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)

Collaborators

Christopher Musco
(NYU)

Cameron Musco
(UMass. Amherst)

David P. Woodruff
(CMU)

Trace Estimation

- Goal: Estimate trace of $n \times n$ matrix $\boldsymbol{A}$:

$$\mathrm{tr}(\boldsymbol{A}) = \sum_{i=1}^{n} \boldsymbol{A}_{ii} = \sum_{i=1}^{n} \lambda_i$$

Trace Estimation

- Goal: Estimate trace of $n \times n$ matrix $\boldsymbol{A}$:

$$\mathrm{tr}(\boldsymbol{A}) = \sum_{i=1}^{n} \boldsymbol{A}_{ii} = \sum_{i=1}^{n} \lambda_i$$

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

No. Triangles	Estrada Index	Log-Determinant
$\mathrm{tr}(\frac{1}{6}\boldsymbol{B}^3)$	$\mathrm{tr}(e^{\boldsymbol{B}})$	$\mathrm{tr}(\ln(\boldsymbol{B}))$

Trace Estimation

- Goal: Estimate trace of $n \times n$ matrix $\boldsymbol{A}$:

$$\mathrm{tr}(\boldsymbol{A}) = \sum_{i=1}^{n} \boldsymbol{A}_{ii} = \sum_{i=1}^{n} \lambda_i$$

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

No. Triangles	Estrada Index	Log-Determinant
$\mathrm{tr}(\frac{1}{6}\boldsymbol{B}^3)$	$\mathrm{tr}(e^{\boldsymbol{B}})$	$\mathrm{tr}(\ln(\boldsymbol{B}))$

- Computing $\boldsymbol{A} = \frac{1}{6}\boldsymbol{B}^3$ takes $O(n^3)$ time, which is too slow
- Computing $\boldsymbol{A}\mathbf{x} = \frac{1}{6}\boldsymbol{B}(\boldsymbol{B}(\boldsymbol{B}\mathbf{x}))$ takes $O(n^2)$ time
- If $\boldsymbol{A} = f(\boldsymbol{B})$, then we can often compute $\boldsymbol{A}\mathbf{x}$ quickly

Matrix-Vector Oracle Model

Idea: Matrix-Vector Product as a Computational Primitive

Matrix-Vector Oracle Model

Idea: Matrix-Vector Product as a Computational Primitive

- Given access to a $n \times n$ matrix $\boldsymbol{A}$ only through a Matrix-Vector Multiplication Oracle

$$\mathbf{x} \quad \xrightarrow{\textit{input}} \quad \text{ORACLE} \quad \xrightarrow{\textit{output}} \quad \boldsymbol{A}\mathbf{x}$$

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

Matrix-Vector Oracle Model

Idea: Matrix-Vector Product as a Computational Primitive

- Given access to a $n \times n$ matrix $\boldsymbol{A}$ only through a Matrix-Vector Multiplication Oracle

$$\mathbf{x} \xRightarrow{\textit{input}} \text{ORACLE} \xRightarrow{\textit{output}} \boldsymbol{A}\mathbf{x}$$

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

Implicit Matrix Trace Estimation: Estimate $\mathrm{tr}(\boldsymbol{A})$ with as few Matrix-Vector products $\boldsymbol{A}\mathbf{x}_1, \ldots, \boldsymbol{A}\mathbf{x}_m$ as possible.

$$(1-\varepsilon)\,\mathrm{tr}(\boldsymbol{A}) \leq \tilde{\mathrm{tr}}(\boldsymbol{A}) \leq (1+\varepsilon)\,\mathrm{tr}(\boldsymbol{A})$$

3 Core Contributions

For PSD matrix trace estimation,

1. Hutch++ algorithm, which uses $\tilde{O}(\frac{1}{\varepsilon})$ matrix-vector products.
    - Improves prior rate of $\tilde{O}(\frac{1}{\varepsilon^2})$
    - Empirically works well

[1] $\tilde{O}$ notation only hide logarithmic dependence on the failure probability.

3 Core Contributions

For PSD matrix trace estimation,

1. Hutch++ algorithm, which uses $\tilde{O}(\frac{1}{\varepsilon})$ matrix-vector products.
   - Improves prior rate of $\tilde{O}(\frac{1}{\varepsilon^2})$
   - Empirically works well
2. All adaptive algorithms with finite-precision oracles use $\Omega(\frac{1}{\varepsilon \log(1/\varepsilon)})$ queries.
3. All nonadaptive algorithms with infinite-precision oracles use $\Omega(\frac{1}{\varepsilon})$ queries.

[1] $\tilde{O}$ notation only hide logarithmic dependence on the failure probability.

Hutchinson's Estimator

The classical approach to trace estimation:

Hutchinson 1991, Girard 1987

1. Draw $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^n$ with i.i.d. uniform $\{+1,\text{-}1\}$ entries
2. Return $\tilde{T} = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_i^\intercal \boldsymbol{A} \mathbf{x}_i$

Avron, Toledo 2011, Roosta, Ascher 2015

With probability $1 - \delta$,

$$|\tilde{T} - \operatorname{tr}(\boldsymbol{A})| \leq \tilde{O}\left(\frac{1}{\sqrt{m}}\right) \|\boldsymbol{A}\|_F$$

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \text{tr}(\boldsymbol{A})$, so that:

$$|\tilde{T} - \text{tr}(\boldsymbol{A})| \leq O(\frac{1}{\sqrt{m}})\|\boldsymbol{A}\|_F$$

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$\begin{aligned} |\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\leq O(\tfrac{1}{\sqrt{m}})\|\boldsymbol{A}\|_F \\ &\leq O(\tfrac{1}{\sqrt{m}})\operatorname{tr}(\boldsymbol{A}) \end{aligned}$$

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$\begin{aligned} |\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\leq O(\tfrac{1}{\sqrt{m}})\|\boldsymbol{A}\|_F \\ &\leq O(\tfrac{1}{\sqrt{m}}) \operatorname{tr}(\boldsymbol{A}) \\ &= \varepsilon \operatorname{tr}(\boldsymbol{A}) \end{aligned}$$

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$\begin{aligned} |\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\leq O\left(\tfrac{1}{\sqrt{m}}\right)\|\boldsymbol{A}\|_F \\ &\leq O\left(\tfrac{1}{\sqrt{m}}\right)\operatorname{tr}(\boldsymbol{A}) \\ &= \varepsilon \operatorname{tr}(\boldsymbol{A}) \end{aligned}$$

- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$
\begin{aligned}
|\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\approx O\left(\frac{1}{\sqrt{m}}\right)\|\boldsymbol{A}\|_F \\
&\leq O\left(\frac{1}{\sqrt{m}}\right)\operatorname{tr}(\boldsymbol{A}) \\
&= \varepsilon \operatorname{tr}(\boldsymbol{A})
\end{aligned}
$$

- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$
\begin{aligned}
|\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{m}})\|\boldsymbol{A}\|_F \\
&\leq O(\tfrac{1}{\sqrt{m}}) \operatorname{tr}(\boldsymbol{A}) \\
&= \varepsilon \operatorname{tr}(\boldsymbol{A})
\end{aligned}
$$

- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?
- When is the bound $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(A)$ tight?

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$
\begin{aligned}
|\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\approx O\left(\tfrac{1}{\sqrt{m}}\right)\|\boldsymbol{A}\|_F \\
&\leq O\left(\tfrac{1}{\sqrt{m}}\right)\operatorname{tr}(\boldsymbol{A}) \\
&= \varepsilon \operatorname{tr}(\boldsymbol{A})
\end{aligned}
$$

- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?
- When is the bound $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(A)$ tight?
- Let $\mathbf{v} = \begin{bmatrix} \lambda_1 & \dots & \lambda_n \end{bmatrix}$ be the eigenvalues of PSD $\boldsymbol{A}$

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \mathrm{tr}(\boldsymbol{A})$, so that:

$$\begin{aligned} |\tilde{T} - \mathrm{tr}(\boldsymbol{A})| &\approx O\left(\tfrac{1}{\sqrt{m}}\right)\|\boldsymbol{A}\|_F \\ &\leq O\left(\tfrac{1}{\sqrt{m}}\right)\mathrm{tr}(\boldsymbol{A}) \\ &= \varepsilon\, \mathrm{tr}(\boldsymbol{A}) \end{aligned}$$

- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?
- When is the bound $\|\boldsymbol{A}\|_F \leq \mathrm{tr}(A)$ tight?
- Let $\mathbf{v} = \begin{bmatrix} \lambda_1 & \dots & \lambda_n \end{bmatrix}$ be the eigenvalues of PSD $\boldsymbol{A}$
- When is the bound $\|\mathbf{v}\|_2 \leq \|\mathbf{v}\|_1$ tight?

Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \text{tr}(\boldsymbol{A})$, so that:

$$\begin{aligned} |\tilde{T} - \text{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{m}})\|\boldsymbol{A}\|_F \\ &\leq O(\tfrac{1}{\sqrt{m}})\,\text{tr}(\boldsymbol{A}) \\ &= \varepsilon\,\text{tr}(\boldsymbol{A}) \end{aligned}$$

- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?
- When is the bound $\|\boldsymbol{A}\|_F \leq \text{tr}(A)$ tight?
- Let $\mathbf{v} = \begin{bmatrix} \lambda_1 & \dots & \lambda_n \end{bmatrix}$ be the eigenvalues of PSD $\boldsymbol{A}$
- When is the bound $\|\mathbf{v}\|_2 \leq \|\mathbf{v}\|_1$ tight?
  - Property of norms: $\|\mathbf{v}\|_2 \approx \|\mathbf{v}\|_1$ only if $\mathbf{v}$ is nearly sparse

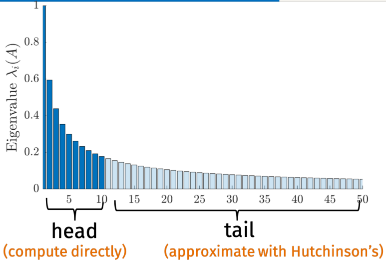
Hutchinson Analysis

For PSD $\boldsymbol{A}$, $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$, so that:

$$
\begin{aligned}
|\tilde{T} - \operatorname{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{m}})\|\boldsymbol{A}\|_F \\
&\leq O(\tfrac{1}{\sqrt{m}})\operatorname{tr}(\boldsymbol{A}) \\
&= \varepsilon \operatorname{tr}(\boldsymbol{A})
\end{aligned}
$$

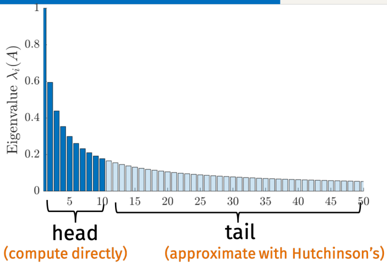
- When does Hutchinson's Estimator truly need $O(\frac{1}{\varepsilon^2})$ queries?
- When is the bound $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(A)$ tight?
- Let $\mathbf{v} = \begin{bmatrix} \lambda_1 & \dots & \lambda_n \end{bmatrix}$ be the eigenvalues of PSD $\boldsymbol{A}$
- When is the bound $\|\mathbf{v}\|_2 \leq \|\mathbf{v}\|_1$ tight?
  - Property of norms: $\|\mathbf{v}\|_2 \approx \|\mathbf{v}\|_1$ only if $\mathbf{v}$ is nearly sparse
- Hutchinson only requires $O(\frac{1}{\varepsilon^2})$ queries if $\boldsymbol{A}$ has a few large eigenvalues

Helping Hutchinson's Estimator



Idea: Explicitly estimate the top few eigenvalues of $\boldsymbol{A}$. Use Hutchinson's for the rest.

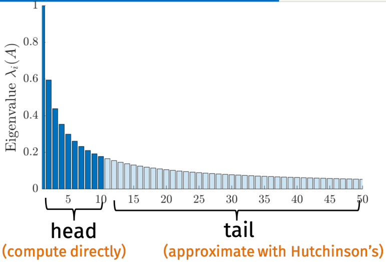
Helping Hutchinson's Estimator



Idea: Explicitly estimate the top few eigenvalues of $\boldsymbol{A}$. Use Hutchinson's for the rest.

1. Find a good rank-k approximation $\tilde{\boldsymbol{A}}_k$
2. Notice that $\mathrm{tr}(\boldsymbol{A}) = \mathrm{tr}(\tilde{\boldsymbol{A}}_k) + \mathrm{tr}(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$
3. Compute $\mathrm{tr}(\tilde{\boldsymbol{A}}_k)$ exactly
4. Compute $\tilde{T} \approx \mathrm{tr}(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$ with m steps of Hutchinson's
5. Return Hutch++$(\boldsymbol{A}) = \mathrm{tr}(\tilde{\boldsymbol{A}}_k) + \tilde{T}$

Helping Hutchinson's Estimator



Idea: Explicitly estimate the top few eigenvalues of $\boldsymbol{A}$. Use Hutchinson's for the rest.

1. Find a good rank-k approximation $\tilde{\boldsymbol{A}}_k$
2. Notice that $\text{tr}(\boldsymbol{A}) = \text{tr}(\tilde{\boldsymbol{A}}_k) + \text{tr}(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$
3. Compute $\text{tr}(\tilde{\boldsymbol{A}}_k)$ exactly
4. Compute $\tilde{T} \approx \text{tr}(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$ with m steps of Hutchinson's
5. Return Hutch++$(\boldsymbol{A}) = \text{tr}(\tilde{\boldsymbol{A}}_k) + \tilde{T}$

- Lemma: $\|\boldsymbol{A} - \tilde{\boldsymbol{A}}_k\|_F \leq \frac{2}{\sqrt{k}} \operatorname{tr}(\boldsymbol{A})$
  - Replaces earlier bound $\|\boldsymbol{A}\|_F \leq \operatorname{tr}(\boldsymbol{A})$
  - For all $\mathbf{v}$, there exists k-sparse $\tilde{\mathbf{v}}$ such that

$$\|\mathbf{v} - \tilde{\mathbf{v}}\|_2 \leq \frac{1}{\sqrt{k}} \|\mathbf{v}\|_1$$

- Final Theorem:
  - Using rank-k approximation and m samples in Hutchinson's
  - $|\operatorname{tr}(\boldsymbol{A}) - \text{Hutch++}(\boldsymbol{A})| \leq O(\frac{1}{\sqrt{km}}) \operatorname{tr}(\boldsymbol{A})$
  - Set $k = m = \tilde{O}(\frac{1}{\varepsilon})$

Implimentation

- Input: Number of matrix-vector queries m

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

```
function T = hutchplusplus(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
```

If you want to learn more

25 Minute Version of this Talk: More Details

- Full proof of Hutch++ Correctness
- Intuitions for both lower bounds
- Discussion of some experiments

In the full paper: Even more details

- Non-Adaptive Algorithm
- Minor Optimizations
- Full Proofs
- Richer discussion of experiments

Code: github.com/RaphaelArkadyMeyerNYU/hutchplusplus

Open Questions

- **In progress:** Lower bounds for e.g. $\mathrm{tr}(\boldsymbol{A}^3)$, $\mathrm{tr}(e^{\boldsymbol{A}})$, $\mathrm{tr}(\boldsymbol{A}^{-1})$
- What about inexact oracles? We often approximate $f(\boldsymbol{A})\mathbf{x}$ with iterative methods. How accurate do these computations need to be?
- Extend to include row/column sampling? This would encapsulate e.g. SGD/SCD.

THANK YOU

Code available at

github.com/RaphaelArkadyMeyerNYU/hutchplusplus