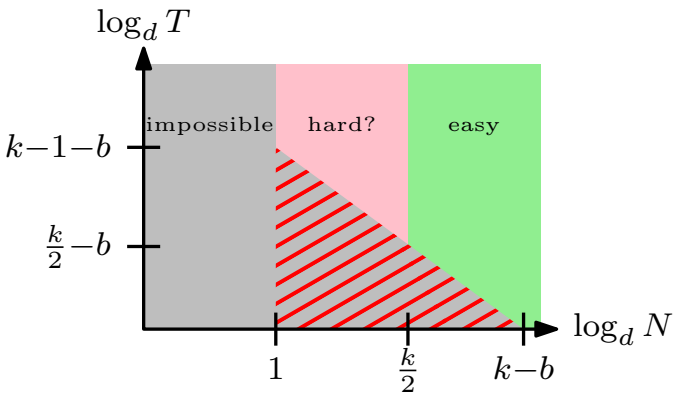
Price of limited over-parameterization in ATPCA

Limited over-parameterization: $B \asymp d^b$ bits of memory, for $b < k/2$



Price of limited over-parameterization in ATPCA

Limited over-parameterization: $B \asymp d^b$ bits of memory, for $b < k/2$

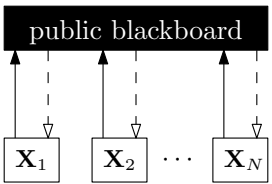


Algorithms with limited over-parameterization have higher run-time vs sample size requirements

Comments on proof via communication complexity

Proof strategy

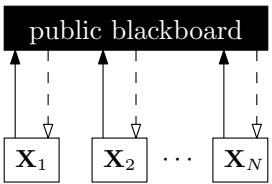
1. Reduction from distributed estimation in blackboard model [Shamir, 2014; Dagan & Shamir, 2018]



(B, T, N)-algorithm $\implies$ protocol with $B \times T \times N$ bits of communication

Proof strategy

1. Reduction from distributed estimation in blackboard model [Shamir, 2014; Dagan & Shamir, 2018]
2. New communication lower bounds for Tensor PCA in blackboard model



(B, T, N)-algorithm $\implies$ protocol with $B \times T \times N$ bits of communication

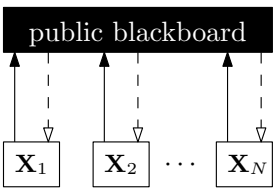
Theorem 3 (informal). Every protocol for $\mathrm{TPCA}(d, k, \lambda^2)$ that accurately estimates θ uses

$$\text{total communication} \gtrsim \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2} \text{ bits}$$

Blackboard model of communication [Bar-Yossef, Jayram, Kumar, Sivakumar, 2004]

Blackboard model of communication:

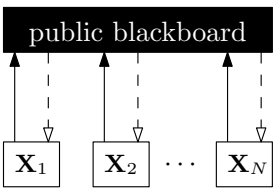
- N machines; machine i receives $\mathbf{X}_i$
- Blackboard (BB) visible to all machines (i.e. no cost to read BB contents)
- Machines take turns writing on BB (as dictated by protocol and BB contents)
- Estimate $\hat{\theta}$ is a function of BB contents



Blackboard model of communication [Bar-Yossef, Jayram, Kumar, Sivakumar, 2004]

Blackboard model of communication:

- N machines; machine i receives $\mathbf{X}_i$
- Blackboard (BB) visible to all machines (i.e. no cost to read BB contents)
- Machines take turns writing on BB (as dictated by protocol and BB contents)
- Estimate $\hat{\theta}$ is a function of BB contents



Reduction [Shamir, 2014; Dagan & Shamir, 2018]

Given (B, T, N)-algorithm $(\mathrm{update}_{t,i}(\cdot,\cdot), \hat{\theta}(\cdot))$, define protocol:

- (Assume initial `state` already on BB)
- For $t = 1, 2, \ldots, T$, and for $i = 1, 2, \ldots, N$:
  - Machine i reads last `state` on BB
  - Machine i computes new

$$\texttt{state} \leftarrow \mathrm{update}_{t,i}(\texttt{state}, \mathbf{X}_i)$$

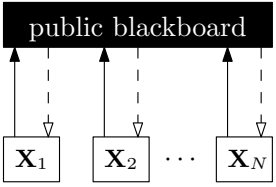
  - Machine i writes new `state` on BB
- Return estimate:

$$\hat{\theta}(\text{final } \texttt{state} \text{ written on BB})$$

Blackboard model of communication [Bar-Yossef, Jayram, Kumar, Sivakumar, 2004]

Blackboard model of communication:

- N machines; machine i receives $\mathbf{X}_i$
- Blackboard (BB) visible to all machines (i.e. no cost to read BB contents)
- Machines take turns writing on BB (as dictated by protocol and BB contents)
- Estimate $\hat{\theta}$ is a function of BB contents



Reduction [Shamir, 2014; Dagan & Shamir, 2018]

Given (B, T, N)-algorithm $(\mathrm{update}_{t,i}(\cdot,\cdot), \hat{\theta}(\cdot))$, define protocol:

- (Assume initial `state` already on BB)
- For $t = 1, 2, \ldots, T$, and for $i = 1, 2, \ldots, N$:
  - Machine i reads last `state` on BB
  - Machine i computes new

$$\texttt{state} \leftarrow \mathrm{update}_{t,i}(\texttt{state}, \mathbf{X}_i)$$

  - Machine i writes new `state` on BB
- Return estimate:

$$\hat{\theta}(\text{final } \texttt{state} \text{ written on BB})$$

Total communication: $B \times T \times N$ bits

More comments on proof

- Leverage version of Fano's inequality for Hellinger information:

$$I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) = \inf_{\mathbb{Q}} \int \mathrm{h}^2(\mathbb{P}_\theta, \mathbb{Q}) \; \pi(\mathrm{d}\theta),$$

where π is prior distribution over Θ, and protocol transcript is $\mathbf{Y} \mid \boldsymbol{\theta} \sim \mathbb{P}_{\boldsymbol{\theta}}$
[Chen, Guntuboyina, Zhang, 2016]

More comments on proof

- Leverage version of Fano's inequality for Hellinger information:

$$I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) = \inf_{\mathbb{Q}} \int \mathrm{h}^2(\mathbb{P}_\theta, \mathbb{Q}) \; \pi(\mathrm{d}\theta),$$

where π is prior distribution over Θ, and protocol transcript is $\mathbf{Y} \mid \boldsymbol{\theta} \sim \mathbb{P}_{\boldsymbol{\theta}}$
[Chen, Guntuboyina, Zhang, 2016]

  - If $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$, then for sufficiently large d, every protocol will fail for some θ

More comments on proof

- Leverage version of Fano's inequality for Hellinger information:

$$I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) = \inf_{\mathbb{Q}} \int \mathrm{h}^2(\mathbb{P}_\theta, \mathbb{Q}) \; \pi(\mathrm{d}\theta),$$

where π is prior distribution over Θ, and protocol transcript is $\mathbf{Y} \mid \boldsymbol{\theta} \sim \mathbb{P}_{\boldsymbol{\theta}}$

[Chen, Guntuboyina, Zhang, 2016]

  - If $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$, then for sufficiently large d, every protocol will fail for some θ
  - We prove $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$ if

$$\text{total communication} \;\; \ll \;\; \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2} \;\; \text{bits}$$

More comments on proof

- Leverage version of Fano's inequality for Hellinger information:

$$I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) = \inf_{\mathbb{Q}} \int \mathrm{h}^2(\mathbb{P}_\theta, \mathbb{Q}) \; \pi(\mathrm{d}\theta),$$

where π is prior distribution over Θ, and protocol transcript is $\mathbf{Y} \mid \boldsymbol{\theta} \sim \mathbb{P}_{\boldsymbol{\theta}}$

[Chen, Guntuboyina, Zhang, 2016]

  - If $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$, then for sufficiently large d, every protocol will fail for some θ
  - We prove $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$ if

$$\text{total communication} \;\ll\; \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2} \text{ bits}$$

  - Leverage properties of blackboard protocols and Gaussian harmonic analysis

More comments on proof

- Leverage version of Fano's inequality for Hellinger information:

$$I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) = \inf_{\mathbb{Q}} \int \mathrm{h}^2(\mathbb{P}_\theta, \mathbb{Q}) \; \pi(\mathrm{d}\theta),$$

  where π is prior distribution over Θ, and protocol transcript is $\mathbf{Y} \mid \boldsymbol{\theta} \sim \mathbb{P}_{\boldsymbol{\theta}}$
  [Chen, Guntuboyina, Zhang, 2016]
  - If $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$, then for sufficiently large d, every protocol will fail for some θ
  - We prove $I_{\mathrm{h}}(\boldsymbol{\theta}; \mathbf{Y}) \to 0$ as $d \to \infty$ if

$$\text{total communication} \;\; \ll \;\; \frac{d^{\lceil (k+1)/2 \rceil}}{\lambda^2} \text{ bits}$$

  - Leverage properties of blackboard protocols and Gaussian harmonic analysis
- For ATPCA, communication lower bound is even simpler
  - In fact, a special case of lower bound for Sparse Gaussian Mean Estimation
    [Braverman, Garg, Ma, Nguyen, Woodruff, 2016]
  - We give a new proof using our framework

In closing . . .

- Of course, no proof yet of exponential complexity for general algorithms in conjectured hard regime of Tensor PCA and variants

In closing . . .

- Of course, no proof yet of exponential complexity for general algorithms in conjectured hard regime of Tensor PCA and variants
- Communication complexity can be used to establish **modest unconditional lower bounds**

In closing . . .

- Of course, no proof yet of exponential complexity for general algorithms in conjectured hard regime of Tensor PCA and variants
- Communication complexity can be used to establish **modest unconditional lower bounds**
  - Some **known efficient algorithms are unimprovable** without degrading some resource complexity measure

In closing . . .

- Of course, no proof yet of exponential complexity for general algorithms in conjectured hard regime of Tensor PCA and variants
- Communication complexity can be used to establish **modest unconditional lower bounds**
  - Some **known efficient algorithms are unimprovable** without degrading some resource complexity measure
  - **Computational & statistical benefits of "over-parameterization"**

In closing . . .

- Of course, no proof yet of exponential complexity for general algorithms in conjectured hard regime of Tensor PCA and variants
- Communication complexity can be used to establish **modest unconditional lower bounds**
  - Some **known efficient algorithms are unimprovable** without degrading some resource complexity measure
  - **Computational & statistical benefits of "over-parameterization"**
- Also have similar results for other estimation problems where tensor-based methods-of-moments have been used

In closing . . .

- Of course, no proof yet of exponential complexity for general algorithms in conjectured hard regime of Tensor PCA and variants
- Communication complexity can be used to establish **modest unconditional lower bounds**
  - Some **known efficient algorithms are unimprovable** without degrading some resource complexity measure
  - **Computational & statistical benefits of "over-parameterization"**
- Also have similar results for other estimation problems where tensor-based methods-of-moments have been used

---

Thank you!

We gratefully acknowledge support from NSF CCF-1740833, a Sloan Research Fellowship, and a Bloomberg Data Science Research Grant