Lecture 3: Trace estimation, low-rank matrix approximation

School of Mathematical Sciences, Xiamen University

1. Implicit trace estimation problem

- Suppose that $\mathbf{A}$ is a square matrix, accessible via matrix-vector products (*matvecs*): $\mathbf{x} \mapsto \mathbf{A}\mathbf{x}$. The problem is to estimate the trace

$$\mathrm{tr}(\mathbf{A}) = \sum_{i=1}^{n} a_{ii}$$

  using as few matvecs as possible.

- Applications of trace estimation:

  Smoothing splines: $\mathrm{tr}(\mathbf{A}^{-1})$;

  Social network analysis: $\mathrm{tr}(\mathbf{A}^3)$;

  Chemical graph theory: $\mathrm{tr}(\exp(\mathbf{A}))$;

  Quantum statistical mechanics: $\mathrm{tr}(\exp(-\beta\mathbf{H}))$, $\mathrm{tr}(\mathbf{H}\exp(-\beta\mathbf{H}))$

- The procedure using $a_{ii} = \mathbf{e}_i^\top \mathbf{A}\mathbf{e}_i$ provides an exact value for the trace, but it requires fully n matvecs. The cost may be prohibitive when n is large.

1.1 Monte Carlo approximation algorithm

- Suppose that $\mathbf{x} \in \mathbb{R}^n$ is a random vector with *isotropic* distribution:

$$\mathbb{E}(\mathbf{x}\mathbf{x}^\top) = \mathbf{I}_n.$$

For examples:

(1) standard normal $\mathbf{x} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_n)$;

(2) uniform on a sphere $\mathbf{x} \sim \text{uniform}(\sqrt{n}\mathbb{S}^{n-1})$;

(3) independent signs $\mathbf{x} \sim \text{uniform}(\{\pm 1\}^n)$.

Exercise: prove that random vectors in (2) and (3) are isotropic.

- Taking one matvec with the matrix $\mathbf{A}$, we can form the scalar random variable

$$Y := \mathbf{x}^\top(\mathbf{A}\mathbf{x}),$$

which is an *unbiased estimator* for the trace of the matrix:

$$\mathbb{E}(Y) = \mathbb{E}(\text{tr}(\mathbf{x}^\top\mathbf{A}\mathbf{x})) = \mathbb{E}(\text{tr}(\mathbf{A}\mathbf{x}\mathbf{x}^\top)) = \text{tr}(\mathbf{A}\mathbb{E}(\mathbf{x}\mathbf{x}^\top)) = \text{tr}(\mathbf{A}).$$

- Although the simple trace estimator Y is correct on average, its fluctuations may be large in comparison with $\mathrm{tr}(\mathbf{A})$. A standard remedy is to *average independent copies* of the simple trace estimator. Taking s matvecs with the matrix $\mathbf{A}$, we can form the Monte Carlo trace estimator:

$$\widehat{\mathrm{tr}}_s := \frac{1}{s}\sum_{i=1}^{s} Y_i, \quad \text{where} \quad Y_i \sim Y \quad \text{i.i.d.}$$

Algorithm: Trace estimation: Simplest Monte Carlo method.

---

for $i = 1, 2, \ldots, s$ **do**
  Sample isotropic i.i.d. random vector $\mathbf{x}_i \in \mathbb{R}^n$
  $Y_i = \mathbf{x}_i^\top (\mathbf{A}\mathbf{x}_i)$
end
$\widehat{\mathrm{tr}}_s = s^{-1}\sum_{i=1}^{s} Y_i$
$\widehat{v}_s = (s(s-1))^{-1}\sum_{i=1}^{s}(Y_i - \widehat{\mathrm{tr}}_s)^2$

Proposition 1 (Monte Carlo trace estimation)

Assume that $\mathbf{A}$ *is a nonzero spsd matrix. Form the Monte Carlo trace estimator* $\widehat{\mathrm{tr}}_s$ *with* s *i.i.d.* $\mathbf{x}_i \sim \mathrm{uniform}(\{\pm 1\}^n)$. *For any* $\varepsilon > 0$, *the estimator satisfies the probability bound*

$$\mathbb{P}\{|\widehat{\mathrm{tr}}_s - \mathrm{tr}(\mathbf{A})| \geq \varepsilon \cdot \mathrm{tr}(\mathbf{A})\} \leq \frac{2\|\mathbf{A}\|_2}{s\varepsilon^2 \cdot \mathrm{tr}(\mathbf{A})} \leq \frac{2}{s\varepsilon^2}.$$

Proof. We have (exercise)

$$\mathbb{V}\mathrm{ar}(Y) = 2\sum_{i\neq j} a_{ij}^2 \leq 2\|\mathbf{A}\|_{\mathrm{F}}^2 \leq 2\|\mathbf{A}\|_2 \mathrm{tr}(\mathbf{A}), \qquad \frac{\mathbb{V}\mathrm{ar}(\widehat{\mathrm{tr}}_s)}{\mathrm{tr}(\mathbf{A})^2} \leq \frac{2\|\mathbf{A}\|_2}{s \cdot \mathrm{tr}(\mathbf{A})}.$$

An application of Chebyshev's inequality yields

$$\mathbb{P}\{|\widehat{\mathrm{tr}}_s - \mathrm{tr}(\mathbf{A})| \geq \varepsilon \cdot \mathrm{tr}(\mathbf{A})\} \leq \frac{\mathbb{E}|\widehat{\mathrm{tr}}_s - \mathrm{tr}(\mathbf{A})|^2}{\varepsilon^2 \cdot \mathrm{tr}(\mathbf{A})^2} = \frac{\mathbb{V}\mathrm{ar}(\widehat{\mathrm{tr}}_s)}{\varepsilon^2 \cdot \mathrm{tr}(\mathbf{A})^2} \leq \frac{2\|\mathbf{A}\|_2}{s\varepsilon^2 \cdot \mathrm{tr}(\mathbf{A})}.$$

This completes the proof. □

- A surprising feature of the bound of Proposition 1 is that we can reliably estimate the trace of an spsd matrix within a factor of two (i.e., $\varepsilon = 0.5$) using just a constant number of samples, say, $s = 16$. This holds true regardless of the dimension of the matrix!
- The probability bound is nontrivial as soon as the number s of samples satisfies $s \geq 2\|\mathbf{A}\|_2/(\varepsilon^2 \cdot \mathrm{tr}(\mathbf{A}))$. The expression for the number s of samples scales with ε^{-2}. The scaling is an inexorable consequence of the central limit theorem, rather than an artifact of the analysis. This is the curse of Monte Carlo estimation. Stronger probability bounds can be obtained by using exponential concentration inequalities in place of Chebyshev's inequality. But the basic scaling for the number s of samples remains the same.
- The best distribution for the test vector depends on the application. The independent sign distribution and the spherical distribution are both excellent candidates. The Gaussian distribution leads to trace estimates with higher variance, so it should usually be avoided.

- Unfortunately, $\|\mathbf{A}\|_2/\mathrm{tr}(\mathbf{A})$ is usually not available. How can we determine the number s of samples we need?
- The sample variance $\widehat{v}_s$ is an unbiased estimator for the variance of the trace estimator $\widehat{\mathrm{tr}}_s$. If the distribution of the test vector $\mathbf{x}$ has four finite moments ($\mathbb{E}(\|\mathbf{x}\|_2^4) < \infty$), then we have the relation (exercise)

$$\mathbb{E}(\widehat{v}_s) = \mathbb{V}\mathrm{ar}(\widehat{\mathrm{tr}}_s).$$

  Therefore, we can employ the sample variance to construct a stopping rule for the trace estimator. For a spsd input matrix $\mathbf{A}$, to attain a relative error tolerance $\varepsilon > 0$, we can continue drawing samples until

$$\widehat{v}_s \leq (\varepsilon \cdot \widehat{\mathrm{tr}}_s)^2.$$

- Ethan N. Epperly, Joel A. Tropp, and Robert J. Webber

  XTrace: making the most of every sample in stochastic trace estimation, SIAM J. Matrix Anal. Appl. 45 (2024), no. 1, 1–23.

2. Low-rank matrix approximation problem

- Suppose $\mathbf{B}$ is a rectangular matrix, accessible via matvecs $\mathbf{x} \mapsto \mathbf{B}\mathbf{x}$ and $\mathbf{y} \mapsto \mathbf{B}^*\mathbf{y}$. The task is to produce a low-rank approximation of $\mathbf{B}$ that is competitive with a best approximation of similar rank.
- The best rank-k approximation is unique if and only if $\sigma_k > \sigma_{k+1}$:

$$\min_{\mathrm{rank}(\mathbf{M}) \le k} \|\mathbf{B} - \mathbf{M}\|_{\mathrm{F}}^2 = \|\mathbf{B} - \mathbf{U}_k \mathbf{U}_k^* \mathbf{B}\|_{\mathrm{F}}^2 = \sum\nolimits_{i>k} \sigma_i^2,$$

  where $\mathbf{U}_k$ is the matrix consisting of the leading k left singular vectors. Cost: $\mathcal{O}(mnp)$ where $p = \min(m, n)$.
- Let $s = k + l$ for a small natural number l. For a tolerance $\varepsilon > 0$, we seek a rank-s approximation $\widehat{\mathbf{B}}_s$ that competes with the best rank-k approximation:

$$\|\mathbf{B} - \widehat{\mathbf{B}}_s\|_{\mathrm{F}}^2 \le (1+\varepsilon)\|\mathbf{B} - \mathbf{U}_k \mathbf{U}_k^* \mathbf{B}\|_{\mathrm{F}}^2 = (1+\varepsilon) \sum\nolimits_{i>k} \sigma_i^2.$$

2.1 Randomized SVD: Intuition

- Draw a standard normal test vector $\mathbf{w} \in \mathbb{R}^n$. we have

$$\mathbf{Bw} = \sum_{i=1}^{p} \sigma_i \mathbf{u}_i (\mathbf{v}_i^* \mathbf{w}) := \sum_{i=1}^{p} \sigma_i \mathbf{u}_i w_i.$$

  The component $w_i := \mathbf{v}_i^* \mathbf{w}$ of the random vector along the ith right singular vector follows a standard normal distribution, and the components $(w_i, i = 1, \ldots, p)$ compose an independent family.

- On average, $\mathbb{E}(w_i^2) = 1$. Therefore, the image $\mathbf{Bw}$ tends to align with the left singular vectors associated with large singular values.

- By repeating this process with a statistically independent family $(\mathbf{w}^{(j)} : j = 1, \ldots, s)$ of random test vectors, we can obtain a family $(\mathbf{Bw}^{(j)} : j = 1, \ldots, s)$ of vectors whose span contains most of range$(\mathbf{U}_k)$. The number $s = k + l$ of test vectors needs to be a bit larger than the target rank k to obtain coverage of the subspace with high probability.

2.2 Randomized SVD: Algorithm (cost $\mathcal{O}(smn)$)

- For a rank parameter s, we draw a random test matrix:

$$\mathbf{\Omega} = \begin{bmatrix} \mathbf{w}^{(1)} & \cdots & \mathbf{w}^{(s)} \end{bmatrix} \quad \text{where} \quad \mathbf{w}^{(j)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_n) \quad \text{i.i.d.}$$

- We obtain $\mathbf{Y} := \mathbf{B\Omega}$. The orthogonal projector $\mathbf{P_Y}$ onto range$(\mathbf{Y})$ serves as a proxy for the ideal projector $\mathbf{U}_k\mathbf{U}_k^*$. Computationally,

$$\mathbf{P_Y} := \mathbf{QQ}^* \quad \text{where} \quad \mathbf{Q} := \text{orth}(\mathbf{Y}).$$

  The function orth returns an orthonormal basis and costs $\mathcal{O}(s^2m)$.

- Finally, we report the approximation $\widehat{\mathbf{B}}_s$ in factored form:

$$\widehat{\mathbf{B}}_s := \mathbf{P_Y B} = \mathbf{Q}(\mathbf{Q}^*\mathbf{B}).$$

- If desired, we can report the SVD of the approximation after a small amount of additional work ($\mathcal{O}(s^2n)$):

$$\widehat{\mathbf{B}}_s = (\mathbf{Q}\widehat{\mathbf{U}}_0)\widehat{\mathbf{\Sigma}}\widehat{\mathbf{V}}^* \quad \text{where} \quad (\widehat{\mathbf{U}}_0, \widehat{\mathbf{\Sigma}}, \widehat{\mathbf{V}}) = \text{svd}(\mathbf{Q}^*\mathbf{B}).$$

Algorithm: Randomized SVD.

$$\begin{aligned}
&\boldsymbol{\Omega} = \mathrm{randn}(n, s)\\
&\mathbf{Y} = \mathbf{B}\boldsymbol{\Omega}\\
&\mathbf{Q} = \mathrm{orth}(\mathbf{Y})\\
&\mathbf{C} = \mathbf{Q}^*\mathbf{B}\\
&(\widehat{\mathbf{U}}_0, \widehat{\boldsymbol{\Sigma}}, \widehat{\mathbf{V}}) = \mathrm{svd}(\mathbf{C})\\
&\widehat{\mathbf{U}} = \mathbf{Q}\widehat{\mathbf{U}}_0
\end{aligned}$$

Theorem 2

Consider a matrix $\mathbf{B} \in \mathbb{R}^{m\times n}$, and fix the target rank $k \le p$. When $s \ge k+2$, the randomized SVD method produces a random rank-s approximation $\widehat{\mathbf{B}}_s$ that satisfies

$$\mathbb{E}\|\mathbf{B} - \widehat{\mathbf{B}}_s\|_{\mathrm{F}}^2 \le \left(1 + \frac{k}{s-k-1}\right) \sum_{i>k} \sigma_i^2(\mathbf{B}).$$

Proof. See A. Kireeva and J. A. Tropp, arXiv:2402.17873, 2024. □

2.3 Randomized subspace iteration

Algorithm: Randomized subspace iteration.

$\mathbf{X}_0 = \text{randn}(n, s)$
for $t = 1, 2, \ldots, T$
  $\mathbf{Q}_t := \text{orth}(\mathbf{B}\mathbf{X}_{t-1})$
  $\mathbf{X}_t := \mathbf{B}^*\mathbf{Q}_t$
end
$\widehat{\mathbf{B}}_s := \mathbf{Q}_T\mathbf{X}_T^*$

- Randomized SVD is the special case of this algorithm with $T = 1$.
- Randomized subspace iteration produces approximations

$$\widehat{\mathbf{B}}_s = \mathbf{Q}_t(\mathbf{Q}_t^*\mathbf{B}) \quad \text{where} \quad \mathbf{Q}_t = \text{orth}((\mathbf{B}\mathbf{B}^*)^{t-1}\mathbf{B}\boldsymbol{\Omega}) \quad \text{for} \quad t = 1, 2, \ldots$$

- Much as the block power method drives its iterates toward the leading eigenspace, subspace iteration drives range($\mathbf{Q}_t$) so that it aligns with range($\mathbf{U}_r$), the leading left singular subspace of $\mathbf{B}$.

3. Low-rank spsd approximation from entries

Consider an spsd matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ that we access via entry evaluations: $(j, k) \mapsto a_{jk}$. The task is to produce a low-rank spsd approximation of $\mathbf{A}$ using as few as entry evaluations.

3.1 Column Nyström approximation

- Given a list $S \subseteq \{1, 2, 3, ..., n\}$ of column indices, the column Nyström approximation:

$$\mathbf{A}_{\langle S \rangle} := \mathbf{A}(:, S)\mathbf{A}(S, S)^{\dagger}\mathbf{A}(S, :).$$

- The column Nyström approximation has several remarkable properties: (1) $\text{range}(\mathbf{A}_{\langle S \rangle}) = \text{range}(\mathbf{A}(:, S))$; (2) $\mathbf{0} \preccurlyeq \mathbf{A}_{\langle S \rangle} \preccurlyeq \mathbf{A}$.
- Our goal is to find a set S of s columns that make the error

$$\|\mathbf{A} - \mathbf{A}_{\langle S \rangle}\|_* = \text{tr}(\mathbf{A} - \mathbf{A}_{\langle S \rangle})$$

as small as possible.

3.2 Pivoted partial Cholesky

- Set $\mathbf{A}_0 := \mathbf{A}$ and $\widehat{\mathbf{A}}_0 := \mathbf{0}$.

  At each step $t = 1, 2, \ldots, s$, select $i_t \in \{1, 2, \ldots, n\}$, and update

$$\mathbf{A}_t := \mathbf{A}_{t-1} - \frac{\mathbf{A}_{t-1}(:, i_t)\mathbf{A}_{t-1}(i_t, :)}{\mathbf{A}_{t-1}(i_t, i_t)};$$

$$\widehat{\mathbf{A}}_t := \widehat{\mathbf{A}}_{t-1} + \frac{\mathbf{A}_{t-1}(:, i_t)\mathbf{A}_{t-1}(i_t, :)}{\mathbf{A}_{t-1}(i_t, i_t)}.$$

  Exercise: Prove the following results: (1) $\mathbf{A}_t + \widehat{\mathbf{A}}_t = \mathbf{A}$; (2)

$$\mathrm{diag}(\mathbf{A}_t) = \mathrm{diag}(\mathbf{A}_{t-1}) - \frac{1}{\mathbf{A}_{t-1}(i_t, i_t)} |\mathbf{A}_{t-1}(:, i_t)|^2.$$

Proposition 3

Suppose that we apply the pivoted partial Cholesky algorithm to an spsd matrix $\mathbf{A}$*, and we select* i_t *from* S *in any order. Then* $\widehat{\mathbf{A}}_{|S|} = \mathbf{A}_{\langle S \rangle}$*, where* $|S|$ *denotes the number of elements of the set* S*.*

The proof is left as an exercise.

3.3 Pivoted partial Cholesky: evaluating fewer entries

- Set $\mathbf{F}_0 := \mathbf{0}$.

  At each step $t = 1, 2, \ldots, s$, select $i_t \in \{1, 2, \ldots, n\}$ and set

$$\mathbf{c}_t := \mathbf{A}(:, i_t) - \mathbf{F}_{t-1}(\mathbf{F}_{t-1}(i_t, :))^*.$$

  Update $\mathbf{F}_t := \begin{bmatrix} \mathbf{F}_{t-1} & \mathbf{c}_t / \sqrt{\mathbf{c}_t(i_t)} \end{bmatrix}$.

  Exercise: Prove that $\widehat{\mathbf{A}}_t = \mathbf{F}_t \mathbf{F}_t^*$ for $t = 0, 1, 2, \ldots, s$.

3.4 Pivot selection rules

- Uniform random pivoting: $i_t \sim \text{uniform}\{1, 2, \ldots, n\}$.

  Assumption: data points represent an i.i.d. sample from a population, so one is just as good as another.

- Greedy pivoting: $i_t \in \text{argmax}\{\mathbf{A}_{t-1}(i, i) : i = 1, 2, \ldots, n\}$.

  Note that $\mathbf{A}_t(i_t, i_t) = 0$.

- Importance sampling pivoting: $\mathbb{P}\{i_t = j\} = \mathbf{A}_{t-1}(j, j) / \text{tr}(\mathbf{A}_{t-1})$.

  Balance between uniform random and greedy.

3.5 Randomly pivoted partial Cholesky

Algorithm: Randomly pivoted partial Cholesky.

$\mathbf{F} = \text{zeros}(n, s)$ (Preallocation)
$\mathbf{d} = \text{diag}(\mathbf{A})$
for $t = 1, 2, \ldots, s$
  Sample $i_t \sim \mathbf{d} / \sum_{j=1}^{n} \mathbf{d}(j)$
  $\mathbf{c} = \mathbf{A}(:, i_t) - \mathbf{F}(\mathbf{F}(i_t, :))^*$
  $\mathbf{F}(:, t) = \mathbf{c} / \sqrt{\mathbf{c}(i_t)}$
  $\mathbf{d} = \mathbf{d} - |\mathbf{F}(:, t)|^2$
  $\mathbf{d} = \max\{\mathbf{d}, \mathbf{0}\}$ (Improve numerical stability)
  Stop when $\sum_{j=1}^{n} \mathbf{d}(j) < \eta \cdot \text{tr}(\mathbf{A})$ (Optional)
end

- To produce a rank-s approximation, the algorithm only requires $(s+1)n - s$ entries of $\mathbf{A}$: its diagonal and the s pivot columns.

- Define the expected residual map: $\Phi(\mathbf{A}) := \mathbb{E}(\mathbf{A}_1)$. This function measures the average progress that we make after one step of the algorithm. A quick calculation yields a formula for the expected residual map:

$$\begin{aligned}\Phi(\mathbf{A}) &= \sum_{j=1}^{n} \left[\mathbf{A} - \frac{\mathbf{A}(:,j)\mathbf{A}(j,:)}{\mathbf{A}(j,j)}\right] \frac{\mathbf{A}(j,j)}{\mathrm{tr}(\mathbf{A})} \\ &= \mathbf{A} - \frac{1}{\mathrm{tr}(\mathbf{A})} \sum_{j=1}^{n} \mathbf{A}(:,j)\mathbf{A}(j,:) = \mathbf{A} - \frac{\mathbf{A}^2}{\mathrm{tr}(\mathbf{A})}.\end{aligned}$$

As a result, we have

$$\mathbb{E}(\mathrm{tr}(\mathbf{A}_1)) = \mathrm{tr}(\mathbb{E}(\mathbf{A}_1)) = \left(1 - \frac{\mathrm{tr}(\mathbf{A}^2)}{(\mathrm{tr}(\mathbf{A}))^2}\right) \mathrm{tr}(\mathbf{A}) \leq \frac{n-1}{n} \mathrm{tr}(\mathbf{A}).$$

In each iteration, we decrease the expected trace of the residual on average.

- The best rank-k approximation in $\|\cdot\|_*$:

$$\min_{\mathrm{rank}(\mathbf{M})\le k} \|\mathbf{A}-\mathbf{M}\|_* = \sum_{j>k} \sigma_j(\mathbf{A}).$$

- Fix a comparison rank k and a tolerance $\varepsilon > 0$. Randomly pivoted Cholesky produces an approximation $\widehat{\mathbf{A}}_s$ that attains the error bound

$$\mathbb{E}(\|\mathbf{A}-\widehat{\mathbf{A}}_s\|_\star) \le (1+\varepsilon)\sum_{j>k}\sigma_j(\mathbf{A})$$

after selecting s columns where

$$s \ge \frac{k}{\varepsilon} + k\log\left(\frac{1}{\varepsilon\eta}\right) \quad \text{and} \quad \eta := \frac{1}{\mathrm{tr}(\mathbf{A})}\sum_{j>k}\sigma_j(\mathbf{A}).$$