September 1, 2026

Announcements

- HW1

Goals

- Low rank approximation
  - Use (SVD)
  - Forming
    - theoretical
    - practical

Review

Low-Rank Point Interaction Matrices

Usable with low-rank complexity reduction?

$$\psi(x_i) = \sum_{j=1}^{N} G(x_i, y_j)\varphi(y_j)$$

$$\vec{y} = A\vec{x} \quad \rightsquigarrow \quad \vec{y} = \sum_i \vec{u}_i \vec{v}_i^T \vec{x}$$

$$\psi(x) = \quad \rightsquigarrow \quad \psi(x) = \sum_i \int u_i(x) \quad v_i(y)\varphi(y)\,dy$$

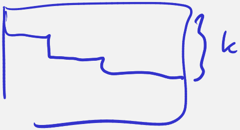
separation of variables

$$\psi(x) = \int G(x,y)\,\varphi(y)\,dy$$

Numerical Rank

What would a *numerical* generalization of ‘rank’ look like?

RREF

$A = C \cdot$ 

Eckart-Young-Mirsky Theorem

$\Sigma = \begin{pmatrix} \sigma_1 & & \\ & \ddots & \\ & & 0 \end{pmatrix} \geq 0$

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)

Constructing a tool

There is still a slight downside, though.

- Build matrix $O(n^2)$
- SVD $O(n^3)$

Outline

Outline

Rephrasing Low-Rank Approximations

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

$$\underset{n\times n}{A} \approx \underset{n\times k}{B}\,\underset{k\times n}{C^T} \quad \leftarrow \text{factorization form}$$

$$A \approx \underset{n\times k}{Q}\,Q^T A \quad \leftarrow \text{projection form}$$

orthonormal columns

Using LRA bases

$A : n\times n$
$Q : n\times k$

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

$k\times n$

	cost
① $C = Q^T A$	$n^2 k$
② SVD $C = \bar{U}\Sigma V^T$	nk^2
③ $U = Q\bar{U}$	nk^2

$$A \approx QQ^T A = \underbrace{Q\bar{U}}_{U}\Sigma V^T$$

$m\times n$ $\quad$ $m \geq n$

D

$m^2 n$

$mn^2 \leftarrow$ (Golub-Reinsch!)

Finding an LRA basis

How would we *find* an LRA basis?

Giving up optimality

What problem should we actually solve then?

$\le C\,\sigma_{k+1}$

$\approx$

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

Recap: The Power Method

$\vec{x}_1 \dots \vec{x}_n$

$AX = XD$

$\lambda_1 \dots \lambda_n$

$|\lambda_1| \geq |\lambda_2| \geq \dots$

How did the power method work again?

$A\vec{x}$

$\vec{x} = \alpha_1 \vec{x}_1 + \alpha_2 \vec{x}_2 + \dots + \alpha_n \vec{x}_n$

$A\vec{x} = \alpha_1 \lambda_1 \vec{x}_1 + \alpha_2 \lambda_2 \vec{x}_2 + \underbrace{\dots + \alpha_n \lambda_n \vec{x}_n}$

How do we construct the LRA basis?

Put randomness to work:

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

2 $Y = A\Omega$

3 $Y = QR$

Tweaking the Range Finder (I)

Can we accelerate convergence?

$$\cancel{Y = A^q \Omega}$$

$$Y = (AA^T)^q A \Omega$$

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

$$U \Sigma V^T V \Sigma U^T U \Sigma V^T$$
$$= U \Sigma^3 V^T$$

Tweaking the Range Finder (II)

What is one possible issue with the power method?

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 $O(n^2 \log \ell)$ time.
How? Pick Ω as a carefully-chosen subsampling of the Fourier transform.
(many other approaches also exist)

Outline

Errors in Random Approximations

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

Theorem (Halko/Tropp/Martinsson '10, (1.9))

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)

What's the role of the oversampling parameter p?