(Algorithmic) Linear Algebraic Tools

LECTURE 5

$$\boxed{\text{Multiplicative Weights}}$$

Solving Linear Systems

$$Ax = b \pmod 2$$

A: $n \times n$ matrix

Iterative Methods

Gaussian elimination

LU decomposition

- $O(n^3)$ runtime
- $O(n^2)$ space
- No dependence on condition number

$e^{-t/K} \rightarrow$ Steepest Descent

Karczmarz method

$e^{-t/\sqrt{K}} \rightarrow$ Conjugate Gradient Descent

- Condition number
- $O(n)$ space
- exploit sparsity

Algorithms underlie linear algebra

Ex: Prove: $\det(AB) = \det(A) \cdot \det(B)$

Gradient Descent on $Ax = b$

[Assume: A is real symmetric psd matrix]

$Ax = b \Longrightarrow \boxed{A^T A} x = A^T b$

psd matrix

a matrix multiplication

$$\boxed{\min f(x) = \frac{1}{2} x^T A x - b \cdot x \quad \text{because } \boxed{A \succeq 0} \quad f \text{ is convex}}$$

$\nabla f = 0 \iff Ax - b = 0$

$$f(x) = \frac{1}{2} x^T A x - bx \qquad \nabla f(x) = Ax - b$$

$$\boxed{\begin{array}{l|l} x_0 = 0 & x_s = \eta L^{-1}[Ax_s - b] \\ x_{s+1} = & x_s - \eta\,[A x_s - b] \end{array}}$$

$A \cdot x_s$ ← 1 matrix vector multiplication $+ O(n)$

If $nnz(A)$ = number of nonzeros of A

$A \cdot x_s$ can be compute in time $nnz(A)$.

Proof:

$$x_{s+1} - x_* = [x_s + \eta \nabla \underbrace{f(x_s)}] - x_*$$

optimal soln

$$\boxed{\begin{array}{l} Ax_s - b = \nabla f \\ Ax_* - b = 0 \end{array}}$$

$$= [I - \eta A] \cdot [x_s - x_*]$$

$$X_{s+1} - X_* = (I - \eta A)(x_s - x_*)$$

psd matrix

$\underline{\lambda_1 \ldots \lambda_n} \geq 0$

eigenvalues of $I - \eta A$ $= \{|1 - \underline{\eta}\lambda_1|, \ldots, |1 - \eta\lambda_n|\}$

< 1

$$\|X_s - X_*\|_2 = \|(I - \eta A)^s (X_0 - X_*)\|_2$$

$$\leq \rho^s \cdot \|\underline{x_0 - x_*}\|_2$$

$\|I - \eta L^{-1} A\|_{op} \rightarrow \|I - \eta A\|_{op}^s$

$$\leq (1 - \kappa(A))^s \|x_0 - x_*\|_2$$

$\kappa(L^{-1}A)$

$\lambda_1 \le \lambda_2 \dots \le \lambda_n$

$1 - \eta \lambda_n \Rightarrow \eta = \frac{1}{\lambda_n}$

$\Rightarrow$ ~~lar~~ largest $\{1 - \eta\lambda_1, \dots, 1 - \eta\lambda_n\}$

$$\boxed{= 1 - \frac{\lambda_1}{\lambda_n}} = 1 - \frac{1}{\kappa}$$

$$\boxed{\begin{array}{l} \kappa = \text{condition number} = \dfrac{\lambda_{max}}{\lambda_{min}} \\ \kappa(A). \end{array}}$$

<u>Thm:</u> error after t steps $\|x_t - x_*\| \le e^{-\frac{t}{\kappa(A)}}$

Notation:

$$\|x\|_2 = \left(\sum x_i^2\right)^{1/2}$$

$$\|x\|_p = \left(\sum x_i^p\right)^{1/p}$$

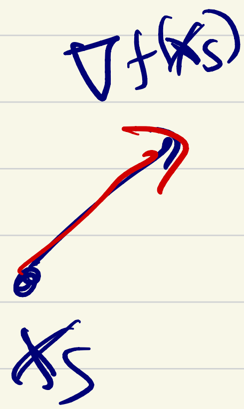
$$A : [\text{Vector}] \to [\text{vector}]$$

$$[\text{vector}] \to \mathbb{R}$$

$$\|A\|_{op} = \max_{\|x\| \leq 1} \|Ax\|$$

If A is real symmetric,

$$\|A\|_{op} = \max_i \{ |\lambda_i| \}$$



$$h(\eta) = f(x_s + \underline{\underline{\eta}}\, \nabla f(x_s))$$

Pick η that minimises $h(\eta)$

$$= (x_s + \eta(Ax_s - b))^T A (x_s + \eta(Ax_s - b))$$

$$- b[x_s + \eta(Ax_s - b)]$$

Preconditioned Gradient Descent

Find a matrix L

- $\kappa(L^{-1}A)$ is small

$$x_{s+1} = x_s - \eta L^{-1}[Ax - b]$$

multiplication by L^{-1}

1 multiplication by A

Quickly Solve $Lx = \omega$

- Quickly multiply by L^{-1}??

To compute $L^{-1}w = x$

$\Downarrow$

Solving $Lx = \omega$

Reduction from Solving $\boxed{Ax = b}$

$$\Big\Downarrow \quad \kappa(L^{-1}A) \text{ times}$$

$$\boxed{\text{Solving } \boxed{Lw = y}}$$

$A \in$ graph

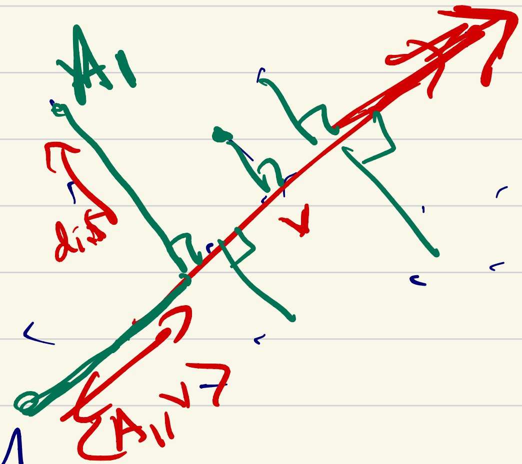
$L \in$ approximates the graph

$$\kappa(L^{-1}A) \approx O(1)$$

$$Lxw = y$$

SVD / PCA

$\in \mathbb{R}^d$



$$\begin{array}{cc} \xleftrightarrow{\quad d \quad} & \\ \begin{bmatrix} - A_1 - \\ - A_2 - \\ \vdots \\ - A_n \rightarrow \end{bmatrix} & \updownarrow n \end{array}$$

Best 1 dimensional approximation

$$\min_{v} \sum_i \mathrm{dist}(A_i, \overset{\text{direction}}{v})^2$$

$$\parallel$$

$$\max_{\|v\|=1} \sum_i \langle A_i, v\rangle^2 = \sum_i \mathrm{Proj}_v(A_i)$$

Best k-dimensional

$$\max_{\substack{k\text{-dimensional} \\ \text{subspace} \\ W = \mathrm{Span}(v_1 \ldots v_k)}} \sum_i \| \mathrm{Proj}_W (A_i) \|^2$$

Thm: [Kannan-Hopcroft-Blum)

$\rightarrow$ Greedy algorithm can compute the best k-dimensional.

$$\boxed{v_1 = \underset{\|v\|=1}{\text{Maximize}} \sum_i \langle A_i, v\rangle^2} \rightarrow \underline{\text{Solve?!}}$$

$$v_2 = \underset{\substack{\|v\| \perp v_1 \\ \|v\| \le 1}}{\text{Maximize}} \sum \langle A_i, v\rangle^2$$

$\vdots$

$$v_k = \underset{v \perp v_1 \ldots v_{k-1}}{\text{Maximize}} \sum_{i=1}^{n} \langle A_i, v\rangle^2$$

$$\max_{\|v\|\leq 1} \quad \sum_{i=1}^{n} \langle A_i, v\rangle^2 = \|Av\|_2^2$$

$$Av = \begin{bmatrix} \langle A_1, v\rangle \\ \vdots \\ \langle A_n, v\rangle \end{bmatrix}$$

$$\Updownarrow$$

$$\max_{\|v\|\leq 1} \quad v^T A^T A v = \lambda_{max}(\underbrace{A^T A}_{\text{psd mat.}})$$

QR-decomposition

$O(n^3)$

Iterative algorithm

(power method)

Let $B = \underline{A^T A}$ ← real symmetric

$$0 \leq \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n = 1$$

$\underline{v_1}$ ... v_n

Compute v_n:

At random

$$x = \sum_{i=1}^{n} \underline{c_i} \, \underline{v_i}$$

v_i are eigen vectors of B.

$$\underline{B} x = B \sum c_i v_i = \sum c_i (B v_i)$$

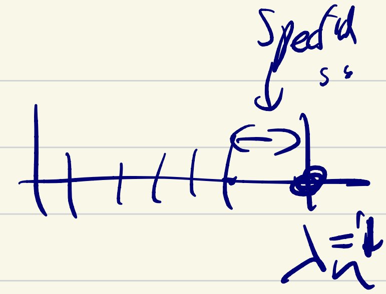
$$= \sum c_i \underbrace{\lambda_i}\, v_i$$

$$B^2 x = \sum_i c_i \lambda_i^2 v_i$$

$$B^3 \cancel{x} = \sum_i c_i \lambda_i^3 v_i$$

$$= c_n v_n (1)^3 + \sum c_i v_i \cdot (\lambda_i^3)$$

$$A = \begin{pmatrix} A_1 & \cdots & A_n \end{pmatrix}$$



$$\begin{pmatrix} A_1 - \langle A_1, v_d \rangle v_d \\ \vdots \\ A_n - \langle A_n, v_d \rangle v_d \end{pmatrix}$$

$x_1 \dots x_k \qquad x_i \perp x_j$

$\to Bx_1 \dots Bx_k$

$\to y_1 \dots y_k$

Runtime: $\underbrace{\lambda_1 \le \lambda_2 \dots \lambda_{n-1}}_{\le (1-\varepsilon)} < \lambda_n = 1$

Accuracy: $(1-\varepsilon)^t d \Rightarrow$ runtime $\frac{1}{\varepsilon}$