Hutch++: Optimal Stochastic Trace Estimation

Raphael A. Meyer[1] Cameron Musco[2] Christopher Musco[1] David P. Woodruff[3]

[1]New York University [2]University of Massachusetts Amherst [3]Carnegie Mellon University

Introduction and Overview

Trace Estimation

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
- 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

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

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

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

Implicit Matrix Trace Estimation:

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

$$\left|\tilde{\mathrm{tr}}(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})\right| \le \varepsilon\, \mathrm{tr}(\boldsymbol{A})$$

Our Contributions

For PSD $\boldsymbol{A}$, we show $\Theta(\frac{1}{\varepsilon})$ products are necessary and sufficient. Prior work used $O(\frac{1}{\varepsilon^2})$ products.

Hutch++ Algorithm:

- Input: Number of matrix-vector queries m, matrix $\boldsymbol{A}$

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 $\boldsymbol{Q} = \mathrm{qr}(\boldsymbol{A}\boldsymbol{S})$
3. Return $\mathrm{tr}(\boldsymbol{Q}^T\boldsymbol{A}\boldsymbol{Q}) + \frac{3}{m}\mathrm{tr}(\boldsymbol{G}^T(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\intercal)\boldsymbol{A}(\boldsymbol{I} - \boldsymbol{Q}\boldsymbol{Q}^\intercal)\boldsymbol{G})$

Main Theorem and Context

Prior Work: Hutchinson's Estimator

Hutchinson's Estimator:

- If $\mathbf{x} \sim \mathcal{N}(0, \boldsymbol{I})$, then

$$\mathbb{E}[\mathbf{x}^\intercal \boldsymbol{A}\mathbf{x}] = \mathrm{tr}(\boldsymbol{A}) \qquad \mathrm{Var}[\mathbf{x}^\intercal \boldsymbol{A}\mathbf{x}] = \|\boldsymbol{A}\|_F^2$$

- Hutchinson's Estimator: $\mathrm{H}_\ell(\boldsymbol{A}) := \frac{1}{\ell}\sum_{i=1}^{\ell} \mathbf{x}_i^\intercal \boldsymbol{A}\mathbf{x}_i$

$$\mathbb{E}[\mathrm{H}_\ell(\boldsymbol{A})] = \mathrm{tr}(\boldsymbol{A}) \qquad \mathrm{Var}[\mathrm{H}_\ell(\boldsymbol{A})] = \tfrac{1}{\ell}\|\boldsymbol{A}\|_F^2$$

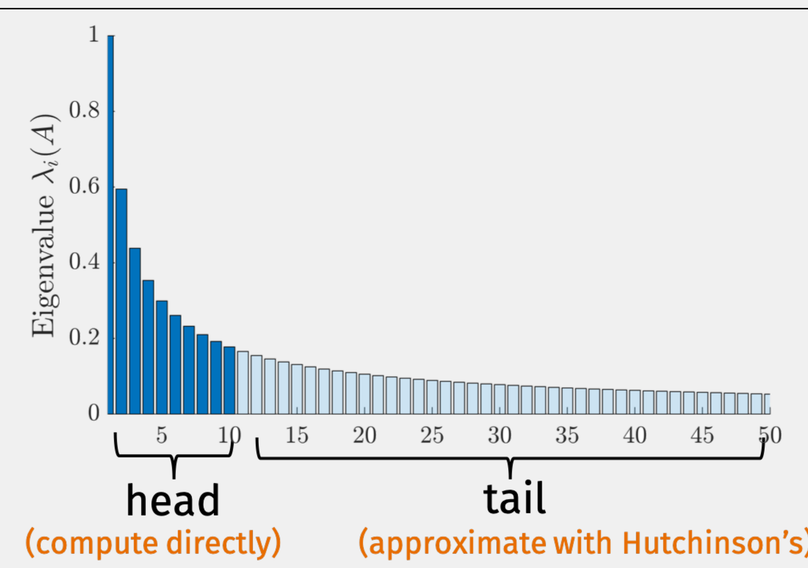
- For $\boldsymbol{A} \succeq 0$, we have $\|\boldsymbol{A}\|_F \le \mathrm{tr}(\boldsymbol{A})$, so that

$$\begin{aligned} |\mathrm{H}_\ell(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| &\le O(\tfrac{1}{\sqrt{\ell}})\|\boldsymbol{A}\|_F && \text{(Chebyshev Ineq.)} \\ &\le O(\tfrac{1}{\sqrt{\ell}})\,\mathrm{tr}(\boldsymbol{A}) && (\|\boldsymbol{A}\|_F \le \mathrm{tr}(\boldsymbol{A})) \\ &= \varepsilon\,\mathrm{tr}(\boldsymbol{A}) && (\ell = O(\tfrac{1}{\varepsilon^2})) \end{aligned}$$

- This analysis is only tight if $\|\boldsymbol{A}\|_F \approx \mathrm{tr}(\boldsymbol{A})$!
- $\|\boldsymbol{A}\|_F \approx \mathrm{tr}(\boldsymbol{A})$ only if $\boldsymbol{A}$'s eigenvalues are nearly sparse!

If $\boldsymbol{A}$ is hard for the Hutchinson estimator, then the sum of the top k eigenvalues represents most of $\mathrm{tr}(\boldsymbol{A})$.

Core Intuition



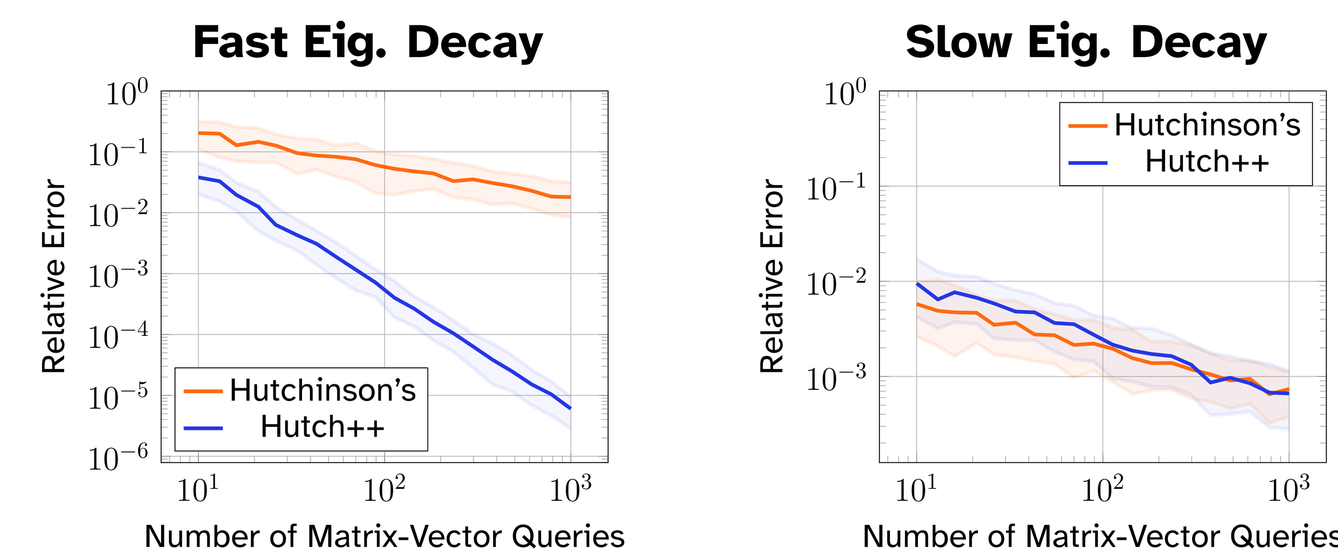
1. Find a good rank-k approximation $\tilde{\boldsymbol{A}}_k$
2. Notice that $\mathrm{tr}(\boldsymbol{A}) = \mathrm{tr}(\tilde{\boldsymbol{A}}_k) + \mathrm{tr}(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$
3. Compute $\mathrm{tr}(\tilde{\boldsymbol{A}}_k)$ exactly
4. Return Hutch++$(\boldsymbol{A}) :=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

Theorem: If $k = \ell = O(\frac{1}{\varepsilon})$, then $|$Hutch++$(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| \le \varepsilon\,\mathrm{tr}(\boldsymbol{A})$

Fundamental Rate: $|$Hutch++$(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| \le O(\frac{1}{\ell})\|\boldsymbol{A} - \boldsymbol{A}_k\|_F$

Conclusions

Experiments



(a) $\|\boldsymbol{A}\|_F = 0.63\,\mathrm{tr}(\boldsymbol{A})$ (b) $\|\boldsymbol{A}\|_F = 0.02\,\mathrm{tr}(\boldsymbol{A})$

Figure 1. When $\|\boldsymbol{A}\|_F \approx \mathrm{tr}(\boldsymbol{A})$, $\boldsymbol{A}$ has quickly decaying eigenvalues, and Hutch++ is much faster than H_ℓ.

Extensions Et Cetera

Lower Bounds

- All algorithms must use $\Omega(\frac{1}{\varepsilon})$ queries.

Indefinite Matrices

- We instead achieve $|\mathrm{tr}(\boldsymbol{A}) -$ Hutch++$(\boldsymbol{A})| \le \varepsilon\|\boldsymbol{A}\|_*$.

Non-Adaptive Algorithms

- We submit all queries in parallel. $\mathbf{x}_2$ cannot depend on $\boldsymbol{A}\mathbf{x}_1$.
- NA-Hutch++: Non-Adaptive variant of Hutch++, still $O(\frac{1}{\varepsilon})$
- No Adaptivity Gap

QR Codes & Links for More Details

(a) Hutch++ for Undergrads (b) Hutch++ Full Paper