Hutch++

Optimal Stochastic Trace Estimation

Raphael A. Meyer (New York University)

With Christopher Musco (New York University), Cameron Musco (University of Massachusetts Amherst), and David P. Woodruff (Carnegie Mellon University)
1. Introduction
   - What problems am I solving?
   - Why are these problems interesting?
   - How am I solving them?

2. Trace Estimation (*SOSA 2021*)

3. Trace Monomial Estimation (*Ongoing Research*)
Scientific Computing relies on Numerical Linear Algebra
We spent decades building better algorithms
Scientific Computing relies on Numerical Linear Algebra

We spent decades building better algorithms

We don’t know which algorithms are optimal
  - Krylov Iteration is optimal for top eigenvalue
  - Hutchinson’s Estimator is suboptimal for trace estimation
Scientific Computing relies on Numerical Linear Algebra

We spent decades building better algorithms

We don’t know which algorithms are optimal
  - Krylov Iteration is optimal for top eigenvalue
  - Hutchinson’s Estimator is suboptimal for trace estimation

My goal: Prove the optimality of linear algebra algorithms
  - Emphasis on building lower bounds
Goal: Estimate trace of $d \times d$ matrix $A$:

$$\text{tr}(A) = \sum_{i=1}^{d} A_{ii} = \sum_{i=1}^{d} \lambda_i$$
Trace Estimation

- Goal: Estimate trace of $d \times d$ matrix $A$:

$$\text{tr}(A) = \sum_{i=1}^{d} A_{ii} = \sum_{i=1}^{d} \lambda_i$$

- In Downstream Applications, $A$ is not stored in memory.
- Instead, $B$ is in memory and $A = f(B)$:

<table>
<thead>
<tr>
<th>No. Triangles</th>
<th>Estrada Index</th>
<th>Log-Determinant</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{tr}(\frac{1}{6} B^3)$</td>
<td>$\text{tr}(e^B)$</td>
<td>$\text{tr}(\ln(B))$</td>
</tr>
</tbody>
</table>
Goal: Estimate trace of $d \times d$ matrix $A$:

$$tr(A) = \sum_{i=1}^{d} A_{ii} = \sum_{i=1}^{d} \lambda_i$$

In Downstream Applications, $A$ is not stored in memory.
Instead, $B$ is in memory and $A = f(B)$:

<table>
<thead>
<tr>
<th>No. Triangles</th>
<th>Estrada Index</th>
<th>Log-Determinant</th>
</tr>
</thead>
<tbody>
<tr>
<td>$tr(\frac{1}{6} B^3)$</td>
<td>$tr(e^B)$</td>
<td>$tr(ln(B))$</td>
</tr>
</tbody>
</table>

- Computing $A = \frac{1}{6} B^3$ takes $O(n^3)$ time
- Computing $Ax = \frac{1}{6} B(B(Bx))$ takes $O(n^2)$ time
- If $A = f(B)$, then we can often compute $Ax$ quickly
Trace Estimation

- Goal: Estimate trace of $d \times d$ matrix $A$:
  \[
  \text{tr}(A) = \sum_{i=1}^{d} A_{ii} = \sum_{i=1}^{d} \lambda_i
  \]

- In Downstream Applications, $A$ is not stored in memory.
- Instead, $B$ is in memory and $A = f(B)$:

<table>
<thead>
<tr>
<th>No. Triangles</th>
<th>Estrada Index</th>
<th>Log-Determinant</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\text{tr}(\frac{1}{6} B^3)$</td>
<td>$\text{tr}(e^B)$</td>
<td>$\text{tr}(\ln(B))$</td>
</tr>
</tbody>
</table>

- Computing $A = \frac{1}{6} B^3$ takes $O(n^3)$ time
- Computing $Ax = \frac{1}{6} B(B(Bx))$ takes $O(n^2)$ time
- If $A = f(B)$, then we can often compute $Ax$ quickly
- Goal: Estimate $\text{tr}(A)$ by computing $Ax_1, \ldots, Ax_k$
Matrix-Vector Oracle Model

Formally: Matrix-Vector Product as a Computational Primitive

e.g. Krylov Methods, Sketching, Streaming, . . .

Very few existing lower bounds

\[ \text{Trace Estimation: Estimate } \text{tr}(A) \text{ with as few Matrix-Vector products } A \cdot x_1, \ldots, A \cdot x_k \text{ as possible.} \]

\[ |\tilde{\text{tr}}(A) - \text{tr}(A)| \leq \epsilon \text{ tr}(A) \]
Matrix-Vector Oracle Model

Formally: Matrix-Vector Product as a Computational Primitive

- Given access to a $d \times d$ matrix $A$ only through a Matrix-Vector Multiplication Oracle

\[
\begin{align*}
\text{x} & \quad \text{input} & \quad \text{ORACLE} & \quad \text{output} \\
& & \quad A\text{x}
\end{align*}
\]

- e.g. Krylov Methods, Sketching, Streaming, ...
- Very few existing lower bounds

\[|\tilde{\text{tr}}(A) - \text{tr}(A)| \leq \varepsilon \text{tr}(A)\]
Matrix-Vector Oracle Model

Formally: Matrix-Vector Product as a Computational Primitive

- Given access to a $d \times d$ matrix $A$ only through a Matrix-Vector Multiplication Oracle
  
  \[
  \begin{align*}
  x & \xrightarrow{\text{input}} \text{ORACLE} \xrightarrow{\text{output}} Ax
  \end{align*}
  \]

- e.g. Krylov Methods, Sketching, Streaming, …
- Very few existing lower bounds

**Trace Estimation:** Estimate $\text{tr}(A)$ with as few Matrix-Vector products $Ax_1, \ldots, Ax_k$ as possible.

\[
|\tilde{\text{tr}}(A) - \text{tr}(A)| \leq \varepsilon \text{tr}(A)
\]
Our Contributions

Prior Work:

- Hutchinson’s Estimator: $O\left(\frac{1}{\varepsilon^2}\right)$ products suffice [AT11]
  - 2 Lines of MATLAB code
- Lower Bound: Hutchinson’s Estimator needs $\Omega\left(\frac{1}{\varepsilon^2}\right)$ products [WWZ14]

Our Results:

- Hutch++ Estimator: $O\left(\frac{1}{\varepsilon}\right)$ products suffice
  - 5 Lines of MATLAB code
- Lower Bound: Any estimator needs $\Omega\left(\frac{1}{\varepsilon}\right)$ products
Symmetric $A \in \mathbb{R}^{d \times d}$ has $A = U \Lambda U^T$

- $U$ is a rotation matrix: $U^T U = I$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
Symmetric $A \in \mathbb{R}^{d \times d}$ has $A = U \Lambda U^\top$

- $U$ is a rotation matrix: $U^\top U = I$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
- $\|A\|_F^2 = \sum_{i,j} A_{i,j}^2 = \sum_i \lambda_i^2$
- $\text{tr}(A) = \sum_i A_{i,i} = \sum_i \lambda_i$
Symmetric $A \in \mathbb{R}^{d \times d}$ has $A = U \Lambda U^T$

- $U$ is a rotation matrix: $U^T U = I$

- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$

- $\|A\|_F^2 = \sum_{i,j} A_{i,j}^2 = \sum_i \lambda_i^2$

- $\text{tr}(A) = \sum_i A_{i,i} = \sum_i \lambda_i$

- Positive Semi-Definite (PSD) $A$ has $\lambda_i \geq 0$ for all $i$
  - $\|A\|_F = \|\lambda\|_2 \leq \|\lambda\|_1 = \text{tr}(A)$
Symmetric $\mathbf{A} \in \mathbb{R}^{d \times d}$ has $\mathbf{A} = \mathbf{U} \Lambda \mathbf{U}^\top$

- $\mathbf{U}$ is a rotation matrix: $\mathbf{U}^\top \mathbf{U} = \mathbf{I}$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
- $\| \mathbf{A} \|_F^2 = \sum_{i,j} \mathbf{A}_{i,j}^2 = \sum_i \lambda_i^2$
- $\text{tr}(\mathbf{A}) = \sum_i \mathbf{A}_{i,i} = \sum_i \lambda_i$

- Positive Semi-Definite (PSD) $\mathbf{A}$ has $\lambda_i \geq 0$ for all $i$
  - $\| \mathbf{A} \|_F = \| \lambda \|_2 \leq \| \lambda \|_1 = \text{tr}(\mathbf{A})$

- Low Rank Approximation:
  $$\mathbf{A}_k = \mathbf{U}_k \Lambda_k \mathbf{U}_k^\top = \arg\min_{\| \mathbf{B} \|_F = k} \| \mathbf{A} - \mathbf{B} \|_F$$
If $\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$, then $\mathbf{A}\mathbf{x} \sim \mathcal{N}(0, \mathbf{AA}^\top)$

If $X_1, \ldots, X_n \sim \mathcal{N}(0, 1)$, then $S := \sum_i X_i^2 \sim \chi_n^2$, $\mathbb{E}[S] = n$, $\text{Var}[S] = 2n$
Probability Review

- If $x \sim \mathcal{N}(0, I)$, then $Ax \sim \mathcal{N}(0, AA^T)$
- If $X_1, \ldots, X_n \sim \mathcal{N}(0, 1)$, then $S := \sum_i X_i^2 \sim \chi^2_n$, $\mathbb{E}[S] = n$, $\text{Var}[S] = 2n$
- Chebyshev’s Ineq: $|X - \mathbb{E}[X]| \leq \frac{1}{\sqrt{\delta}} \sqrt{\text{Var}[X]}$ w.p. $\geq 1 - \delta$
If \( x \sim \mathcal{N}(0, I) \), then \( Ax \sim \mathcal{N}(0, AA^T) \)

If \( X_1, \ldots, X_n \sim \mathcal{N}(0, 1) \), then \( S := \sum_i X_i^2 \sim \chi^2_n \), \( \mathbb{E}[S] = n \), \( \operatorname{Var}[S] = 2n \)

Chebyshev’s Ineq: \( |X - \mathbb{E}[X]| \leq \frac{1}{\sqrt{\delta}} \sqrt{\operatorname{Var}[X]} \) w.p. \( \geq 1 - \delta \)

Chebyshev’s Ineq: \( |X - \mathbb{E}[X]| \leq O(\sqrt{\operatorname{Var}[X]}) \) w.p. \( \geq \frac{2}{3} \)
Towards Optimal

Trace Estimation in the

Matrix-Vector Oracle Model
Hutchinson’s Estimator

If \( x \sim \mathcal{N}(0, I) \), then

\[
\mathbb{E}[x^T Ax] = \text{tr}(A) \quad \text{Var}[x^T Ax] = 2\|A\|_F^2
\]
If $x \sim \mathcal{N}(0, I)$, then
\[
\mathbb{E}[x^T A x] = \text{tr}(A) \quad \text{Var}[x^T A x] = 2 \| A \|^2_F
\]

Hutchinson’s Estimator: $H_\ell(A) := \frac{1}{\ell} \sum_{i=1}^{\ell} x_i^T A x_i$
\[
\mathbb{E}[H_\ell(A)] = \text{tr}(A) \quad \text{Var}[H_\ell(A)] = \frac{2}{\ell} \| A \|^2_F
\]
Hutchinson’s Estimator

- If \( \mathbf{x} \sim \mathcal{N}(0, \mathbf{I}) \), then
  \[
  \mathbb{E}[\mathbf{x}^\top \mathbf{A} \mathbf{x}] = \text{tr}(\mathbf{A}) \quad \text{Var}[\mathbf{x}^\top \mathbf{A} \mathbf{x}] = 2\|\mathbf{A}\|_F^2
  \]

- Hutchinson’s Estimator: \( H_\ell(\mathbf{A}) := \frac{1}{\ell} \sum_{i=1}^{\ell} \mathbf{x}_i^\top \mathbf{A} \mathbf{x}_i \)
  \[
  \mathbb{E}[H_\ell(\mathbf{A})] = \text{tr}(\mathbf{A}) \quad \text{Var}[H_\ell(\mathbf{A})] = \frac{2}{\ell} \|\mathbf{A}\|_F^2
  \]

**Proof:** \( H_\ell(\mathbf{A}) \) needs \( \ell = O\left(\frac{1}{\varepsilon^2}\right) \) for PSD \( \mathbf{A} \)

- For PSD \( \mathbf{A} \), we have \( \|\mathbf{A}\|_F \leq \text{tr}(\mathbf{A}) \), so that
If \( x \sim \mathcal{N}(0, I) \), then
\[
\mathbb{E}[x^T A x] = \text{tr}(A) \quad \text{Var}[x^T A x] = 2\|A\|^2_F
\]

Hutchinson’s Estimator: \( H_\ell(A) := \frac{1}{\ell} \sum_{i=1}^{\ell} x_i^T A x_i \)
\[
\mathbb{E}[H_\ell(A)] = \text{tr}(A) \quad \text{Var}[H_\ell(A)] = \frac{2}{\ell} \|A\|^2_F
\]

Proof: \( H_\ell(A) \) needs \( \ell = O\left(\frac{1}{\varepsilon^2}\right) \) for PSD \( A \)

For PSD \( A \), we have \( \|A\|_F \leq \text{tr}(A) \), so that
\[
|H_\ell(A) - \text{tr}(A)| \leq O\left(\frac{1}{\sqrt{\ell}}\right) \|A\|_F \quad \text{(Chebyshev Ineq.)}
\]
\[
\leq O\left(\frac{1}{\sqrt{\ell}}\right) \text{tr}(A) \quad \text{(} \|A\|_F \leq \text{tr}(A)\text{)}
\]
\[
= \varepsilon \text{tr}(A) \quad \text{(} \ell = O\left(\frac{1}{\varepsilon^2}\right)\text{)}
\]
Hutchinson’s Estimator

For what $A$ is this analysis tight?

$$|H_ℓ(A) - \text{tr}(A)| \leq O\left(\frac{1}{\sqrt{ℓ}}\right)\|A\|_F$$

$$\leq O\left(\frac{1}{\sqrt{ℓ}}\right)\text{tr}(A)$$

$$= \varepsilon \text{tr}(A)$$
Hutchinson’s Estimator

For what $A$ is this analysis tight?

\[ |H_\ell(A) - \text{tr}(A)| \approx O\left( \frac{1}{\sqrt{\ell}} \right) \|A\|_F \]
\[ \leq O\left( \frac{1}{\sqrt{\ell}} \right) \text{tr}(A) \]
\[ = \varepsilon \text{tr}(A) \]

When is the bound $\|A\|_F \leq \text{tr}(A)$ tight?

Let $v = [\lambda_1 \ldots \lambda_n]$ be the eigenvalues of PSD $A$.

When is the bound $\|v\|_2 \leq \|v\|_1$ tight?

Property of norms: $\|v\|_2 \approx \|v\|_1$ only if $v$ is nearly sparse.

Hutchinson only requires $O\left( \frac{1}{\varepsilon^2} \right)$ queries if $A$ has a few large eigenvalues.
For what $A$ is this analysis tight?

$$|H_\ell(A) - \text{tr}(A)| \approx O\left(\frac{1}{\sqrt{\ell}}\right)\|A\|_F$$

$$\leq O\left(\frac{1}{\sqrt{\ell}}\right) \text{tr}(A)$$

$$= \varepsilon \text{tr}(A)$$

When is the bound $\|A\|_F \leq \text{tr}(A)$ tight?
Hutchinson’s Estimator

For what $A$ is this analysis tight?

$$|H_\ell(A) - \text{tr}(A)| \approx O\left(\frac{1}{\sqrt{\ell}}\right)\|A\|_F$$

$$\leq O\left(\frac{1}{\sqrt{\ell}}\right) \text{tr}(A)$$

$$= \varepsilon \text{tr}(A)$$

- When is the bound $\|A\|_F \leq \text{tr}(A)$ tight?
- Let $v = \begin{bmatrix} \lambda_1 & \ldots & \lambda_n \end{bmatrix}$ be the eigenvalues of PSD $A$
Hutchinson’s Estimator

For what \( A \) is this analysis tight?

\[
\left| H_\ell(A) - \text{tr}(A) \right| \approx O\left( \frac{1}{\sqrt{\ell}} \right) \| A \|_F \\
\leq O\left( \frac{1}{\sqrt{\ell}} \right) \text{tr}(A) \\
= \varepsilon \text{tr}(A)
\]

- When is the bound \( \| A \|_F \leq \text{tr}(A) \) tight?
- Let \( v = \begin{bmatrix} \lambda_1 & \ldots & \lambda_n \end{bmatrix} \) be the eigenvalues of PSD \( A \)
- When is the bound \( \| v \|_2 \leq \| v \|_1 \) tight?
Hutchinson’s Estimator

For what $A$ is this analysis tight?

$$|H_\ell(A) - \text{tr}(A)| \approx O\left(\frac{1}{\sqrt{\ell}}\right)\|A\|_F$$

$$\leq O\left(\frac{1}{\sqrt{\ell}}\right)\text{tr}(A)$$

$$= \varepsilon \text{tr}(A)$$

- When is the bound $\|A\|_F \leq \text{tr}(A)$ tight?
- Let $v = [\lambda_1 \ldots \lambda_n]$ be the eigenvalues of PSD $A$
- When is the bound $\|v\|_2 \leq \|v\|_1$ tight?
  - Property of norms: $\|v\|_2 \approx \|v\|_1$ only if $v$ is nearly sparse
For what $A$ is this analysis tight?

\[ |H_\ell(A) - \text{tr}(A)| \approx O\left( \frac{1}{\sqrt{\ell}} \right) \|A\|_F \]
\[ \leq O\left( \frac{1}{\sqrt{\ell}} \right) \text{tr}(A) \]
\[ = \varepsilon \text{tr}(A) \]

- When is the bound $\|A\|_F \leq \text{tr}(A)$ tight?
- Let $v = [\lambda_1 \ldots \lambda_n]$ be the eigenvalues of PSD $A$
- When is the bound $\|v\|_2 \leq \|v\|_1$ tight?
  - Property of norms: $\|v\|_2 \approx \|v\|_1$ only if $v$ is nearly sparse
- Hutchinson only requires $O\left( \frac{1}{\varepsilon^2} \right)$ queries if $A$ has a few large eigenvalues
Helping Hutchinson’s Estimator

Idea: Explicitly estimate the top few eigenvalues of $A$. Use Hutchinson’s for the rest.

The diagram shows the distribution of eigenvalues, with the top few (head) computed directly and the rest (tail) approximated using Hutchinson’s method.
Idea: Explicitly estimate the top few eigenvalues of $A$. Use Hutchinson’s for the rest.

1. Find a good rank-$k$ approximation $\tilde{A}_k$
2. Notice that $\text{tr}(A) = \text{tr}(\tilde{A}_k) + \text{tr}(A - \tilde{A}_k)$
3. Compute $\text{tr}(\tilde{A}_k)$ exactly
4. Return $\text{Hutch}^{++}(A) = \text{tr}(\tilde{A}_k) + H_\ell(A - \tilde{A}_k)$
Idea: Explicitly estimate the top few eigenvalues of $\mathbf{A}$. Use Hutchinson’s for the rest.

1. Find a good rank-$k$ approximation $\tilde{\mathbf{A}}_k$
2. Notice that $\text{tr} (\mathbf{A}) = \text{tr} (\tilde{\mathbf{A}}_k) + \text{tr} (\mathbf{A} - \tilde{\mathbf{A}}_k)$
3. Compute $\text{tr} (\tilde{\mathbf{A}}_k)$ exactly
4. Return $\text{Hutch}^{++}(\mathbf{A}) = \text{tr} (\tilde{\mathbf{A}}_k) + H_\ell (\mathbf{A} - \tilde{\mathbf{A}}_k)$

If $k = \ell = O(\frac{1}{\varepsilon})$, then $|\text{Hutch}^{++}(\mathbf{A}) - \text{tr}(\mathbf{A})| \leq \varepsilon \text{tr}(\mathbf{A})$. (Whiteboard)
Let $A_k$ be the best rank-$k$ approximation of $A$.

**Lemma [Sar06, Woo14]**

Let $S \in \mathbb{R}^{d \times k}$ have i.i.d. uniform ±1 entries, $Q = \text{orth}(AS)$, and $\tilde{A}_k = AQQ^T$. Then, with probability $1 - \delta$,

$$\|A - \tilde{A}_k\|_F \leq 2\|A - A_k\|_F$$

so long as $S$ has $m = O(k + \log(1/\delta))$ columns.
Let \( A_k \) be the best rank-\( k \) approximation of \( A \).

**Lemma [Sar06, Woo14]**

Let \( S \in \mathbb{R}^{d \times k} \) have i.i.d. uniform \( \pm 1 \) entries, \( Q = \text{orth}(AS) \), and \( \tilde{A}_k = AQ Q^T \). Then, with probability \( 1 - \delta \),

\[
\|A - \tilde{A}_k\|_F \leq 2\|A - A_k\|_F
\]

so long as \( S \) has \( m = O(k + \log(1/\delta)) \) columns.

We can compute the trace of \( \tilde{A}_k \) with \( m \) queries and \( O(mn) \) space:

\[
\text{tr}(\tilde{A}_k) = \text{tr}(AQ Q^T) = \text{tr}(Q^T(AQ))
\]
Hutch++ Algorithm:

- **Input:** Number of matrix-vector queries $m$, matrix $A$

1. Sample $S \in \mathbb{R}^{d \times \frac{m}{3}}$ and $G \in \mathbb{R}^{d \times \frac{m}{3}}$ with i.i.d. $\mathcal{N}(0, I)$ entries
2. Compute $Q = \text{qr}(AS)$
3. Return $\text{tr}(Q^T AQ) + \frac{3}{m} \text{tr}(G^T(I - QQ^T)A(I - QQ^T)G)$
Hutch++ Algorithm:

- **Input:** Number of matrix-vector queries $m$, matrix $A$
  1. Sample $S \in \mathbb{R}^{d \times \frac{m}{3}}$ and $G \in \mathbb{R}^{d \times \frac{m}{3}}$ with i.i.d. $\mathcal{N}(0, I)$ entries
  2. Compute $Q = \text{qr}(AS)$
  3. Return $\text{tr}(Q^T AQ) + \frac{3}{m} \text{tr}(G^T (I - QQ^T) A (I - QQ^T) G)$

This algorithm is **adaptive**:

\[ x_{k+1} \xrightarrow{} \text{ORACLE} \xrightarrow{} Ax_k \]

\[ \text{ALGORITHM} \]
Hutch++ Algorithm:

- Input: Number of matrix-vector queries $m$, matrix $A$

1. Sample $S \in \mathbb{R}^{d \times \frac{m}{3}}$ and $G \in \mathbb{R}^{d \times \frac{m}{3}}$ with i.i.d. $\mathcal{N}(0, I)$ entries
2. Compute $Q = \text{qr}(AS)$
3. Return $\text{tr}(Q^T AQ) + \frac{3}{m} \text{tr}(G^T(I - QQ^T)A(I - QQ^T)G)$

This algorithm is **adaptive**:

There is a **non-adaptive** variant of Hutch++:

$$\{x_1, \ldots, x_m\} \rightarrow \text{ORACLE} \rightarrow \{Ax_1, \ldots, Ax_m\}$$

$$\uparrow \hspace{1cm} \downarrow$$

$$\text{ALGORITHM} \hspace{1cm} \text{ALGORITHM}$$
Experiments

When \( \| A \|_F \approx \text{tr}(A) \), Hutch++ is much faster than \( H_\ell \):

\[ \| A \|_F = 0.63 \text{tr}(A) \]  
\[ \| A \|_F = 0.02 \text{tr}(A) \]

function T = hutchplussplus(A, m)
    S = 2*randi(2,size(A,1),m/3);
    G = 2*randi(2,size(A,1),m/3);
    [Q,~] = qr(A*S,0);
    G = G - Q*(Q'*G);
    T = trace(Q'*A*Q) + 1/size(G,2)*trace(G'*A*G);
end
Trace Estimation Lower Bounds
Super Rough Intuition

View oracle as a **limit on information** about $A$:

1. Suppose $A \sim \mathcal{D}$ is a random matrix
2. Then $\text{tr}(A)$ is a random variable with variance
3. If an algorithm computes few queries, it has little information about $\text{tr}(A)$
4. Then the algorithm cannot predict $\text{tr}(A)$ well
Super Rough Intuition

\[
x \xrightarrow{\text{input}} \text{ORACLE} \xrightarrow{\text{output}} Ax
\]

View oracle as a limit on information about \( A \):

1. Suppose \( A \sim \mathcal{D} \) is a random matrix
2. Then \( \text{tr}(A) \) is a random variable with variance
3. If an algorithm computes few queries, it has little information about \( \text{tr}(A) \)
4. Then the algorithm cannot predict \( \text{tr}(A) \) well
Removing the Algorithm’s Agency

- **Problem:** The user can pick many different query vectors $x$.
- If the user had no freedom, we could use statistics to make lower bounds.
Problem: The user can pick many different query vectors $x$.

If the user had no freedom, we could use statistics to make lower bounds.

Two Observations:

1. WLOG, the user submits orthonormal query vectors
Problem: The user can pick many different query vectors $\mathbf{x}$.

If the user had no freedom, we could use statistics to make lower bounds.

Two Observations:

1. WLOG, the user submits orthonormal query vectors

2. Let $\mathbf{G}$ be a $\mathcal{N}(0, 1)$ Gaussian matrix
   Let $\mathbf{Q}$ be an orthogonal matrix
   Then $\mathbf{GQ}$ is a $\mathcal{N}(0, 1)$ Gaussian matrix
   (informal) If $\mathbf{A}$ uses Gaussians, the responses from the oracle are independent of the queries submitted.
Removing the Algorithm’s Agency

- **Problem:** The user can pick many different query vectors $\mathbf{x}$. If the user had no freedom, we could use statistics to make lower bounds.

Two Observations:

1. WLOG, the user submits orthonormal query vectors
2. Let $\mathbf{G}$ be a $\mathcal{N}(0, 1)$ Gaussian matrix
   - Let $\mathbf{Q}$ be an orthogonal matrix
   - Then $\mathbf{GQ}$ is a $\mathcal{N}(0, 1)$ Gaussian matrix
     - (informal) If $\mathbf{A}$ uses Gaussians, the responses from the oracle are independent of the queries submitted.

- (informal) WLOG, the user observes the first $k$ columns of $\mathbf{A}$. 
Let $G \in \mathbb{R}^{d \times d}$ be a $\mathcal{N}(0, 1)$ Gaussian Matrix.

Let $A = G^T G$ be a Wishart Matrix.

An algorithm sends query vectors $x_1, \ldots, x_k$, gets responses $w_1, \ldots, w_k$

Analogous holds for Wigner Matrices:

$A = \frac{1}{2} (G + G^T)$

Has been used for T trace, Max Eigenvalue, Linear Systems, SVD Lower Bounds
Wigner/Wishart Anti-Concentration Method

Theorem (Wishart Case)

- Let \( G \in \mathbb{R}^{d \times d} \) be a \( \mathcal{N}(0, 1) \) Gaussian Matrix.
- Let \( A = G^T G \) be a Wishart Matrix.
- An algorithm sends query vectors \( x_1, \ldots, x_k \), gets responses \( w_1, \ldots, w_k \).
- Then there exists orthogonal matrix \( V \) such that
  \[
  VAV^T = \Delta + \begin{bmatrix}
  0 & 0 \\
  0 & \tilde{A}
  \end{bmatrix}
  \]
  where \( \tilde{A} \in \mathbb{R}^{(d-k) \times (d-k)} \) is distributed as \( \tilde{A} = \tilde{G}^T \tilde{G} \), conditioned on all observations \( x_1, \ldots, x_k, w_1, \ldots, w_k \).
- \( \Delta \) is known exactly.
Theorem (Wishart Case)

- Let $G \in \mathbb{R}^{d \times d}$ be a $\mathcal{N}(0, 1)$ Gaussian Matrix.
- Let $A = G^T G$ be a Wishart Matrix.
- An algorithm sends query vectors $x_1, \ldots, x_k$, gets responses $w_1, \ldots, w_k$
- Then there exists orthogonal matrix $V$ such that

$$VAV^T = \Delta + \begin{bmatrix} 0 & 0 \\ 0 & \tilde{A} \end{bmatrix}$$

where $\tilde{A} \in \mathbb{R}^{(d-k) \times (d-k)}$ is distributed as $\tilde{A} = \tilde{G}^T \tilde{G}$, conditioned on all observations $x_1, \ldots, x_k, w_1, \ldots, w_k$
- $\Delta$ is known exactly

- Analogous holds for Wigner Matrices: $A = \frac{1}{2}(G + G^T)$
Consider any adaptive algorithm after $k$ steps:

1. $\text{tr}(A) = \text{tr}(VAV^\top) = \text{tr}(\Delta) + \text{tr}(\tilde{A})$
Consider any adaptive algorithm after $k$ steps:

1. $\text{tr}(A) = \text{tr}(VAV^T) = \text{tr}(\Delta) + \text{tr}(\tilde{A})$
2. Let $t$ estimate $\text{tr}(A)$. Define $\tilde{t} := t - \text{tr}(\Delta)$. 
Consider any adaptive algorithm after $k$ steps:

1. $\text{tr}(\mathbf{A}) = \text{tr}(\mathbf{VAV}^\top) = \text{tr}(\Delta) + \text{tr}(\tilde{\mathbf{A}})$

2. Let $t$ estimate $\text{tr}(\mathbf{A})$. Define $\tilde{t} := t - \text{tr}(\Delta)$.

3. Note $\text{tr}(\mathbf{A}) = \|\mathbf{G}\|_F^2 \sim \chi_{d^2}^2$ and $\text{tr}(\tilde{\mathbf{A}}) \sim \chi_{(d-k)^2}^2$
Consider any adaptive algorithm after $k$ steps:

1. $\text{tr}(A) = \text{tr}(VAV^\top) = \text{tr}(\Delta) + \text{tr}(\tilde{A})$

2. Let $t$ estimate $\text{tr}(A)$. Define $\tilde{t} := t - \text{tr}(\Delta)$.

3. Note $\text{tr}(A) = \|G\|_F^2 \sim \chi^2_{d^2}$ and $\text{tr}(\tilde{A}) \sim \chi^2_{(d-k)^2}$
   - $|t - \text{tr}(A)| = |	ilde{t} - \text{tr}(\tilde{A})| \geq \Omega(d - k)$
Consider any adaptive algorithm after \( k \) steps:

1. \( \text{tr}(A) = \text{tr}(VAV^T) = \text{tr}(\Delta) + \text{tr}(\tilde{A}) \)
2. Let \( t \) estimate \( \text{tr}(A) \). Define \( \tilde{t} := t - \text{tr}(\Delta) \).
3. Note \( \text{tr}(A) = \|G\|_F^2 \sim \chi^2_d \) and \( \text{tr}(\tilde{A}) \sim \chi^2_{(d-k)^2} \)
   - \( |t - \text{tr}(A)| = |\tilde{t} - \text{tr}(\tilde{A})| \geq \Omega(d - k) \)
   - \( \text{tr}(A) \leq O(d^2) \)
Consider any adaptive algorithm after $k$ steps:

1. $\text{tr}(A) = \text{tr}(VAV^T) = \text{tr}(\Delta) + \text{tr}(\tilde{A})$

2. Let $t$ estimate $\text{tr}(A)$. Define $\tilde{t} := t - \text{tr}(\Delta)$.

3. Note $\text{tr}(A) = \|G\|_F^2 \sim \chi^2_{d^2}$ and $\text{tr}(\tilde{A}) \sim \chi^2_{(d-k)^2}$
   - $|t - \text{tr}(A)| = |\tilde{t} - \text{tr}(\tilde{A})| \geq \Omega(d - k)$
   - $\text{tr}(A) \leq O(d^2)$

4. Enforce $|t - \text{tr}(A)| \leq \epsilon \text{tr}(A)$
   
   $(d - k) \leq \epsilon \cdot Cd^2$
Consider any adaptive algorithm after $k$ steps:

1. $\text{tr}(A) = \text{tr}(VAV^\top) = \text{tr}(\Delta) + \text{tr}(\tilde{A})$

2. Let $t$ estimate $\text{tr}(A)$. Define $\tilde{t} := t - \text{tr}(\Delta)$.

3. Note $\text{tr}(A) = \|G\|^2_F \sim \chi^2_{d^2}$ and $\text{tr}(\tilde{A}) \sim \chi^2_{(d-k)^2}$
   - $|t - \text{tr}(A)| = |\tilde{t} - \text{tr}(\tilde{A})| \geq \Omega(d - k)$
   - $\text{tr}(A) \leq O(d^2)$

4. Enforce $|t - \text{tr}(A)| \leq \epsilon \text{tr}(A)$
   
   $(d - k) \leq \epsilon \cdot Cd^2$

5. Set $d = \frac{1}{2C\epsilon}$ and simplify: $k \geq \frac{1}{4C\epsilon}$
### Non-Adaptive Proof Framework

Design distributions $\mathcal{P}_0$ and $\mathcal{P}_1$, for large enough $n$:

<table>
<thead>
<tr>
<th>$\mathcal{P}_0$</th>
<th>$A = G^T G$ for $G \in \mathbb{R}^{\left(\frac{1}{\epsilon}\right) \times d}$ Gaussian</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{P}_1$</td>
<td>$A = G^T G$ for $G \in \mathbb{R}^{\left(\frac{1}{\epsilon}+1\right) \times d}$ Gaussian</td>
</tr>
</tbody>
</table>
Statistical Hypothesis Testing

Non-Adaptive Proof Framework

Design distributions $\mathcal{P}_0$ and $\mathcal{P}_1$, for large enough $n$:

<table>
<thead>
<tr>
<th>$\mathcal{P}_0$</th>
<th>$A = G^T G$ for $G \in \mathbb{R}^{(1/\varepsilon) \times d}$ Gaussian</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{P}_1$</td>
<td>$A = G^T G$ for $G \in \mathbb{R}^{(1/\varepsilon+1) \times d}$ Gaussian</td>
</tr>
</tbody>
</table>

1. A trace estimator can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$
   - If $A_0 \sim \mathcal{P}_0$ and $A_1 \sim \mathcal{P}_1$
     - With high probability, $\text{tr}(A_0) \leq (1 - 2\varepsilon) \text{tr}(A_1)$
Non-Adaptive Proof Framework

Design distributions $\mathcal{P}_0$ and $\mathcal{P}_1$, for large enough $n$:

$$
\begin{align*}
\mathcal{P}_0 & \quad A = G^T G \quad \text{for} \quad G \in \mathbb{R}^{\left(\frac{1}{\varepsilon}\right) \times d} \text{ Gaussian} \\
\mathcal{P}_1 & \quad A = G^T G \quad \text{for} \quad G \in \mathbb{R}^{\left(\frac{1}{\varepsilon}+1\right) \times d} \text{ Gaussian}
\end{align*}
$$

1. A trace estimator can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$
   - If $A_0 \sim \mathcal{P}_0$ and $A_1 \sim \mathcal{P}_1$
   - With high probability, $\text{tr}(A_0) \leq (1 - 2\varepsilon) \text{tr}(A_1)$

2. No algorithm can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$ with $\Omega\left(\frac{1}{\varepsilon}\right)$ queries
   - Nature samples $i \sim \{0, 1\}$, and $A \sim \mathcal{P}_i$
   - User access $A$ through the oracle
Non-Adaptive Proof Framework

Design distributions $\mathcal{P}_0$ and $\mathcal{P}_1$, for large enough $n$:

<table>
<thead>
<tr>
<th>$\mathcal{P}_0$</th>
<th>$A = G^T G$ for $G \in \mathbb{R}^{(1/\epsilon) \times d}$ Gaussian</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\mathcal{P}_1$</td>
<td>$A = G^T G$ for $G \in \mathbb{R}^{(1/\epsilon+1) \times d}$ Gaussian</td>
</tr>
</tbody>
</table>

1. A trace estimator can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$
   - If $A_0 \sim \mathcal{P}_0$ and $A_1 \sim \mathcal{P}_1$
   - With high probability, $\text{tr}(A_0) \leq (1 - 2\epsilon) \text{tr}(A_1)$

2. No algorithm can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$ with $\Omega(1/\epsilon)$ queries
   - Nature samples $i \sim \{0, 1\}$, and $A \sim \mathcal{P}_i$
   - User access $A$ through the oracle
   - WLOG User picks standard basis vectors
Non-Adaptive Proof Framework

Design distributions $\mathcal{P}_0$ and $\mathcal{P}_1$, for large enough $n$:

$$
\begin{align*}
\mathcal{P}_0 & \quad A = G^T G \quad \text{for} \quad G \in \mathbb{R}^{\left(\frac{1}{\varepsilon}\right) \times d}\ \text{Gaussian} \\
\mathcal{P}_1 & \quad A = G^T G \quad \text{for} \quad G \in \mathbb{R}^{\left(\frac{1}{\varepsilon} + 1\right) \times d}\ \text{Gaussian}
\end{align*}
$$

1. A trace estimator can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$
   - If $A_0 \sim \mathcal{P}_0$ and $A_1 \sim \mathcal{P}_1$
   - With high probability, $\text{tr}(A_0) \leq (1 - 2\varepsilon) \text{tr}(A_1)$

2. No algorithm can distinguish $\mathcal{P}_0$ from $\mathcal{P}_1$ with $\Omega\left(\frac{1}{\varepsilon}\right)$ queries
   - Nature samples $i \sim \{0, 1\}$, and $A \sim \mathcal{P}_i$
   - User access $A$ through the oracle
   - WLOG User picks standard basis vectors
   - Bound Total Variation between first $k$ columns of $A_0$ and $A_1$
1. Introduced Hutchinson’s Estimator for PSD $\mathbf{A}$
2. Improved it: Hutch++ uses $O(\frac{1}{\varepsilon})$
3. Two lower bounds: Adaptive & Non-Adaptive require $\Omega(\frac{1}{\varepsilon})$
4. Trace Estimation requires $\Theta(\frac{1}{\varepsilon})$ queries
Open Questions

- When is adaptivity helpful?
- What about inexact oracles? We often approximate $f(A)x$ with iterative methods. How accurate do these computations need to be?
- Extend to include row/column sampling? This would encapsulate e.g. SGD/SCD.
- Memory-limited lower bounds? This is a realistic model for iterative methods.
THANK YOU
Haim Avron and Sivan Toledo.
Randomized algorithms for estimating the trace of an implicit symmetric positive semi-definite matrix.

Tamas Sarlos.
Improved approximation algorithms for large matrices via random projections.

David P. Woodruff.
Sketching as a tool for numerical linear algebra.