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)

Motivation: Road Network Connectivity

- NYC is adding a new bus line. Where should it go?

Motivation: Road Network Connectivity

- NYC is adding a new bus line. Where should it go?
- Let's add a bus line that maximizes *connectivity*

Motivation: Road Network Connectivity

- NYC is adding a new bus line. Where should it go?
- Let's add a bus line that maximizes *connectivity*
- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory

Motivation: Road Network Connectivity

- NYC is adding a new bus line. Where should it go?
- Let's add a bus line that maximizes *connectivity*
- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- For each possible new route, build a new $(n+1) \times (n+1)$ adjacency matrix, and compute the change in connectivity

Motivation: Road Network Connectivity

- NYC is adding a new bus line. Where should it go?
- Let's add a bus line that maximizes *connectivity*
- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- For each possible new route, build a new $(n+1) \times (n+1)$ adjacency matrix, and compute the change in connectivity
- We have to compute the connectivity of a graph *very quickly*

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $= \operatorname{tr}(e^{\boldsymbol{B}})$

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $=$ $\operatorname{tr}(e^{\boldsymbol{B}})$
  - Num of Triangles $=$ $\operatorname{tr}(\frac{1}{6}\boldsymbol{B}^3)$

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $=$ $\mathrm{tr}(e^{\boldsymbol{B}})$
  - Num of Triangles $=$ $\mathrm{tr}(\frac{1}{6}\boldsymbol{B}^3)$
- The trace is the sum of the diagonal of a matrix

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $=$ $\operatorname{tr}(e^{\boldsymbol{B}})$
  - Num of Triangles $=$ $\operatorname{tr}(\frac{1}{6}\boldsymbol{B}^3)$
- The trace is the sum of the diagonal of a matrix
- Computing $\boldsymbol{B}^3$ takes $O(n^3)$ time. *slow*

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $=$ $\text{tr}(e^{\boldsymbol{B}})$
  - Num of Triangles $=$ $\text{tr}(\frac{1}{6}\boldsymbol{B}^3)$
- The trace is the sum of the diagonal of a matrix
- Computing $\boldsymbol{B}^3$ takes $O(n^3)$ time. *slow*
- Computing $\boldsymbol{B}^3\mathbf{x} = \boldsymbol{B}(\boldsymbol{B}(\boldsymbol{B}\mathbf{x}))$ takes $O(n^2)$ time. *fast*

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $=$ $\mathrm{tr}(e^{\boldsymbol{B}})$
  - Num of Triangles $=$ $\mathrm{tr}(\frac{1}{6}\boldsymbol{B}^3)$
- The trace is the sum of the diagonal of a matrix
- Computing $\boldsymbol{B}^3$ takes $O(n^3)$ time. *slow*
- Computing $\boldsymbol{B}^3\mathbf{x} = \boldsymbol{B}(\boldsymbol{B}(\boldsymbol{B}\mathbf{x}))$ takes $O(n^2)$ time. *fast*

Can we approximate $\mathrm{tr}(\boldsymbol{B}^3)$ by computing few $\boldsymbol{B}^3\mathbf{x}_1, \ldots, \boldsymbol{B}^3\mathbf{x}_k$?

Concrete Computational Problem

- We have an adjacency matrix $\boldsymbol{B} \in \mathbb{R}^{n \times n}$ in memory
- We want to measure the connectivity of the graph:
  - Estrada Index $=$ $\text{tr}(e^{\boldsymbol{B}})$
  - Num of Triangles $=$ $\text{tr}(\frac{1}{6}\boldsymbol{B}^3)$
- The trace is the sum of the diagonal of a matrix
- Computing $\boldsymbol{B}^3$ takes $O(n^3)$ time. *slow*
- Computing $\boldsymbol{B}^3\mathbf{x} = \boldsymbol{B}(\boldsymbol{B}(\boldsymbol{B}\mathbf{x}))$ takes $O(n^2)$ time. *fast*

Can we approximate $\text{tr}(\boldsymbol{B}^3)$ by computing few $\boldsymbol{B}^3\mathbf{x}_1, \ldots, \boldsymbol{B}^3\mathbf{x}_k$?

- **Yes we can!**

Overview

1. Introduction
   - What problems am I solving?
   - Why are these problems interesting?
   - How am I solving them?
2. Trace Estimation *(SOSA 2021)*
   - Prior State-of-the-Art
   - When can this be improved?
   - New Algorithm: Hutch++

General Picture: Trace Estimation

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

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

General Picture: Trace Estimation

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

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

- 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
$\mathrm{tr}(\frac{1}{6}\boldsymbol{B}^3)$	$\mathrm{tr}(e^{\boldsymbol{B}})$	$\mathrm{tr}(\ln(\boldsymbol{B}))$

General Picture: Trace Estimation

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

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

- 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
$\text{tr}(\frac{1}{6}\boldsymbol{B}^3)$	$\text{tr}(e^{\boldsymbol{B}})$	$\text{tr}(\ln(\boldsymbol{B}))$

- Goal: Estimate $\text{tr}(\boldsymbol{A})$ by computing $\boldsymbol{A}\mathbf{x}_1, \ldots \boldsymbol{A}\mathbf{x}_k$

General Picture: Trace Estimation

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

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

- 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
$\operatorname{tr}(\frac{1}{6}\boldsymbol{B}^3)$	$\operatorname{tr}(e^{\boldsymbol{B}})$	$\operatorname{tr}(\ln(\boldsymbol{B}))$

- Goal: Estimate $\operatorname{tr}(\boldsymbol{A})$ by computing $\boldsymbol{A}\mathbf{x}_1, \ldots \boldsymbol{A}\mathbf{x}_k$

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

$$|\tilde{\operatorname{tr}}(\boldsymbol{A}) - \operatorname{tr}(\boldsymbol{A})| \leq \varepsilon \operatorname{tr}(\boldsymbol{A})$$

Our Contributions

Prior Work:

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

Our Contributions

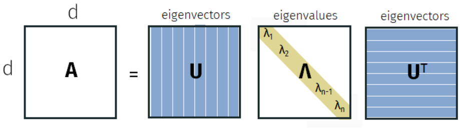
Prior Work:

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

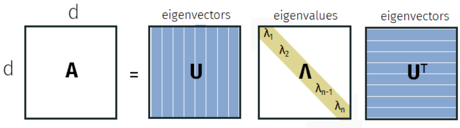
Our Results:

- Hutch++ Estimator: $O(\frac{1}{\varepsilon})$ products suffice
  - 5 Lines of MATLAB code
- Lower Bound: Any estimator needs $\Omega(\frac{1}{\varepsilon})$ products

Linear Algebra Review

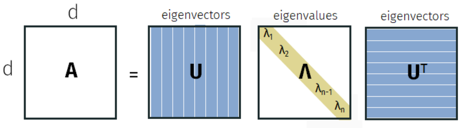


- Symmetric $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ has $\boldsymbol{A} = \boldsymbol{U} \Lambda \boldsymbol{U}^\intercal$
- $\boldsymbol{U}$ is a rotation matrix: $\boldsymbol{U}^\intercal \boldsymbol{U} = \boldsymbol{I}$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$



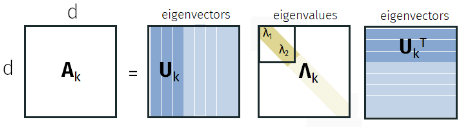
- Symmetric $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ has $\boldsymbol{A} = \boldsymbol{U} \Lambda \boldsymbol{U}^\top$
- $\boldsymbol{U}$ is a rotation matrix: $\boldsymbol{U}^\top \boldsymbol{U} = \boldsymbol{I}$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
- $\|\boldsymbol{A}\|_F^2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

Linear Algebra Review



- Symmetric $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ has $\boldsymbol{A} = \boldsymbol{U}\Lambda\boldsymbol{U}^\intercal$
- $\boldsymbol{U}$ is a rotation matrix: $\boldsymbol{U}^\intercal\boldsymbol{U} = \boldsymbol{I}$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
- $\|\boldsymbol{A}\|_F^2 = \sum_{i,j} \boldsymbol{A}_{i,j}^2 = \sum_i \lambda_i^2$
- $\mathrm{tr}(\boldsymbol{A}) = \sum_i \boldsymbol{A}_{i,i} = \sum_i \lambda_i$
- Positive Semi-Definite (PSD) $\boldsymbol{A}$ has $\lambda_i \geq 0$ for all i
  - $\|\boldsymbol{A}\|_F = \|\boldsymbol{\lambda}\|_2 \leq \|\boldsymbol{\lambda}\|_1 = \mathrm{tr}(\boldsymbol{A})$

Linear Algebra Review



- Symmetric $\boldsymbol{A} \in \mathbb{R}^{d \times d}$ has $\boldsymbol{A} = \boldsymbol{U} \Lambda \boldsymbol{U}^\intercal$
- $\boldsymbol{U}$ is a rotation matrix: $\boldsymbol{U}^\intercal \boldsymbol{U} = \boldsymbol{I}$
- Eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_d$
- $\|\boldsymbol{A}\|_F^2 = \sum_{i,j} \boldsymbol{A}_{i,j}^2 = \sum_i \lambda_i^2$
- $\mathrm{tr}(\boldsymbol{A}) = \sum_i \boldsymbol{A}_{i,i} = \sum_i \lambda_i$
- Positive Semi-Definite (PSD) $\boldsymbol{A}$ has $\lambda_i \geq 0$ for all i
  - $\|\boldsymbol{A}\|_F = \|\boldsymbol{\lambda}\|_2 \leq \|\boldsymbol{\lambda}\|_1 = \mathrm{tr}(\boldsymbol{A})$
- Low Rank Approximation:
  $\boldsymbol{A}_k = \boldsymbol{U}_k \Lambda_k \boldsymbol{U}_k^\intercal = \mathrm{argmin}_{rank(\boldsymbol{B})=k} \|\boldsymbol{A} - \boldsymbol{B}\|_F$

Hutchinson's Estimator

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

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

Hutchinson's Estimator

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

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

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

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

Hutchinson's Estimator

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

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

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

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

Proof: $\mathrm{H}_\ell(\boldsymbol{A})$ needs $\ell = O(\frac{1}{\varepsilon^2})$ for PSD $\boldsymbol{A}$

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

Hutchinson's Estimator

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

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

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

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

Proof: $\mathrm{H}_\ell(\boldsymbol{A})$ needs $\ell = O(\frac{1}{\varepsilon^2})$ for PSD $\boldsymbol{A}$

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

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

Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

$$\begin{aligned} |\mathrm{H}_\ell(\boldsymbol{A}) - \operatorname{tr}(\boldsymbol{A})| &\leq O(\tfrac{1}{\sqrt{\ell}})\|\boldsymbol{A}\|_F \\ &\leq O(\tfrac{1}{\sqrt{\ell}}) \operatorname{tr}(\boldsymbol{A}) \\ &= \varepsilon \operatorname{tr}(\boldsymbol{A}) \end{aligned}$$

Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

$$\begin{aligned} |\mathrm{H}_\ell(\boldsymbol{A}) - \operatorname{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{\ell}})\|\boldsymbol{A}\|_F \\ &\leq O(\tfrac{1}{\sqrt{\ell}}) \operatorname{tr}(\boldsymbol{A}) \\ &= \varepsilon \operatorname{tr}(\boldsymbol{A}) \end{aligned}$$

Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

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

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

Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

$$
\begin{aligned}
|\mathrm{H}_\ell(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{\ell}}) \|\boldsymbol{A}\|_F \\
&\leq O(\tfrac{1}{\sqrt{\ell}})\, \mathrm{tr}(\boldsymbol{A}) \\
&= \varepsilon\, \mathrm{tr}(\boldsymbol{A})
\end{aligned}
$$

- 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 $\boldsymbol{A}$

For what $\boldsymbol{A}$ is this analysis tight?

$$\begin{aligned} |\mathrm{H}_\ell(\boldsymbol{A}) - \operatorname{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{\ell}}) \|\boldsymbol{A}\|_F \\ &\leq O(\tfrac{1}{\sqrt{\ell}}) \operatorname{tr}(\boldsymbol{A}) \\ &= \varepsilon \operatorname{tr}(\boldsymbol{A}) \end{aligned}$$

- 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?

Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

$$
\begin{aligned}
|\mathrm{H}_\ell(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{\ell}})\|\boldsymbol{A}\|_F \\
&\leq O(\tfrac{1}{\sqrt{\ell}})\,\mathrm{tr}(\boldsymbol{A}) \\
&= \varepsilon\,\mathrm{tr}(\boldsymbol{A})
\end{aligned}
$$

- 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?
    - Property of norms: $\|\mathbf{v}\|_2 \approx \|\mathbf{v}\|_1$ only if $\mathbf{v}$ is nearly sparse

Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

$$
\begin{aligned}
|\mathrm{H}_\ell(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{\ell}})\|\boldsymbol{A}\|_F \\
&\leq O(\tfrac{1}{\sqrt{\ell}})\,\mathrm{tr}(\boldsymbol{A}) \\
&= \varepsilon\,\mathrm{tr}(\boldsymbol{A})
\end{aligned}
$$

- 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?
  - Property of norms: $\|\mathbf{v}\|_2 \approx \|\mathbf{v}\|_1$ only if $\mathbf{v}$ is nearly sparse
  - Otherwise $\|\mathbf{v}\|_2 \ll \|\mathbf{v}\|_1$

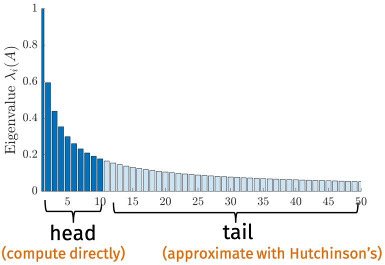
Hutchinson's Estimator

For what $\boldsymbol{A}$ is this analysis tight?

$$
\begin{aligned}
|\mathrm{H}_\ell(\boldsymbol{A}) - \mathrm{tr}(\boldsymbol{A})| &\approx O(\tfrac{1}{\sqrt{\ell}})\|\boldsymbol{A}\|_F \\
&\leq O(\tfrac{1}{\sqrt{\ell}})\,\mathrm{tr}(\boldsymbol{A}) \\
&= \varepsilon\,\mathrm{tr}(\boldsymbol{A})
\end{aligned}
$$

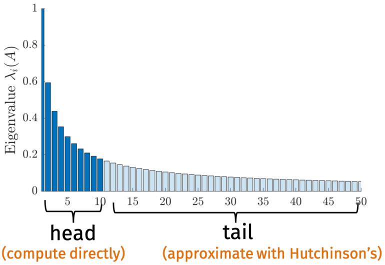
- When is the bound $\|\boldsymbol{A}\|_F \leq \mathrm{tr}(A)$ tight?
- Let $\mathbf{v} = \begin{bmatrix} \lambda_1 & \dots & \lambda_n \end{bmatrix}$ be the eigenvalues of PSD $\boldsymbol{A}$
- When is the bound $\|\mathbf{v}\|_2 \leq \|\mathbf{v}\|_1$ tight?
  - Property of norms: $\|\mathbf{v}\|_2 \approx \|\mathbf{v}\|_1$ only if $\mathbf{v}$ is nearly sparse
  - Otherwise $\|\mathbf{v}\|_2 \ll \|\mathbf{v}\|_1$
- 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.

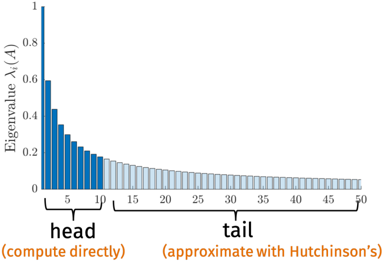
Helping Hutchinson's Estimator



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 $\text{tr}(\boldsymbol{A}) = \text{tr}(\tilde{\boldsymbol{A}}_k) + \text{tr}(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$
3. Compute $\text{tr}(\tilde{\boldsymbol{A}}_k)$ exactly
4. Return $\text{Hutch++}(\boldsymbol{A}) = \text{tr}(\tilde{\boldsymbol{A}}_k) + \text{H}_\ell(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$

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. Return $\text{Hutch++}(\boldsymbol{A}) = \text{tr}(\tilde{\boldsymbol{A}}_k) + \text{H}_\ell(\boldsymbol{A} - \tilde{\boldsymbol{A}}_k)$

If $k = \ell = O(\frac{1}{\varepsilon})$, then $|\text{Hutch++}(\boldsymbol{A}) - \text{tr}(\boldsymbol{A})| \leq \varepsilon \, \text{tr}(\boldsymbol{A})$.

(Whiteboard)

Finding a Good Low-Rank Approximation

Let $\boldsymbol{A}_k$ be the best rank-k approximation of $\boldsymbol{A}$.

Lemma [Sar06, Woo14]

Let $\boldsymbol{S} \in \mathbb{R}^{d \times O(k)}$ have $\mathcal{N}(0,1)$ entries

Let $\boldsymbol{Q} = \text{qr}(\boldsymbol{AS})$

Let $\tilde{\boldsymbol{A}}_k = \boldsymbol{AQQ}^\intercal$

Then, with high probability

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

Finding a Good Low-Rank Approximation

Let $\boldsymbol{A}_k$ be the best rank-k approximation of $\boldsymbol{A}$.

Lemma [Sar06, Woo14]

Let $\boldsymbol{S} \in \mathbb{R}^{d \times O(k)}$ have $\mathcal{N}(0,1)$ entries

Let $\boldsymbol{Q} = \mathrm{qr}(\boldsymbol{AS})$

Let $\tilde{\boldsymbol{A}}_k = \boldsymbol{AQQ}^\intercal$

Then, with high probability

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

We can compute the trace of $\tilde{\boldsymbol{A}}_k$ with $O(k)$ queries and $O(dk)$ space:

$$\mathrm{tr}(\tilde{\boldsymbol{A}}_k) = \mathrm{tr}(\boldsymbol{AQQ}^\intercal) = \mathrm{tr}(\boldsymbol{Q}^\intercal(\boldsymbol{AQ}))$$

Hutch++

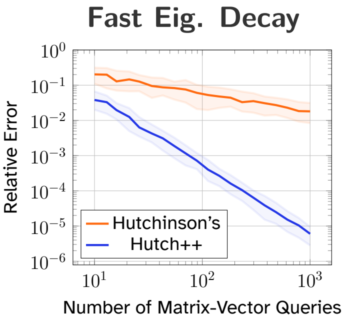
Hutch++ Algorithm:

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

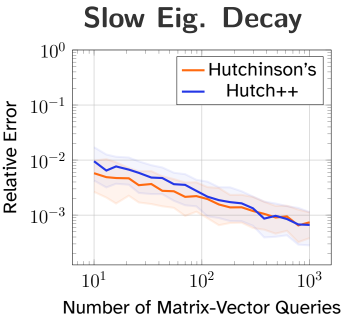
1. Sample $\boldsymbol{S} \in \mathbb{R}^{d \times \frac{m}{3}}$ and $\boldsymbol{G} \in \mathbb{R}^{d \times \frac{m}{3}}$ with i.i.d. $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$ entries
2. Compute $\boldsymbol{Q} = \mathrm{qr}(\boldsymbol{AS})$
3. Return $\mathrm{tr}(\boldsymbol{Q}^T \boldsymbol{AQ})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

```
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
```

When $\|\boldsymbol{A}\|_F \approx \mathrm{tr}(\boldsymbol{A})$, Hutch++ is much faster than H_ℓ:



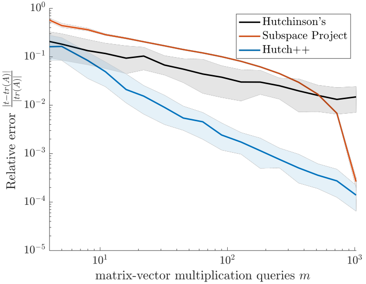
(a) $\|\boldsymbol{A}\|_F = 0.63\,\mathrm{tr}(\boldsymbol{A})$



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

Hutch++ works great for most matrices:

Error in $\text{tr}(\frac{1}{6}\boldsymbol{B}^3)$



Eigs. of $\boldsymbol{B}$

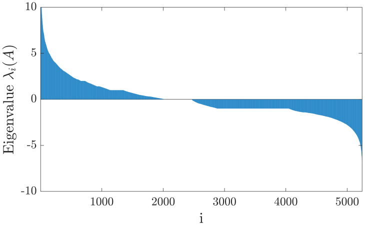


Figure: Estimating num of triangles of arXiv Citation Network

Open Questions

- When is adaptivity helpful?
- 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.
- 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.
Journal of the ACM, 58(2), 2011.

Tamas Sarlos.
Improved approximation algorithms for large matrices via random projections.
In Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS), pages 143–152, 2006.

David P. Woodruff.
Sketching as a tool for numerical linear algebra.
Foundations and Trends in Theoretical Computer Science, 10(1–2):1–157, 2014.

Karl Wimmer, Yi Wu, and Peng Zhang.
Optimal query complexity for estimating the trace of a matrix.
In Proceedings of the 41st International Colloquium on Automata, Languages and Programming (ICALP), pages 1051–1062, 2014.