Review:

- Int operators

$$A_{ij} = \frac{1}{|x_i - y_j|}$$

(x_i ↑ tgt, y_j ↑ src)

"compact"

- LRA
  ↳ SVD

Goals:

- randomized LRA
- two different forms

- HW1

$$ABC = (AB)C = A(BC)$$

$$ABx = (AB)x = A(Bx)$$

$$(\mathcal{F}\varphi)(x) = \int_a^x k_1(x,y)\,\varphi(y)\,dy$$

$$= \int_a^b \tilde{k}_1(x,y)\,\varphi(y)\,dy$$

Eckart-Young-Mirsky Theorem

Theorem (Eckart-Young-Mirsky)

SVD $A = U\Sigma V^T$. *If* $k < r = \operatorname{rank}(A)$ *and*

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T,$$

then

$$\min_{\operatorname{rank}(B)=k} |A - B|_2 = |A - A_k|_2 = \sigma_{k+1}.$$

Q: What's that error in the Frobenius norm?

So in principle that's good news:

- We can find the numerical rank.
- We can also find a factorization that reveals that rank (!)

Demo: Rank of a Potential Evaluation Matrix (Attempt 2)

$$\min_{\operatorname{rank}(B)=k} |A - B|_2 = |A - A_k|_2 = \sigma_{k+1}.$$

$$\min_{\operatorname{rank}(B)=k} \|A - B\|_F = \|A - A_k\|_F = \|\Sigma - \Sigma_k\|_F = \sqrt{\sigma_{k+1}^2 + \cdots + \sigma_n^2}$$

Constructing a tool

There is still a slight downside, though.

- Forming the matrix: $\mathcal{O}(N^2)$
- Computing the SVD: $\mathcal{O}(N^3)$ ← too expensive

Representation

What does all this have to do with (right-)preconditioning?

$$A\vec{u} = b \qquad \rightarrow \quad \Delta u = 0 \text{ on } \Omega, \quad u = g \text{ on } \partial\Omega$$

Left precond: $M_1 A \vec{u} = M_1 b$

Right precond: $\vec{u} = M_2 \vec{\sigma}$

tall + skinny

$$(A M_2)\vec{\sigma} = \vec{b}$$

"right-precondition away the PDE"

Representation (in context)

Outline

Outline

Rephrasing Low-Rank Approximations

SVD answers low-rank-approximation ('LRA') question. But: too expensive. First, rephrase the LRA problem:

$A \approx BC^T$ ← factorization form $B = [\;]$ (k columns) $C = [\;]$ (k columns)

$A \approx Q \underbrace{Q^T A}$ ← projection form $Q = [\;]$ (k columns) orth.

proj. → fact form?

$A \approx \underbrace{Q}_{B} \underbrace{Q^T A}_{C}$

Using LRA bases

If we have an LRA basis Q, can we compute an SVD?

(1) $B = Q^T A \in \mathbb{R}^{k \times n}$

(2) $B = \bar{U} \Sigma V^T \qquad \bar{U} \in \mathbb{R}^{k \times k}$

(3) $A \approx Q Q^T A = QB = \underbrace{Q \bar{U}}_{U} \Sigma V^T \; ; \; U = Q\bar{U}$

Complexities: $A \in \mathbb{R}^{n \times n} \qquad Q \in \mathbb{R}^{n \times k}$

(1) $k N^2$

(2) $k^2 N$

(3) $k^2 N$

Finding an LRA basis

How would we *find* an LRA basis?

(1) Using the SVD (...)

(2) use a different way?

↳ need a way to make err.-rank tradeoff

- "fixed-rank"
- adaptive.

Giving up optimality

What problem should we actually solve then?

$$\|A - QQ^T A\|_2 \approx \min_{\text{rank}(X) \le k} \|A - X\|_2$$

where $Q \in \mathbb{R}^{n \times (k+p)}$

p is the "unnecessary" extra rank.

Recap: The Power Method

How did the power method work again?

A square, eigenvalues; diagonalizable

$$|\lambda_1| \geq |\lambda_2| \geq \dots \geq |\lambda_n| \geq 0 \qquad A\vec{y}_i = \lambda_i \vec{y}_i$$

$\vec{x}_0 \in \mathbb{R}^n$ random $\qquad \vec{x}_0 = \alpha_1 \vec{y}_1 \dots + \alpha_n \vec{y}_n$

$$A^q \vec{x}_0 \quad \rightsquigarrow \quad \frac{A^q \vec{x}_0}{\lambda_1^q} = \frac{\alpha_1 \lambda_1^q \vec{y}_1}{\lambda_1^q} + \alpha_2 \left(\frac{\lambda_2^q}{\lambda_1^q}\right) \vec{y}_2 + \dots$$

$\left(\frac{\lambda_2^q}{\lambda_1^q}\right) \leq 1$

How do we construct the LRA basis?

Put randomness to work:

① Draw an $n \times \ell$ Gaussian iid random matrix Ω

② $Y = A\Omega$

③ $Y = Q R$

orth. $\in \mathbb{R}^{n \times \ell}$

want: more power

Tweaking the Range Finder (I)

Can we accelerate convergence?

$$Y = (AA^T)^k A$$

$$A = U\Sigma V^T$$

$$Y = (U\Sigma V^T V \Sigma U^T)^k U\Sigma V^T$$

$$\sigma_i(AA^TA) = \sigma_i(A)^3$$

Tweaking the Range Finder (II)

What is one possible issue with the power method?

- overflow → normalize
- non-orth → orthogonalize.

Even Faster Matvecs for Range Finding

Assumptions on Ω are pretty weak–can use more or less anything we want.
$\rightarrow$ Make it so that we can apply the matvec $A\Omega$ in $O(n \log \ell)$ time.
How? Pick Ω as a carefully-chosen subsampling of the Fourier transform.

?

Outline

Errors in Random Approximations

If we use the randomized range finder, how close do we get to the optimal answer?

Theorem

For an $m \times n$ matrix A, a target rank $k \geqslant 2$ and an oversampling parameter $p \geqslant 2$ with $k + p \leqslant \min(m, n)$, with probability $1 - 6 \cdot p^{-p}$,

$$\left| A - QQ^T A \right|_2 \leqslant \left(1 + 11\sqrt{k+p}\sqrt{\min(m,n)} \right) \sigma_{k+1}.$$

(given a few more very mild assumptions on p)

[Halko/Tropp/Martinsson '10, 10.3]

Message: We can *probably* (!) get away with oversampling parameters as small as $p = 5$.

A-posteriori and Adaptivity

The result on the previous slide was *a-priori*. Once we're done, can we find out 'how well it turned out'?

$$E = A - QQ^T A$$

$$\|E\|_2 = \sigma_1(E)$$

$\tilde{\omega} \in \mathbb{R}^n$ iid Gaussian

$$\|E\|_2 \approx \max \frac{\|E\tilde{\omega}\|_2}{\|\tilde{\omega}\|_2}$$

Adaptive Range Finding: Algorithm

1. Compute (small-ish) fixed-rank LRA
2. Check error
3. Add more columns if needed.

Outline