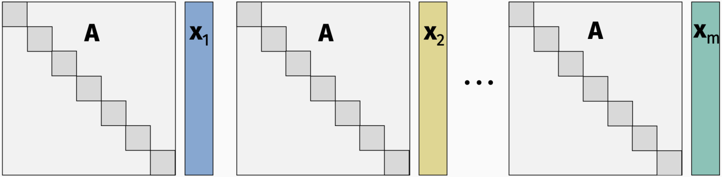
Hutch++: Optimal Stochastic Trace Estimation

Cameron Musco
University of Massachusetts Amherst

With: Raphael Meyer (NYU), Chris Musco (NYU), David Woodruff (CMU)

- Given access to an $n \times n$ matrix $\mathbf{A}$ through matrix-vector multiplication.
- Goal is to approximate $\text{tr}(\mathbf{A}) = \sum_{i=1}^{n} \mathbf{A}_{ii}$.



Main question: How many matrix-vector multiplication "queries" $\mathbf{A}\mathbf{x}_1, \ldots, \mathbf{A}\mathbf{x}_m$ are required to approximate $\text{tr}(\mathbf{A})$?

Algorithms in this model are called matrix-free methods.

- Useful when A is not given explicitly, but we have an efficient algorithm for multiplying A by a vector.

Algorithms in this model are called matrix-free methods.

- Useful when $\mathbf{A}$ is not given explicitly, but we have an efficient algorithm for multiplying $\mathbf{A}$ by a vector.

Example 1: Hessian/Jacobian matrix-vector products.

- For vector $\mathbf{x}$, $\nabla \mathbf{f}(\mathbf{y})\mathbf{x}$ and $\nabla^2 f(\mathbf{y})\mathbf{x}$ can often be computed efficiently using finite difference methods or explicit differentiation.
- Do not need to fully form $\nabla \mathbf{f}(\mathbf{y})$ or $\nabla^2 f(\mathbf{y})$.

Example 2: When $\mathbf{A}$ is a function of another (explicit) matrix $\mathbf{B}$:

$$\mathbf{A} = f(\mathbf{B})$$

- E.g., $\mathbf{A} = \mathbf{B}^3$ requires n^3 operations to form explicitly.
- Computing a matrix-vector product $\mathbf{A}\mathbf{x} = \mathbf{B}^3\mathbf{x}$ requires just $3n^2$ operations – as $\mathbf{B}(\mathbf{B}(\mathbf{B}\mathbf{x}))$.

For more complex matrix functions, we can often compute $\mathbf{A}\mathbf{x} = f(\mathbf{B})\mathbf{x}$ efficiently using iterative methods:

- Conjugate gradient, MINRES, or any linear system solver:

$$\mathbf{A} = \mathbf{B}^{-1}.$$

- Lanczos method, polynomial/rational approximation:

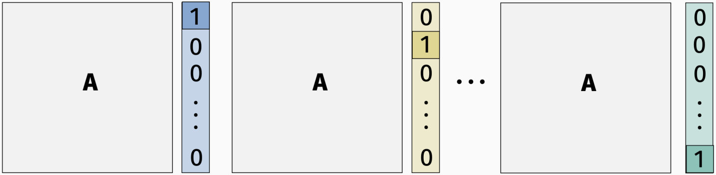
$$\mathbf{A} = \exp(\mathbf{B}),\ \mathbf{A} = \sqrt{\mathbf{B}},\ \mathbf{A} = \log(\mathbf{B}),\ \text{etc.}$$

These methods run in $n^2 \cdot C$ time, where C depends on properties of $\mathbf{B}$. Typically $C \ll n$ so $n^2 \cdot C \ll n^3$.

- Triangle counting in graphs. $\text{tr}(\mathbf{B}^3) = 6 \cdot (\#\ \textit{triangles})$, where $\mathbf{B}$ is the adjacency matrix.
- Log-likelihood computation in Bayesian optimization, experimental design. $\text{tr}(\log(\mathbf{B})) = \text{logdet}(\mathbf{B})$.
- Estrada index, a measure of protein folding degree and more generally, network connectivity. $\text{tr}(\exp(\mathbf{B}))$.
- Information about the matrix eigenvalue spectrum, since $\text{tr}(\mathbf{A}) = \sum_{i=1}^{n} \lambda_i$, where λ_i is $\mathbf{A}$'s i^{th} eigenvalue.
- E.g., counting the number of eigenvalues in an interval, spectral density estimation, matrix norms
- See e.g., [Ubaru, and Saad 2017].

Naive matrix-free trace estimation:

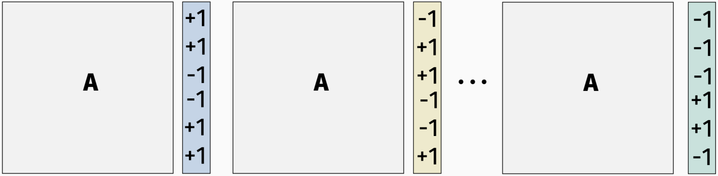
- Set $\mathbf{x}_i = \mathbf{e}_i$ for $i = 1, \ldots, n$.
- Return $\text{tr}(\mathbf{A}) = \sum_{i=1}^{n} \mathbf{x}_i^T \mathbf{A} \mathbf{x}_i$.



Returns exact solution, but requires n matrix-vector multiplies. We want $\ll n$ multiplies. Will achieve this by allowing for approximation.

Hutchinson 1991, Girard 1987:

- Draw $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^n$ i.i.d. with random $\{+1, -1\}$ entries.
- Return $\tilde{T} = \frac{1}{m}\sum_{i=1}^{m} \mathbf{x}_i^T \mathbf{A}\mathbf{x}_i$ as an approximation to $\text{tr}(\mathbf{A})$.



- Can also let $\mathbf{x}_1, \ldots, \mathbf{x}_m \in \mathbb{R}^n$ have i.i.d. Gaussian entries, however the distinction isn't important for this talk.

Claim (Hanson, Wright '71, Avron, Toledo '11, Roosta, Ascher '15, Cortinovis, Kressner '20)

Let $\tilde{T}$ be the trace estimate returned by Hutchinson's method. If $m \approx \frac{1}{\epsilon^2}$, then with 'high probability',

$$\left|\tilde{T} - \mathrm{tr}(\mathbf{A})\right| \leq \epsilon \|\mathbf{A}\|_F$$

If $\mathbf{A}$ is symmetric positive semidefinite (PSD) then

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^{n} \lambda_i^2} \leq \sum_{i=1}^{n} \lambda_i = \mathrm{tr}(\mathbf{A}).$$

So for PSD A: $\quad (1-\epsilon)\,\mathrm{tr}(\mathbf{A}) \leq \tilde{T} \leq (1+\epsilon)\,\mathrm{tr}(\mathbf{A}).$

Result: $\approx 1/\epsilon^2$ matrix-vector multiplies suffice to return a trace estimate for a PSD matrix satisfying:

$$(1-\epsilon)\,\mathrm{tr}(\mathbf{A}) \leq \tilde{T} \leq (1 \pm \epsilon)\,\mathrm{tr}(\mathbf{A}).$$

Research Question: Can this be improved?

Broader line of work: Tight upper bounds and lower bounds on complexity of basic linear algebra problems in "matrix-vector query" model.

- **Top eigenvector:** Simchowitz, Alaoui, Recht, 2018.
- **Least squares regression:** Braverman, Hazan, Simchowitz, Woodworth, 2020.
- **Rank, symmetry test, and more:** Sun, Woodruff, Yang, and Zhang, 2019.

The matrix-vector query model generalizes some of the most common models of computation in linear algebra.

Krylov subpace model:

- Compute $\mathbf{A}\mathbf{x}, \mathbf{A}^2\mathbf{x}, \ldots, \mathbf{A}^m\mathbf{x}$ for a single vector $\mathbf{x}$.
- Lower bounds typically via approximation theoretic arguments (understanding the limits of polynomials).

Matrix sketching model:

- Compute $\mathbf{A}\mathbf{x}_1, \ldots, \mathbf{A}\mathbf{x}_m$ where $\mathbf{x}_1, \ldots, \mathbf{x}_m$ are chosen non-adaptively (usually randomly).
- Lower bounds typically via one-round communication complexity. See e.g., [Woodruff '14].

Merits of this model:

- Captures many algorithms that are used in practice.
- Allowing arbitrary adaptivity makes the model quite a bit richer. Proving lower bounds seems harder but doable.
- Seems to be a "sweet spot" for understanding problem complexity in linear algebra.

Limitation:

- Does not capture methods like stochastic gradient or coordinate descent, certain sparse methods and preconditioning approaches, etc.

Upper bound: $\approx 1/\epsilon$ matrix-vector multiplies suffice to return, with high prob., a trace estimate for a PSD matrix satisfying:

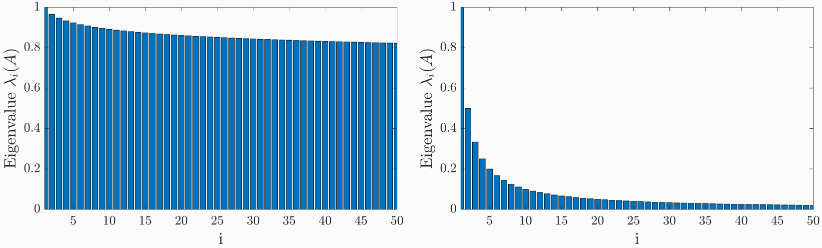
$$(1-\epsilon)\operatorname{tr}(\mathsf{A}) \leq \tilde{T} \leq (1+\epsilon)\operatorname{tr}(\mathsf{A}).$$

- Quadratic improvement over Hutchinson's $\approx 1/\epsilon^2$.
- Algorithm achieving bound is nearly as simple.
- Variants have been studied e.g. in [Gambhir, Stathopoulos, Orginos '17] and [Lin '17].
- Performs much better experimentally.

Lower bound: $\gtrsim 1/\epsilon$ matrix-vector multiplies are necessary to obtain such an approximation.

- Two different approaches: reduction from multi-round communication complexity, and from hypothesis testing for negatively spiked covariance matrices.

Observation: Hutchinson's method performs much better when $\mathbf{A}$ has a flat spectrum.

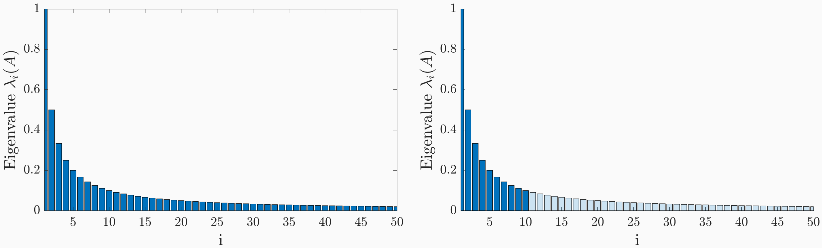


We have that: $|\tilde{T} - \mathrm{tr}(\mathbf{A})| \leq \epsilon \|\mathbf{A}\|_F \leq \epsilon\, \mathrm{tr}(\mathbf{A})$, but when the spectrum is flat $\|\mathbf{A}\|_F \ll \mathrm{tr}(\mathbf{A})$.

In the extreme case when $\lambda_1 = \lambda_2 = \ldots = \lambda_n$, we have:

$$\|\mathbf{A}\|_F = \sqrt{\sum_{i=1}^n \lambda_i^2} = \frac{1}{\sqrt{n}} \sum_{i=1}^n \lambda_i = \frac{1}{\sqrt{n}} \mathrm{tr}(\mathbf{A}).$$

On the other hand, when A's spectrum is decaying, we get a good approximation by simply computing its top eigenvalues.



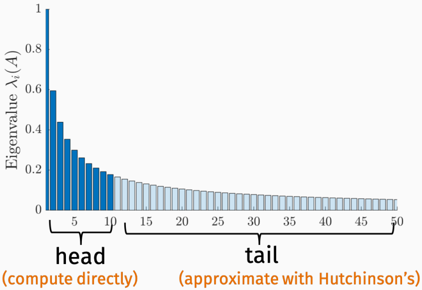
$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} \lambda_i \approx \sum_{i=1}^{k} \lambda_k = \mathrm{tr}(\mathbf{A}\mathbf{Q}\mathbf{Q}^T)$$

where $\mathbf{Q} \in \mathbb{R}^{n \times m}$ is an orthonormal span for A's top k eigenvectors. .

- Q itself can be computed with $\approx k$ matrix-vector multiplication queries using block power method or a Krylov method.
- Then $\text{tr}(\mathbf{AQQ}^T) = \text{tr}(\mathbf{Q}^T(\mathbf{AQ}))$ can be computed with k additional matrix-vector multiplies.
- Fairly common approach, employed e.g. by [Tsourakakis '08], [Lin Lin '17], [Gambhir, Stathopoulos, Orginos '17], [Saibaba, Alexanderian, Ipsen, '18], and [Zhu, Li '20].

Our Observation: Every spectrum is either "flat enough" or "decaying enough" to prove a better bound than $\approx 1/\epsilon^2$.

1. Find approximate span for top k eigenvectors $\mathbf{Q}$.
2. Observe that $\text{tr}(\mathbf{A}) = \text{tr}(\mathbf{A}\mathbf{Q}\mathbf{Q}^T) + \text{tr}(\mathbf{A}(\mathbf{I} - \mathbf{Q}\mathbf{Q}^T))$
3. Approximate $\tilde{P} = \text{tr}(\mathbf{A}(\mathbf{I} - \mathbf{Q}\mathbf{Q}^T))$ using Hutchinson's with ℓ random query vectors.
4. Return $\tilde{T} = \text{tr}(\mathbf{A}\mathbf{Q}\mathbf{Q}^T) + \tilde{P}$.



The only error is from the estimator for $\text{tr}(\mathbf{A}(\mathbf{I} - \mathbf{Q}\mathbf{Q}^T))$, which will have much lower variance if $\|\mathbf{A}(\mathbf{I} - \mathbf{Q}\mathbf{Q}^T)\|_F \ll \|\mathbf{A}\|_F$.

Standard result in Randomized Numerical Linear Algebra:

Lemma (Sarlos 2006)

If $\mathsf{S} \in \mathbb{R}^{n \times m}$ *is chosen with i.i.d.* ± 1 *entries for* $m \approx k$*, then* $\mathsf{Q} = \text{orth}(\mathsf{AS})$ *satisfies with high probability,*

$$\|\mathsf{A}(\mathsf{I} - \mathsf{Q}\mathsf{Q}^T)\|_F \leq 2\|\mathsf{A} - \mathsf{A}_k\|_F,$$

Here A_k is the optimal k-rank approximation to A, obtained by projecting onto A's top k eigenvectors.

Q can be viewed as the result of running a single step of block power method on A.

Basic Fact: For any PSD matrix $\mathbf{A}$:

$$\|\mathbf{A}-\mathbf{A}_k\|_F \leq \frac{1}{\sqrt{k}} \cdot \mathrm{tr}(\mathbf{A})$$

So if $\left\|\mathbf{A}(\mathbf{I}-\mathbf{Q}\mathbf{Q}^T)\right\|_F \leq 2\,\|\mathbf{A}-\mathbf{A}_k\|_F$, then with high probability,

$$\left|\mathrm{tr}(\mathbf{A}(\mathbf{I}-\mathbf{Q}\mathbf{Q}^T) - \tilde{P}\right| \lesssim \frac{1}{\sqrt{\ell}}\left\|\mathbf{A}(\mathbf{I}-\mathbf{Q}\mathbf{Q}^T)\right\|_F \leq \frac{1}{\sqrt{\ell}} \cdot \frac{2}{\sqrt{k}}\,\mathrm{tr}(\mathbf{A}).$$

Setting $\ell = k \approx 1/\epsilon$ gives error $\epsilon\,\mathrm{tr}(\mathbf{A})$ and thus:

$$\left|\mathrm{tr}(\mathbf{A}) - \tilde{T}\right| = \left|\mathrm{tr}(\mathbf{A}(\mathbf{I}-\mathbf{Q}\mathbf{Q}^T)) - \tilde{P}\right| \leq \epsilon\,\mathrm{tr}(\mathbf{A}).$$

FINAL ALGORITHM

Theorem (Final Result)

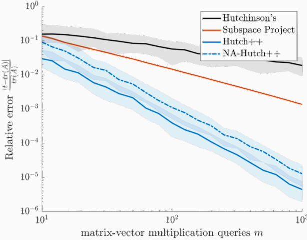
If $\ell = k \approx \frac{1}{\epsilon}$ and $\mathbf{A}$ is PSD then with high probability, Hutch++ uses $2k + \ell$ queries and returns $\tilde{T}$ satisfying:

$$(1-\epsilon)\operatorname{tr}(\mathbf{A}) \leq \tilde{T} \leq (1+\epsilon)\operatorname{tr}(\mathbf{A}).$$

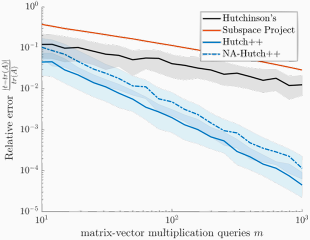
```
function T = hutchplusplus(A, m)
    S = 2*randi(2,size(A,1),m/3);
    G = 2*randi(2,size(A,1),m/3);
    [Q,~] = qr(A*S,0);
    G = G - Q*(Q'*G);
    T = trace(Q'*A*Q) + 1/size(G,2)*trace(G'*A*G);
end
```

Hutch++ is adaptive, meaning that the choice of $\mathbf{x}_i$ depends on $\mathbf{A}\mathbf{x}_1, \ldots, \mathbf{A}\mathbf{x}_{i-1}$. We also give a non-adaptive method, NA-Hutch++ that achieves the same bound, up to constants.

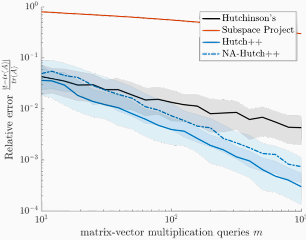
Results on synthetic matrix $\mathbf{A}$ with spectrum $\lambda_i = i^{-c}$ for different values of c.



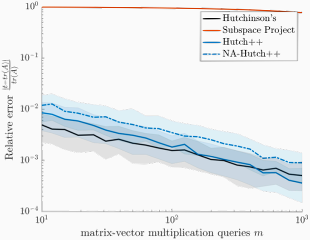
(a) Fast Eigenvalue Decay ($c = 2$)



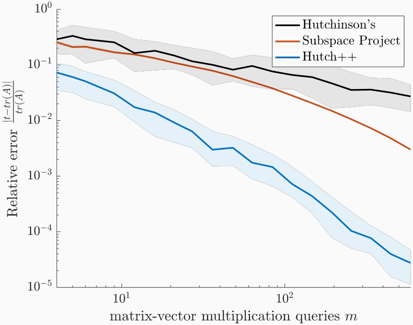
(b) Medium Eigenvalue Decay ($c = 1.5$)



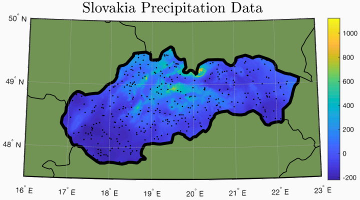
(c) Slow Eigenvalue Decay ($c = 1$)

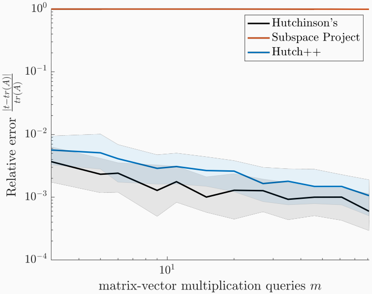


(d) Very Slow Eigenvalue Decay ($c = .5$)



$\mathrm{A} = \exp(\mathrm{B})$ for graph adjacency matrix B from linguistics application. $\mathrm{tr}(\mathrm{A})$ is the well known Estrada Index or "natural connectivity", originally used in analyzing protein folding.



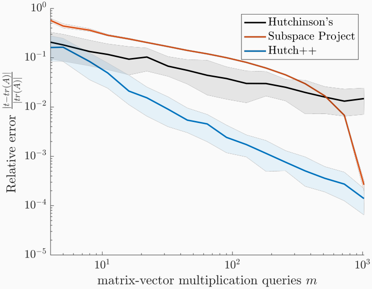


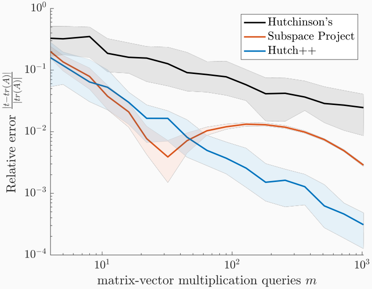
$\mathbf{A} = \log(\mathbf{B} + \lambda\mathbf{I})$ for kernel matrix $\mathbf{B}$ from Gaussian process regression. $\text{tr}(\mathbf{A}) = \log\det(\mathbf{B})$, which is used in log likelihood calculations for hyperparameter optimization.

Takeaway: For matrix functions that flatten $\mathbf{B}$'s spectrum, Hutchinson's estimator performs far better than the $\approx 1/\epsilon^2$ bound predicts. Hutch++ doesn't perform much worse.

Hutch++ works well empirically for many non-PSD matrices.

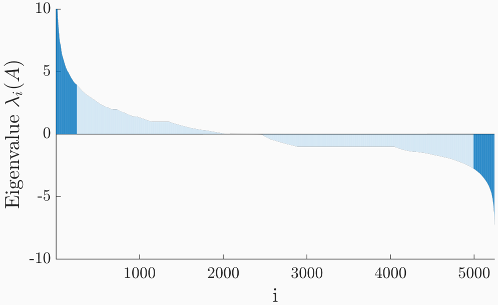
Let $\mathbf{B}$ be the adjacency matrix of an undirected graph G, $\text{tr}(\mathbf{B}^3)/6$ is equal to the number of triangles in G.





$\mathbf{A} = \mathbf{B}^3$ for arXiv.org citation network and Wikipedia voting network.

For non-PSD $\mathbf{A}$, the projection step, $\mathbf{A}(\mathbf{I} - \mathbf{Q}\mathbf{Q}^T)$ approximately removes $\mathbf{A}$'s largest magnitude eigenvalues, which can still reduce variance substantially.



Spectrum of $\mathbf{A} = \mathbf{B}^3$ for arXiv.org citation network.

Theorem

Any algorithm that accesses PSD matrix $\mathbf{A}$ *via queries* $\mathbf{A}\mathbf{x}_1, \ldots, \mathbf{A}\mathbf{x}_m$*, where* $\mathbf{x}_1, \ldots, \mathbf{x}_m$ *are possibly adaptively chosen vectors with integer entries in* $\{-2^b, ..., 2^b\}$*, needs*

$$m \gtrsim \frac{1}{\epsilon \cdot [b + \log(1/\epsilon)]} \text{ queries}$$

to approximate $\mathrm{tr}(\mathbf{A})$ *to multiplicative error* $(1 \pm \epsilon)$*.*

- **Reduction to 2-party multi-round communication problem.** "Hard" input distribution will involve $\mathbf{A}$ with integer entries, which is why we need the bit complexity bound b.
- Also have a tight lower bound in the real-RAM model of computation.

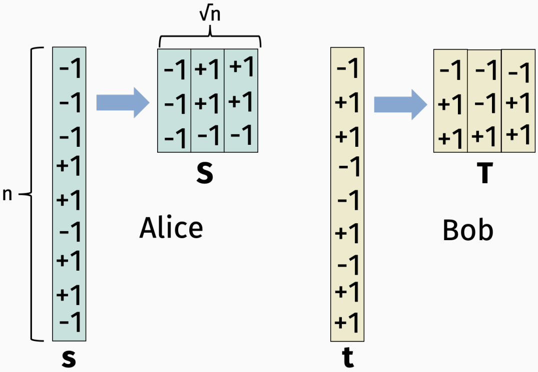
Problem (Gap Hamming)

Let Alice and Bob be communicating parties who hold vectors $\mathbf{s}, \mathbf{t} \in \{-1, 1\}^n$*, respectively. Must decide with few bits of communication if:*

$$\langle \mathbf{s}, \mathbf{t} \rangle \geq \sqrt{n} \qquad \text{or} \qquad \langle \mathbf{s}, \mathbf{t} \rangle \leq -\sqrt{n}$$

Theorem (Chakrabarti, Regev 2012)

The randomized communication complexity for solving Problem 1 with probability at least $2/3$ *is* $\gtrsim n$ *bits.*



Let $\mathbf{Z} = \mathbf{S} + \mathbf{T}$ and $\mathbf{A} = \mathbf{Z}^T\mathbf{Z}$.

$$\operatorname{tr}(\mathbf{A}) = \|\mathbf{Z}\|_F^2 = \|\mathbf{s} + \mathbf{t}\|_2^2 = 2n - 2\langle \mathbf{s}, \mathbf{t}\rangle.$$

So if Alice and Bob estimate $\operatorname{tr}(\mathbf{A})$ up to error $(1 \pm 1/\sqrt{n})$, then they will solve the Gap Hamming problem.

Claim: Alice and Bob can simulate any m query algorithm for estimating the trace of $\mathbf{A} = (\mathbf{S}+\mathbf{T})^T(\mathbf{S}+\mathbf{T})$ with $\approx m\sqrt{n}(\log n + b)$ bits of communication.

- Alice decides on $\mathbf{x}_1$, sends to Bob with $\sqrt{n} \cdot \log(2^b)$ bits.
- Bob computes $\mathbf{T}\mathbf{x}_1$, sends to Alice with $\sqrt{n} \cdot \log(\sqrt{n}2^b)$ bits.
- Alice computes $(\mathbf{S}+\mathbf{T})\mathbf{x}_1$.
- Repeat to multiply $(\mathbf{S}+\mathbf{T})\mathbf{x}_1$ by $(\mathbf{S}+\mathbf{T})^T$
- Alice decides on $\mathbf{x}_2$, process repeats m times.

So, by the $\gtrsim n$ lower bound for Gap Hamming, we have

$$m \gtrsim \frac{\sqrt{n}}{\log n + b} = \frac{1}{\epsilon \cdot (\log 1/\epsilon + b)} \text{ for } \epsilon = 1/\sqrt{n}.$$

FUTURE WORK

- Lower bounds for e.g., $\mathrm{tr}(\mathbf{A}^3)$, $\mathrm{tr}(\exp(\mathbf{A}))$, $\mathrm{tr}(\mathbf{A}^{-1})$ showing that Hutch++ combined with iterative matrix methods is optimal in the matrix-vector query model.
- Conditional lower bounds for simple problems like triangle counting in a more general computational model.
- Faster algorithms for spectral density estimation and other problems by combining trace estimation with randomized approximate matrix vector multiplication (using e.g., entrywise sampling).
- Practical use cases and implementations of Hutch++.
- Recent applications include to quantum typicality methods [Weinberg '21] and Hessian trace estimation in optimization [Agrawal, Ali, Boyd '21]

THANKS! QUESTIONS?