ORIE 7391: Faster: Algorithmic Ideas for Speeding Up Optimization

Randomized Numerical Linear Algebra

Professor Udell

Operations Research and Information Engineering
Cornell

February 2, 2022

Outline

Why randomize?

trade time for accuracy when you're solving a subproblem

Basic building block: random matvec

given

- operator $A : \mathbf{R}^n \to \mathbf{R}^m$
- random test vector $\omega \in \mathbf{R}^n$

form

$$A\omega$$

Basic building block: random matvec

given

- operator $A : \mathbf{R}^n \to \mathbf{R}^m$
- random test vector $\omega \in \mathbf{R}^n$

form

$$A\omega$$

what random vectors?

- Gaussian $\omega \sim \mathcal{N}(0, I_n)$
- Rademacher $\omega \sim \mathcal{U}\{\pm 1\}^n$
- sparse
- fast Johnson-Lindenstrauss transform
- subsampled randomized fourier transform
- ...

Outline

Trace computation

compute $\mathbf{tr}(A)$ without randomization:

Trace computation

compute $\mathbf{tr}(A)$ without randomization:

- $\sum_{i=1}^{n} A_{ii}$ requires access to individual entries of A, $O(n)$ flops

Trace computation

compute $\mathbf{tr}(A)$ without randomization:

- $\sum_{i=1}^{n} A_{ii}$ requires access to individual entries of A, $O(n)$ flops
- $\sum_{i=1}^{n} e_i^T A e_i$ can be computed with n matvecs + $O(n)$ flops

Trace computation

compute $\mathbf{tr}(A)$ without randomization:

- $\sum_{i=1}^{n} A_{ii}$ requires access to individual entries of A, $O(n)$ flops
- $\sum_{i=1}^{n} e_i^T A e_i$ can be computed with n matvecs + $O(n)$ flops

can we do it with fewer matvecs?

Trace estimation

suppose $\omega \in \mathbf{R}^n$ is isotropic: $\mathbb{E}\omega\omega^T = I$. then

$$X = \omega^T A \omega \quad \text{satisfies} \quad \mathbb{E}X = \mathbf{tr}(A)$$

Trace estimation

suppose $\omega \in \mathbf{R}^n$ is isotropic: $\mathbb{E}\omega\omega^T = I$. then

$$X = \omega^T A \omega \quad \text{satisfies} \quad \mathbb{E}X = \mathbf{tr}(A)$$

proof:

$$\begin{aligned} \omega^T A \omega &= \mathbf{tr}(\omega^T A \omega) = \mathbf{tr}(A \omega \omega^T) \\ \mathbb{E}\omega^T A \omega &= \mathbb{E}\,\mathbf{tr}(A \omega \omega^T) = \mathbf{tr}(AI) = \mathbf{tr}(A) \end{aligned}$$

Improve variance by averaging

estimate trace by averaging over many iid copies:

$$\bar{X}_k = \frac{1}{k} \sum_{i=1^k} X_i \quad \text{where} \quad X_i \sim X \text{ are iid}$$

cost: k random n-vectors + k matvecs + $O(kn)$ arithmetic

Improve variance by averaging

estimate trace by averaging over many iid copies:

$$\bar{X}_k = \frac{1}{k} \sum_{i=1^k} X_i \quad \text{where} \quad X_i \sim X \text{ are iid}$$

cost: k random n-vectors + k matvecs + $O(kn)$ arithmetic

$$\mathbb{E}[\bar{X}_k] = \mathbf{tr}(A) \qquad \mathrm{Var}[\bar{X}_k] = \frac{1}{k}\mathrm{Var}[X]$$

Outline

Approximate eigenvectors

Definition

For a symmetric matrix $M \in \mathbf{S}^n_+$, we say a unit vector v is an ε-**approximate maximum eigenvector** if

$$v^* M v \geq (1 - \varepsilon)\lambda_{\max}(M).$$

Approximate eigenvectors

Definition

For a symmetric matrix $M \in \mathbf{S}_+^n$, we say a unit vector v is an ε-**approximate maximum eigenvector** if

$$v^* M v \geq (1 - \varepsilon)\lambda_{\max}(M).$$

how to find approximate maximum eigenvector (efficiently)?

- (-) Krylov methods (*e.g.*, ARPACK `eigs`)
  - unstable, hard to control precision
- (+) power method
  - converges in $\mathcal{O}(\varepsilon^{-1})$ iterations, needs $\mathcal{O}(n)$ storage
- (+) randomized Lanczos method
  - converges in $q = \mathcal{O}(\varepsilon^{-1/2})$ iterations, needs $\mathcal{O}(nq)$ storage
  - (or can use $\mathcal{O}(n)$ storage by running it twice)

Approximate eigenvectors: power method

Algorithm ApproxMaxEvec via randomized power method

Input: $M \in \mathbf{S}_n$, and maxiters q

Output: Approximate minimum eigenpair $(\xi, v) \in \mathbf{R} \times \mathbf{R}^n$ of M

```
1  function ApproxMaxEvec(M; q)
2      ω ← randn(n)
3      v ← ω/‖ω‖
4      for i ← 1, 2, 3, ..., q do
5          v ← Av
6          v ← v/‖v‖
7      return (v*(Mv), v)
```

Power method: guarantees

Fact (Randomized power method ([KW92]))

Let M be a real psd matrix. After $q \geq 2$ iterations, the randomized power method computes an ϵ-approximate maximum eigenvector v with

$$\mathbb{E}\epsilon \geq 0.871 \frac{\log n}{q-1}.$$

- arithmetic cost is $\mathcal{O}(q)$ matrix–vector multiplies with M and $\mathcal{O}(qn)$ extra operations
- working storage is about $2n$ numbers

Outline

Approximate eigenvectors

Definition

For a symmetric matrix $M \in \mathbf{S}^n_+$, we say a unit vector v is an ε-**approximate minimum eigenvector** if

$$v^* M v \leq \lambda_{\min}(M) + \varepsilon \|M\|.$$

Approximate eigenvectors

Definition

For a symmetric matrix $M \in \mathbf{S}_+^n$, we say a unit vector v is an ε-**approximate minimum eigenvector** if

$$v^* M v \leq \lambda_{\min}(M) + \varepsilon \|M\|.$$

how to find approximate minimum eigenvector (efficiently)?

- (-) Krylov methods (*e.g.*, ARPACK `eigs`)
  - unstable, hard to control precision
- (+) shifted power method
  - converges in $\mathcal{O}(\varepsilon^{-1})$ iterations, needs $\mathcal{O}(n)$ storage
- (+) randomized Lanczos method
  - converges in $q = \mathcal{O}(\varepsilon^{-1/2})$ iterations, needs $\mathcal{O}(nq)$ storage
  - (or can use $\mathcal{O}(n)$ storage by running it twice)

Approximate eigenvectors: shifted power method

- use power iteration to find max eigenvalue
- min eigenvalue of M is max eigenvalue of $\|M\|I - M$

Algorithm ApproxMinEvec via randomized shifted power method

Input: $M \in \mathbf{S}_n$, and maxiters q

Output: Approximate minimum eigenpair $(\xi, v) \in \mathbf{R} \times \mathbf{R}^n$ of M

1 **function** ApproxMinEvec(M; q)
2 $\quad\sigma \leftarrow \|M\|$
3 $\quad v \leftarrow \texttt{randn}(n, 1)/\sqrt{n}$
4 $\quad$ **for** $i \leftarrow 1, 2, 3, \ldots, q$ **do**
5 $\quad\quad v \leftarrow \sigma v - Mv$
6 $\quad\quad v \leftarrow v/\|v\|$
7 $\quad$ **return** $(v^*(Mv), v)$

Power method: guarantees

Fact (Randomized shifted power method ([KW92]))

Let $M \in \mathbf{S}_n$. For $\varepsilon \in (0, 1]$ and $\delta \in (0, 1]$, the shifted power method computes a unit vector $u \in \mathbf{R}^n$ that satisfies

$$u^* M u \leq \lambda_{\min}(M) + \varepsilon \|M\| \quad \text{w/prob} \geq 1 - \delta$$

after $q \geq \frac{1}{2} + \varepsilon^{-1} \log(n/\delta^2)$ iterations.[a]

[a] All logarithms are base-e.

- arithmetic cost is $\mathcal{O}(q)$ matrix–vector multiplies with M and $\mathcal{O}(qn)$ extra operations
- working storage is about $2n$ numbers

Outline

Randomized rangefinder

given matrix $A \in \mathbf{R}^{m \times n}$, subspace dimension ℓ

- draw random test matrix $\Omega \in \mathbf{R}^{n \times \ell}$
- compute sketch $Y = A\Omega$ $\quad \triangleright$ (ℓ matvecs)
- form QR decomposition $QR = Y$ $\quad \triangleright$ ($O(m\ell^2)$ flops)

Randomized rangefinder

given matrix $A \in \mathbf{R}^{m \times n}$, subspace dimension ℓ

- draw random test matrix $\Omega \in \mathbf{R}^{n \times \ell}$
- compute sketch $Y = A\Omega$ $\quad \triangleright$ (ℓ matvecs)
- form QR decomposition $QR = Y$ $\quad \triangleright$ ($O(m\ell^2)$ flops)

then

- $Q \in \mathrm{range}(A)$
- if $\ell > \mathrm{Rank}(A)$, then $\mathrm{span}(Q) = \mathrm{range}(A)$ with probability 1
- for quantitative error bounds, see ch 11 PGMJAT20

Outline

How to catch a low rank matrix

if $\hat{X}$ has the same rank as $X^\star$,
and $\hat{X}$ acts like $X^\star$ (on its range and co-range),
then $\hat{X}$ is $X^\star$

use single-pass randomized sketch
[TYUC19, TYUC17b, TYUC17a]

- learn how the matrix acts on a random subspace
- reconstruct a low rank matrix that acts like $X^\star$
- storage cost for sketch and arithmetic cost of update are $\mathcal{O}(r(m+n))$
- reconstruction is $\mathcal{O}(r^2(m+n))$

Single-pass randomized sketch

- Draw and fix two independent standard normal matrices

$$\Omega \in \mathbf{R}^{n \times k} \quad \text{and} \quad \Psi \in \mathbf{R}^{\ell \times m}$$

with $k = 2r + 1$, $\ell = 4r + 2$.

Single-pass randomized sketch

- Draw and fix two independent standard normal matrices

$$\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad \Psi \in \mathbf{R}^{\ell\times m}$$

  with $k = 2r + 1$, $\ell = 4r + 2$.
- The sketch consists of two matrices that capture the range and co-range of X:

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

Single-pass randomized sketch

- Draw and fix two independent standard normal matrices

$$\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad \Psi \in \mathbf{R}^{\ell\times m}$$

  with $k = 2r + 1$, $\ell = 4r + 2$.
- The sketch consists of two matrices that capture the range and co-range of X:

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

- Rank-1 updates to X can be performed on sketch:

$$X' = \beta_1 X + \beta_2 uv^*$$

$$\Downarrow$$

$$Y' = \beta_1 Y + \beta_2 uv^*\Omega \quad \text{and} \quad W' = \beta_1 W + \beta_2 \Psi uv^*$$

Single-pass randomized sketch

- Draw and fix two independent standard normal matrices

$$\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad \Psi \in \mathbf{R}^{\ell\times m}$$

  with $k = 2r + 1$, $\ell = 4r + 2$.
- The sketch consists of two matrices that capture the range and co-range of X:

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

- Rank-1 updates to X can be performed on sketch:

$$X' = \beta_1 X + \beta_2 uv^*$$

$$\Downarrow$$

$$Y' = \beta_1 Y + \beta_2 uv^*\Omega \quad \text{and} \quad W' = \beta_1 W + \beta_2 \Psi uv^*$$

- Both the storage cost for the sketch and the arithmetic cost of an update are $\mathcal{O}(r(m + n))$.

Recovery from sketch

To recover rank-r approximation $\hat{X}$ from the sketch, compute

1. $Y = QR$ ▷ (tall-skinny QR)
2. $B = (\Psi Q)^\dagger W$ ▷ (small QR + backsub)
3. $\hat{X} = Q[B]_r$ ▷ (tall-skinny SVD)

Recovery from sketch

To recover rank-r approximation $\hat{X}$ from the sketch, compute

1. $Y = QR$ ▷ (tall-skinny QR)
2. $B = (\Psi Q)^\dagger W$ ▷ (small QR + backsub)
3. $\hat{X} = Q[B]_r$ ▷ (tall-skinny SVD)

Theorem (Reconstruction [YUTC17])

Fix a target rank r. Let X be a matrix, and let (Y, W) be a sketch of X. The reconstruction procedure above yields a rank-r matrix $\hat{X}$ with

$$\mathbb{E}\|X - \hat{X}\|_F \leq 2\|X - [X]_r\|_F.$$

Similar bounds hold with high probability.

Recovery from sketch: intuition

recall

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

Recovery from sketch: intuition

recall

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

- if Q is an orthonormal basis for $\mathrm{range}(X)$, then

$$X = QQ^*X$$

Recovery from sketch: intuition

recall

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

- if Q is an orthonormal basis for $\mathrm{range}(X)$, then

$$X = QQ^*X$$

- if $QR = X\Omega$, then Q is (approximately) a basis for $\mathrm{range}(X)$

Recovery from sketch: intuition

recall

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

- if Q is an orthonormal basis for $\mathrm{range}(X)$, then

$$X = QQ^*X$$

- if $QR = X\Omega$, then Q is (approximately) a basis for $\mathrm{range}(X)$
- and if $W = \Psi X$, we can estimate

$$\begin{aligned} W &= \Psi X \\ &\approx \Psi QQ^*X \\ (\Psi Q)^\dagger W &\approx Q^*X \end{aligned}$$

Recovery from sketch: intuition

recall

$$Y = X\Omega \in \mathbf{R}^{n\times k} \quad \text{and} \quad W = \Psi X \in \mathbf{R}^{\ell\times m}$$

- if Q is an orthonormal basis for $\mathrm{range}(X)$, then

$$X = QQ^*X$$

- if $QR = X\Omega$, then Q is (approximately) a basis for $\mathrm{range}(X)$
- and if $W = \Psi X$, we can estimate

$$\begin{aligned} W &= \Psi X \\ &\approx \Psi QQ^*X \\ (\Psi Q)^\dagger W &\approx Q^*X \end{aligned}$$

- hence we may reconstruct X as

$$X \approx QQ^*X \approx Q(\Psi Q)^\dagger W$$


J. Kuczyński and H. Woźniakowski.
Estimating the largest eigenvalue by the power and Lanczos algorithms with a random start.
SIAM J. Matrix Anal. Appl., 13(4):1094–1122, 1992.


J. A. Tropp, A. Yurtsever, M. Udell, and V. Cevher.
Practical sketching algorithms for low-rank matrix approximation.
SIAM Journal of Matrix Analysis and Applications (SIMAX), 38(4):1454–1485, 2017.


Joel A. Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher.
Fixed-rank approximation of a positive-semidefinite matrix from streaming data.
In *Advances in Neural Information Processing Systems*, 2017.


Joel A Tropp, Alp Yurtsever, Madeleine Udell, and Volkan Cevher.

Streaming low-rank matrix approximation with an application to scientific simulation.
SIAM Scientific Computing (SISC), 2019.

Alp Yurtsever, Madeleine Udell, Joel Tropp, and Volkan Cevher.
Sketchy decisions: Convex low-rank matrix optimization with optimal storage.
In *Artificial Intelligence and Statistics*, pages 1188–1196, 2017.