Error Estimation for Randomized Numerical Linear Algebra via the Bootstrap

Miles E. Lopes Shusen Wang Michael W. Mahoney

UC Davis ICSI & UC Berkeley

Randomized numerical linear algebra (RandNLA)

- Randomized (sketching) methods have been intensively studied in order to accelerate large-scale matrix computations.

Randomized numerical linear algebra (RandNLA)

- Randomized (sketching) methods have been intensively studied in order to accelerate large-scale matrix computations.
    - matrix multiplication
    - least squares
    - SVD / low-rank approximation
    - Netwon methods
    - . . .

Randomized numerical linear algebra (RandNLA)

- Randomized (sketching) methods have been intensively studied in order to accelerate large-scale matrix computations.
  - matrix multiplication
  - least squares
  - SVD / low-rank approximation
  - Netwon methods
  - ...
- Randomized methods can be competitive with highly optimized software (e.g. LAPACK)

Randomized numerical linear algebra (RandNLA)

- Randomized (sketching) methods have been intensively studied in order to accelerate large-scale matrix computations.
    - matrix multiplication
    - least squares
    - SVD / low-rank approximation
    - Netwon methods
    - ...
- Randomized methods can be competitive with highly optimized software (e.g. LAPACK)
- In exchange for reduced cost, randomized solutions also come with (random) approximation error.

Trading off computational cost and accuracy

Key question: How large is the error of a given randomized solution?

Trading off computational cost and accuracy

Key question: How large is the error of a given randomized solution?

- For many types of problems, theoretical guarantees can provide a good qualitative description of the relationship between cost and accuracy.

Trading off computational cost and accuracy

Key question: How large is the error of a given randomized solution?

- For many types of problems, theoretical guarantees can provide a good qualitative description of the relationship between cost and accuracy.
- However, such guarantees typically have limitations:
  - worst-case/pessimistic

Trading off computational cost and accuracy

Key question: How large is the error of a given randomized solution?

- For many types of problems, theoretical guarantees can provide a good qualitative description of the relationship between cost and accuracy.

- However, such guarantees typically have limitations:
  - worst-case/pessimistic
  - conservative or unknown constants

Trading off computational cost and accuracy

Key question: How large is the error of a given randomized solution?

- For many types of problems, theoretical guarantees can provide a good qualitative description of the relationship between cost and accuracy.

- However, such guarantees typically have limitations:
  - worst-case/pessimistic
  - conservative or unknown constants
  - ignore unique problem structure

Practical error bounds?

- **Problem:** It is difficult to use theoretical error bounds in practice to assess the error of a given solution.

Practical error bounds?

- **Problem:** It is difficult to use theoretical error bounds in practice to assess the error of a given solution.
- An alternative is to *numerically estimate* the error of a given solution: a posteriori error estimation (see, e.g. Verfürth 1996, Ainsworth and Oden 2000).

Practical error bounds?

- **Problem:** It is difficult to use theoretical error bounds in practice to assess the error of a given solution.
- An alternative is to *numerically estimate* the error of a given solution: a posteriori error estimation (see, e.g. Verfürth 1996, Ainsworth and Oden 2000).
- This has been considered in a few works in RandNLA, but is underdeveloped:
  Lopes et al., 2017, 2018, Halko et al., 2011, Woolfe et al., 2008, Liberty et al., 2007

Practical error bounds?

- **Problem:** It is difficult to use theoretical error bounds in practice to assess the error of a given solution.
- An alternative is to *numerically estimate* the error of a given solution: a posteriori error estimation (see, e.g. Verfürth 1996, Ainsworth and Oden 2000).
- This has been considered in a few works in RandNLA, but is underdeveloped: Lopes et al., 2017, 2018, Halko et al., 2011, Woolfe et al., 2008, Liberty et al., 2007
- **Our approach:** Estimate error via bootstrap.
  1. Randomized matrix multiplication (MM)
  2. Randomized least squares (LS)

Part I: Error estimation for matrix multiplication

Review of randomized MM

Consider two extremely large (non-random) matrices $A, B \in \mathbb{R}^{n\times d}$ with

$$d \ll n.$$

Suppose we want to compute

$$A^\top B.$$

Review of randomized MM

Consider two extremely large (non-random) matrices $A, B \in \mathbb{R}^{n \times d}$ with

$$d \ll n.$$

Suppose we want to compute

$$A^\top B.$$

Ordinary matrix multiplication has cost $\mathcal{O}(nd^2)$.

This cost can be a major bottleneck if matrix multiplication is used repeatedly in the analysis of large datasets.

Review of randomized MM

Recall that A and B each have a very large number of rows n.

One way to speed up the computation of $A^\top B$ is to use smaller matrices, called "sketches" $\tilde{A}$ and $\tilde{B}$, each having t rows, where $d \ll t \ll n$.

Review of randomized MM

Recall that A and B each have a very large number of rows n.

One way to speed up the computation of $A^\top B$ is to use smaller matrices, called "sketches" $\tilde{A}$ and $\tilde{B}$, each having t rows, where $d \ll t \ll n$.

Most commonly, the sketches are formed using a "sketching matrix" $S \in \mathbb{R}^{t \times n}$,

$$\tilde{A} = SA \quad \text{and} \quad \tilde{B} = SB.$$

Review of randomized MM

Recall that A and B each have a very large number of rows n.

One way to speed up the computation of $A^\top B$ is to use smaller matrices, called "sketches" $\tilde{A}$ and $\tilde{B}$, each having t rows, where $d \ll t \ll n$.

Most commonly, the sketches are formed using a "sketching matrix" $S \in \mathbb{R}^{t \times n}$,

$$\tilde{A} = SA \quad \text{and} \quad \tilde{B} = SB.$$

The sketching matrix is generated randomly, satisfying $\mathbb{E}[S^\top S] = \mathbf{I}_{n\times n}$. Hence,

$$\mathbb{E}[\tilde{A}^\top \tilde{B}] = \mathbb{E}[A^\top S^\top SB] = A^\top B.$$

(Many sophisticated types of S matrices have been proposed, but we omit these details.)

Review of randomized MM

Choosing the number of rows of S specifies a tradeoff between cost and accuracy.

Review of randomized MM

Choosing the number of rows of S specifies a tradeoff between cost and accuracy.

If $\mathbf{s}_1, \ldots, \mathbf{s}_t$ are the rows of $\sqrt{t}S$, then $S^\top S$ can be written as

$$S^\top S = \frac{1}{t} \sum_{i=1}^{t} \mathbf{s}_i \mathbf{s}_i^\top .$$

Review of randomized MM

Choosing the number of rows of S specifies a tradeoff between cost and accuracy.

If $\mathbf{s}_1, \ldots, \mathbf{s}_t$ are the rows of $\sqrt{t}S$, then $S^\top S$ can be written as

$$S^\top S = \frac{1}{t}\sum_{i=1}^{t} \mathbf{s}_i \mathbf{s}_i^\top .$$

Usually the rows $\mathbf{s}_1, \ldots, \mathbf{s}_t$ are (nearly) i.i.d., and so as t becomes large, LLN suggests $S^\top S \approx \mathbf{I}_{n\times n}$, giving

$$\tilde{A}^\top \tilde{B} \; = \; A^\top S^\top S B \; \approx \; A^\top B.$$

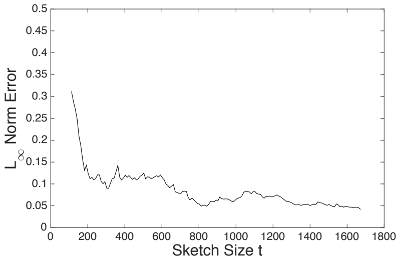
However, the cost of sketching grows proportionally with t.

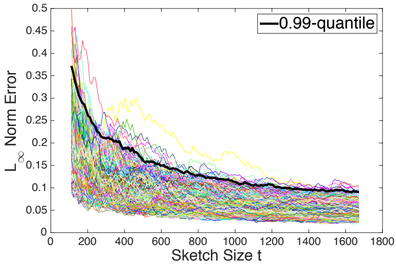
How does error depend on sketch size?

Consider the error

$$\varepsilon_t \ := \ \left\|\tilde{A}^\top \tilde{B} - A^\top B\right\|, \tag{1}$$

which is a random variable, since the sketches $\tilde{A}$ and $\tilde{B}$ are random.



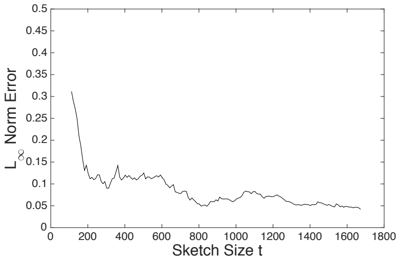


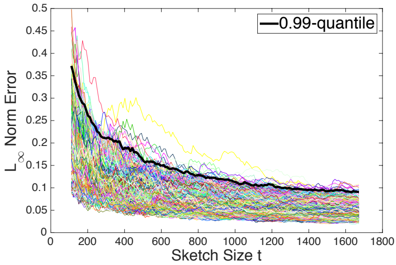
How does error depend on sketch size?

Consider the error

$$\varepsilon_t \;:=\; \left\|\tilde{A}^\top \tilde{B} - A^\top B\right\|, \tag{1}$$

which is a random variable, since the sketches $\tilde{A}$ and $\tilde{B}$ are random.



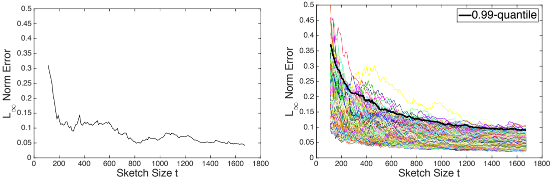


Note: The user is not able to see these curves in practice.

How does error depend on sketch size?

Let $q_{1-\alpha}(t)$ be the $(1-\alpha)$-quantile of ε_t.

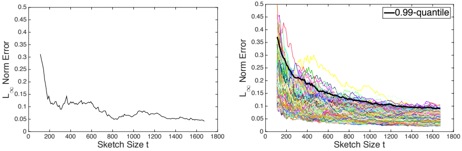
This is the tightest upper bound on ε_t that holds w.p. at least $1-\alpha$.



If the user knew the function $q_{1-\alpha}(t)$, they could know two things:

1. How accurate $\tilde{A}^\top \tilde{B}$ is likely to be for any given t.
2. How large t needs to be in order to achieve a given degree of accuracy.

Estimating the error quantiles



Problem formulation:

- We want to estimate the thick black curve $q_{1-\alpha}(t)$ from only one run of sketching. (i.e. just $\tilde{A}$ and $\tilde{B}$)
- It's not clear this is even possible, because $q_{1-\alpha}(t)$ reflects variation over many runs.
- We are computationally constrained: Any method we come up with should be cheap, so that it does not defeat the purpose of sketching.
- Also note that in practice, the user gets to see none of the curves above.

Intuition for bootstrap

- If we could generate samples of $\|\tilde{A}^\top \tilde{B} - A^\top B\|$, we would be done.

- For instance, if we could generate 100 samples, then we could take the 99th largest to estimate $q_{.99}(t)$.

- However, this is not possible since we don't know $A^\top B$.

- The bootstrap gives a way to generate "pseudo-samples" of $\|\tilde{A}^\top \tilde{B} - A^\top B\|$ using only the observed matrices $\tilde{A}$ and $\tilde{B}$.

Bootstrap procedure

Input: a positive integer m and the sketches $\tilde{A}$ and $\tilde{B}$.

For $l = 1, \ldots, m$ **do**

1. Draw a vector $(i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.

Bootstrap procedure

Input: a positive integer m and the sketches $\tilde{A}$ and $\tilde{B}$.

For $l = 1, \ldots, m$ **do**

1. Draw a vector $(i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.
2. Let $\tilde{A}^*$ and $\tilde{B}^*$ denote the matrices obtained by selecting the rows from $\tilde{A}$ and $\tilde{B}$ that are indexed by $(i_1, \ldots, i_t)$.

Bootstrap procedure

Input: a positive integer m and the sketches $\tilde{A}$ and $\tilde{B}$.

For $l = 1, \ldots, m$ **do**

1. Draw a vector $(i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.
2. Let $\tilde{A}^*$ and $\tilde{B}^*$ denote the matrices obtained by selecting the rows from $\tilde{A}$ and $\tilde{B}$ that are indexed by $(i_1, \ldots, i_t)$.
3. Compute the bootstrap sample $\varepsilon_l^* := \left\|(\tilde{A}^*)^\top(\tilde{B}^*) - \tilde{A}^\top \tilde{B}\right\|$.

Bootstrap procedure

Input: a positive integer m and the sketches $\tilde{A}$ and $\tilde{B}$.

For $l = 1, \ldots, m$ **do**

1. Draw a vector $(i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.
2. Let $\tilde{A}^*$ and $\tilde{B}^*$ denote the matrices obtained by selecting the rows from $\tilde{A}$ and $\tilde{B}$ that are indexed by $(i_1, \ldots, i_t)$.
3. Compute the bootstrap sample $\varepsilon_l^* := \left\|(\tilde{A}^*)^\top(\tilde{B}^*) - \tilde{A}^\top \tilde{B}\right\|$.

Return: $\widehat{q}_{1-\alpha}(t) \longleftarrow$ the $(1-\alpha)$-quantile of the values $\varepsilon_1^*, \ldots, \varepsilon_m^*$.

Speeding things up with extrapolation

The CLT indicates that $q_{1-\alpha}(t)$ should decay like $1/\sqrt{t}$.

Hence, we can bootstrap small "initial sketches" with t_0 rows, and then use

$$\widehat{q}_{1-\alpha}^{\,\text{ext}}(t) := \tfrac{\sqrt{t_0}}{\sqrt{t}}\,\widehat{q}_{1-\alpha}(t_0).$$

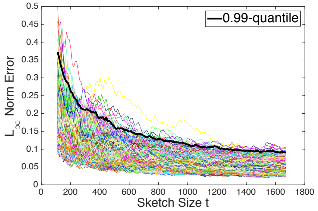
for $t \gg t_0$.

The CLT indicates that $q_{1-\alpha}(t)$ should decay like $1/\sqrt{t}$.

Hence, we can bootstrap small "initial sketches" with t_0 rows, and then use

$$\widehat{q}_{1-\alpha}^{\,\text{ext}}(t) := \frac{\sqrt{t_0}}{\sqrt{t}}\,\widehat{q}_{1-\alpha}(t_0).$$

for $t \gg t_0$.



The cost of the bootstrap

Existing sketching methods can compute $\tilde{A}^\top \tilde{B}$ with cost

$$\mathcal{O}(t \cdot d^2 + n \cdot d \cdot \log(t)).$$

The cost of the bootstrap

Existing sketching methods can compute $\tilde{A}^\top \tilde{B}$ with cost

$$\mathcal{O}(t \cdot d^2 + n \cdot d \cdot \log(t)).$$

The cost of applying the extrapolated bootstrap to $\tilde{A}$ and $\tilde{B}$ is independent of large dimension n,

$$\mathcal{O}(m \cdot t_0 \cdot d^2).$$

The cost of the bootstrap

Existing sketching methods can compute $\tilde{A}^\top \tilde{B}$ with cost

$$\mathcal{O}(t \cdot d^2 + n \cdot d \cdot \log(t)).$$

The cost of applying the extrapolated bootstrap to $\tilde{A}$ and $\tilde{B}$ is independent of large dimension n,

$$\mathcal{O}(m \cdot t_0 \cdot d^2).$$

Hence, the cost of bootstrapping does not outweigh sketching when the number of bootstrap samples satisifes

$$m = \mathcal{O}(\tfrac{t}{t_0} + \tfrac{n \log(t)}{d\, t_0}).$$

The cost of the bootstrap

Existing sketching methods can compute $\tilde{A}^\top \tilde{B}$ with cost

$$\mathcal{O}(t \cdot d^2 + n \cdot d \cdot \log(t)).$$

The cost of applying the extrapolated bootstrap to $\tilde{A}$ and $\tilde{B}$ is independent of large dimension n,

$$\mathcal{O}(m \cdot t_0 \cdot d^2).$$

Hence, the cost of bootstrapping does not outweigh sketching when the number of bootstrap samples satisifes

$$m = \mathcal{O}(\tfrac{t}{t_0} + \tfrac{n \log(t)}{d\, t_0}).$$

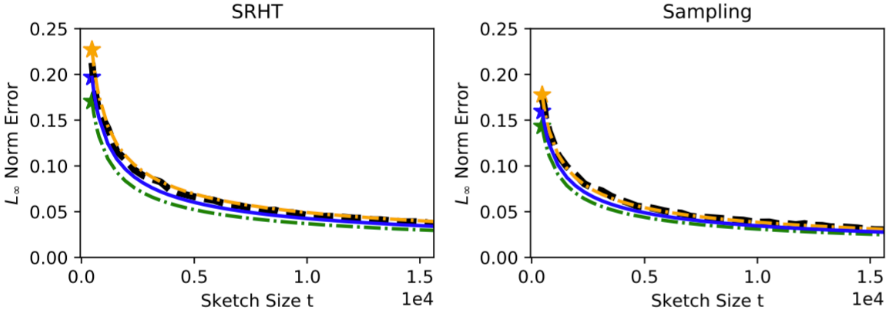
Empirically, merely $m = 20$ produces good results! (plots given later).

Also, we take $\frac{t}{t_0} \geq 20$ in many experiments.

Empirical performance

MNIST data: computing $A^\top A$ with $A \in \mathbb{R}^{60,000\times 780}$.

- initial sketch size $t_0 = 390$
- bootstrap samples $m = 20$



Comments on theoretical results

- It is possible to measure the quality of the estimator $\widehat{q}_{1-\alpha}(t)$ in terms of the Lévy-Prohorov metric between $\mathcal{L}(\sqrt{t}\varepsilon_t)$ and $\mathcal{L}(\sqrt{t}\varepsilon_t^*|S)$.

Comments on theoretical results

- It is possible to measure the quality of the estimator $\widehat{q}_{1-\alpha}(t)$ in terms of the Lévy-Prohorov metric between $\mathcal{L}(\sqrt{t}\varepsilon_t)$ and $\mathcal{L}(\sqrt{t}\varepsilon_t^*|S)$.

- Results hold for several choices of S:
  - i.i.d. sub-Gaussian entries
  - "length sampling"
  - sub-sampled randomized Hadamard transform

Comments on theoretical results

- It is possible to measure the quality of the estimator $\widehat{q}_{1-\alpha}(t)$ in terms of the Lévy-Prohorov metric between $\mathcal{L}(\sqrt{t}\varepsilon_t)$ and $\mathcal{L}(\sqrt{t}\varepsilon_t^*|S)$.

- Results hold for several choices of S:
  - i.i.d. sub-Gaussian entries
  - "length sampling"
  - sub-sampled randomized Hadamard transform

- Roughly speaking, our main results show that for ℓ_∞-norm error,

$$d_{LP}\big(\mathcal{L}(\sqrt{t}\varepsilon_t), \mathcal{L}(\sqrt{t}\varepsilon_t^*|S)\big) \to 0$$

  as long as

$$t \gg (\|A^\top A\|_\infty \|B^\top B\|_\infty)^3 \log(d)^5 \log(n)^2.$$

Comments on theoretical results

- It is possible to measure the quality of the estimator $\widehat{q}_{1-\alpha}(t)$ in terms of the Lévy-Prohorov metric between $\mathcal{L}(\sqrt{t}\varepsilon_t)$ and $\mathcal{L}(\sqrt{t}\varepsilon_t^*|S)$.

- Results hold for several choices of S:
  - i.i.d. sub-Gaussian entries
  - "length sampling"
  - sub-sampled randomized Hadamard transform

- Roughly speaking, our main results show that for ℓ_∞-norm error,

$$d_{LP}\big(\mathcal{L}(\sqrt{t}\varepsilon_t), \mathcal{L}(\sqrt{t}\varepsilon_t^*|S)\big) \to 0$$

  as long as

$$t \gg (\|A^\top A\|_\infty \|B^\top B\|_\infty)^3 \log(d)^5 \log(n)^2.$$

- Proof makes use of recent ideas on the "multiplier bootstrap" method in the high-dimensional statistics literature, as well as sharp constants in Rosenthal's inequality.

Part II: Error estimation for randomized least squares

Review of randomized LS

Consider a deterministic matrix $A \in \mathbb{R}^{n \times d}$ and vector $b \in \mathbb{R}^n$, with $n \gg d$.

The exact solution $x_{\text{opt}} := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \|Ax - b\|_2$ is too costly to compute.

Review of randomized LS

Consider a deterministic matrix $A \in \mathbb{R}^{n \times d}$ and vector $b \in \mathbb{R}^n$, with $n \gg d$.

The exact solution $x_{\text{opt}} := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \|Ax - b\|_2$ is too costly to compute.

We reduce problem with a random sketching matrix $S \in \mathbb{R}^{t \times n}$ with $d \ll t \ll n$.
Define $\tilde{A} := SA$ and $\tilde{b} := Sb$.

Review of randomized LS

Consider a deterministic matrix $A \in \mathbb{R}^{n\times d}$ and vector $b \in \mathbb{R}^n$, with $n \gg d$.

The exact solution $x_{\text{opt}} := \underset{x\in\mathbb{R}^d}{\text{argmin}} \|Ax - b\|_2$ is too costly to compute.

We reduce problem with a random sketching matrix $S \in \mathbb{R}^{t\times n}$ with $d \ll t \ll n$. Define $\tilde{A} := SA$ and $\tilde{b} := Sb$.

We focus on two particular randomized LS algorithms:

---

1. **Classic Sketch (CS).** (Drineas et al, 2006)

$$\tilde{x} := \underset{x\in\mathbb{R}^d}{\text{argmin}} \left\|\tilde{A}x - \tilde{b}\right\|_2$$

Review of randomized LS

Consider a deterministic matrix $A \in \mathbb{R}^{n \times d}$ and vector $b \in \mathbb{R}^n$, with $n \gg d$.

The exact solution $x_{\text{opt}} := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \|Ax - b\|_2$ is too costly to compute.

We reduce problem with a random sketching matrix $S \in \mathbb{R}^{t \times n}$ with $d \ll t \ll n$.
Define $\tilde{A} := SA$ and $\tilde{b} := Sb$.

We focus on two particular randomized LS algorithms:

---

1. **Classic Sketch (CS).** (Drineas et al, 2006)

$$\tilde{x} := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \left\| \tilde{A}x - \tilde{b} \right\|_2$$

---

2. **Iterative Hessian Sketch (IHS).** (Pilanci & Wainwright 2016)

$$\widehat{x}_{i+1} := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \left\{ \tfrac{1}{2} \|\tilde{A}(x - \widehat{x}_i)\|_2^2 + \langle A^\top (A\widehat{x}_i - b)\,,\, x \rangle \right\}, \qquad i = 1, \dots, k.$$

Problem formulation (error estimation)

We will estimate the errors of the random solutions $\tilde{x}$ and $\widehat{x}_k$ in terms of high-probability bounds.

Let $\|\cdot\|$ denote any norm on $\mathbb{R}^d$, and let $\alpha \in (0,1)$ be fixed.

Goal: Compute numerical estimates $q_{1-\alpha}(t)$ and $\widehat{q}_{1-\alpha}(t,k)$, such that the bounds

$$\|\tilde{x} - x_{\text{opt}}\| \leq \tilde{q}_{1-\alpha}(t)$$

$$\|\widehat{x}_k - x_{\text{opt}}\| \leq \widehat{q}_{1-\alpha}(t,k)$$

each hold with probability at least $1-\alpha$.

Intuition for the bootstrap

Key idea: Artificially generate a bootstrapped solution $\tilde{x}^*$ such that the fluctuations of $\tilde{x}^* - \tilde{x}$ are statistically similar to the fluctuations of $\tilde{x} - x_{\text{opt}}$.

In the "bootstrap world", $\tilde{x}$ plays the role of x_{opt}, and $\tilde{x}^*$ plays the role of $\tilde{x}$.

The bootstrap sample $\tilde{x}^*$ is the LS solution obtained by "perturbing" $\tilde{A}$ and $\tilde{b}$.

(The same intuition also applies to the IHS solution $\widehat{x}_k$.)

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

For: $l = 1, \ldots, m$ **do**

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

For: $l = 1, \ldots, m$ **do**

- Draw a random vector $\mathbf{i} := (i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

For: $l = 1, \ldots, m$ **do**

- Draw a random vector $\mathbf{i} := (i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.
- Form the matrix $\tilde{A}^* := \tilde{A}(\mathbf{i}, :)$, and vector $\tilde{b}^* := \tilde{b}(\mathbf{i})$.

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

For: $l = 1, \ldots, m$ **do**

- Draw a random vector $\mathbf{i} := (i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.
- Form the matrix $\tilde{A}^* := \tilde{A}(\mathbf{i}, :)$, and vector $\tilde{b}^* := \tilde{b}(\mathbf{i})$.
- Compute the vector
$$\tilde{x}^* := \operatorname*{argmin}_{x \in \mathbb{R}^d} \|\tilde{A}^* x - \tilde{b}^*\|_2,$$
and the scalar $\varepsilon_l^* := \|\tilde{x}^* - \tilde{x}\|$.

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

For: $l = 1, \ldots, m$ **do**

- Draw a random vector $\mathbf{i} := (i_1, \ldots, i_t)$ by sampling t numbers with replacement from $\{1, \ldots, t\}$.
- Form the matrix $\tilde{A}^* := \tilde{A}(\mathbf{i}, :)$, and vector $\tilde{b}^* := \tilde{b}(\mathbf{i})$.
- Compute the vector
$$\tilde{x}^* := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \|\tilde{A}^* x - \tilde{b}^*\|_2,$$
and the scalar $\varepsilon_l^* := \|\tilde{x}^* - \tilde{x}\|$.

Return: $\tilde{q}_{1-\alpha}(t) := \text{quantile}(\varepsilon_1^*, \ldots, \varepsilon_m^*; 1 - \alpha)$.

Algorithm (Error estimate for Classic Sketch)

Input: A positive integer m, and the sketches $\tilde{A}$, $\tilde{b}$, and $\tilde{x}$.

For: $l = 1, \dots, m$ **do**

- Draw a random vector $\mathbf{i} := (i_1, \dots, i_t)$ by sampling t numbers with replacement from $\{1, \dots, t\}$.
- Form the matrix $\tilde{A}^* := \tilde{A}(\mathbf{i}, :)$, and vector $\tilde{b}^* := \tilde{b}(\mathbf{i})$.
- Compute the vector
$$\tilde{x}^* := \underset{x \in \mathbb{R}^d}{\operatorname{argmin}} \|\tilde{A}^* x - \tilde{b}^*\|_2,$$
  and the scalar $\varepsilon_l^* := \|\tilde{x}^* - \tilde{x}\|$.

Return: $\tilde{q}_{1-\alpha}(t) := \text{quantile}(\varepsilon_1^*, \dots, \varepsilon_m^*; 1 - \alpha)$.

Note: A similar algorithm works for IHS.

Computational cost

1. Cost of error estimation is independent of large dimension n, whereas most randomized LS algorithms scale linearly in n.

Computational cost

1. Cost of error estimation is independent of large dimension n, whereas most randomized LS algorithms scale linearly in n.

2. In practice, as few as $m = 20$ bootstrap samples are sufficient.

Computational cost

1. Cost of error estimation is independent of large dimension n, whereas most randomized LS algorithms scale linearly in n.

2. In practice, as few as $m = 20$ bootstrap samples are sufficient.

3. Implementation is embarrassingly parallel.
   (Per-processor cost is $\mathcal{O}(td^2)$, with modest communication.)

Computational cost

1. Cost of error estimation is independent of large dimension n, whereas most randomized LS algorithms scale linearly in n.

2. In practice, as few as $m = 20$ bootstrap samples are sufficient.

3. Implementation is embarrassingly parallel.
(Per-processor cost is $\mathcal{O}(td^2)$, with modest communication.)

4. Bootstrap computations have free warm starts.

Computational cost

1. Cost of error estimation is independent of large dimension n, whereas most randomized LS algorithms scale linearly in n.

2. In practice, as few as $m = 20$ bootstrap samples are sufficient.

3. Implementation is embarrassingly parallel.
(Per-processor cost is $\mathcal{O}(td^2)$, with modest communication.)

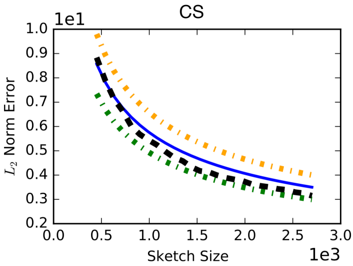
4. Bootstrap computations have free warm starts.

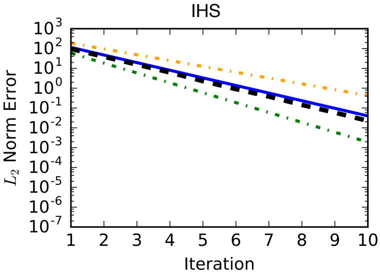
5. Error estimates can be extrapolated (similar to MM context).

Empirical performance

'YearPredictionMSD' data from LIBSVM: $n \approx 5 \times 10^5$ and $d = 90$

- **CS:** fix initial sketch size $t_0 = 5d$ and extrapolate on $t \gg t_0$
- **IHS:** fix sketch size $t = 10d$ and extrapolate on number of iterations
- bootstrap samples $m = 20$





Comments on theoretical results

- Main result shows that under certain asymptotic assumptions

$$\liminf_{n\to\infty} \mathbb{P}\Big(\|\tilde{x} - x_{\text{opt}}\| \leq \tilde{q}_{1-\alpha}(t)\Big) \;\geq 1-\alpha,$$

and similarly for $\widehat{q}_{1-\alpha}(t,k)$ with regard to IHS.

Comments on theoretical results

- Main result shows that under certain asymptotic assumptions

$$\liminf_{n\to\infty} \mathbb{P}\Big(\|\tilde{x} - x_{\text{opt}}\| \leq \tilde{q}_{1-\alpha}(t) \Big) \ \geq 1 - \alpha,$$

  and similarly for $\widehat{q}_{1-\alpha}(t,k)$ with regard to IHS.

- Result holds for any choice of norm $\|\cdot\|$, provided
  - $(n,t) \to \infty$ with d held fixed
  - S has i.i.d. entries.
  - the matrix $A^\top A$ and $A^\top b$ are "stable" as $n \to \infty$

Comments on theoretical results

- Main result shows that under certain asymptotic assumptions

$$\liminf_{n\to\infty} \mathbb{P}\Big(\|\tilde{x} - x_{\text{opt}}\| \leq \tilde{q}_{1-\alpha}(t) \Big) \;\geq 1 - \alpha,$$

  and similarly for $\widehat{q}_{1-\alpha}(t,k)$ with regard to IHS.

- Result holds for any choice of norm $\|\cdot\|$, provided
  - $(n,t) \to \infty$ with d held fixed
  - S has i.i.d. entries.
  - the matrix $A^\top A$ and $A^\top b$ are "stable" as $n \to \infty$

- The most difficult part of the proof concerns the IHS algorithm which is iterative. This leads to analyzing the distribution of $\widehat{x}_k$ conditionally on the previous iterates, and this requires approximations that hold "uniformly" over past iterates.

Summary

- Bootstrapping is a flexible approach to error estimation that can be adapted to a variety of RandNLA algorithms.

Summary

- Bootstrapping is a flexible approach to error estimation that can be adapted to a variety of RandNLA algorithms.
- This provides a practical alternative to worst-case error bounds, and adapts to the input at hand.

Summary

- Bootstrapping is a flexible approach to error estimation that can be adapted to a variety of RandNLA algorithms.
- This provides a practical alternative to worst-case error bounds, and adapts to the input at hand.
- The cost of bootstrapping does not outweigh the benefits of sketching.

Summary

- Bootstrapping is a flexible approach to error estimation that can be adapted to a variety of RandNLA algorithms.
- This provides a practical alternative to worst-case error bounds, and adapts to the input at hand.
- The cost of bootstrapping does not outweigh the benefits of sketching.
- The bootstrap computations are highly scalable – since they do not depend on large dimension n, are easily parallelized, and can be extrapolated.

Summary

- Bootstrapping is a flexible approach to error estimation that can be adapted to a variety of RandNLA algorithms.
- This provides a practical alternative to worst-case error bounds, and adapts to the input at hand.
- The cost of bootstrapping does not outweigh the benefits of sketching.
- The bootstrap computations are highly scalable – since they do not depend on large dimension n, are easily parallelized, and can be extrapolated.
- Numerical performance is encouraging, and is supported by theoretical guarantees.

Recent work

- *A Bootstrap Method for Error Estimation in Randomized Matrix Multiplication*
  arxiv:1708.01945
- *Error Estimation for Randomized Least-Squares Algorithms via the Bootstrap*
  ICML 2018, and arxiv:1803.08021
- *Estimating the Algorithmic Variance of Randomized Ensembles via the Bootstrap*
  The Annals of Statistics (to appear) 2018

Thanks for your attention and NSF DMS-1613218 for partial support.