Lecture 17

- Avg. asymptotic runtime of iterative methods: ✓ When to use vs CG?

- Why $\alpha = \frac{2}{\lambda_1 + \lambda_n}$ in Richardson? ✓

How does change

- SVD w/ single vector ✓

^

- $\sigma = \sqrt{\lambda}$, how does $M = U \Sigma V^T$ vs. $M = P^{-1} \Sigma P$ ✓

- Cond. # and speed of conv. for coord. descent?

- Scale to avoid bad direction in ill-condition problems? ✓

- How does Julia do

- How do we w/a Hilbert

- We use K why not K

- if $M(\underline{x})$ is What to do?

- Can we always (By changing

QR? ✓

Solve ✓ How do we get the nasty inverse?

Matrix?

$(A) = \|A\|_2 \cdot \|A^{-1}\|_2$

$(A) = \|A\|_1 \cdot \|A^{-1}\|_1$? ✓

not differentiable

→ look @ Higham's book.

avoid ill-conditioning?

the formula?) ✓

Rules of Thumb on Iteration method

- If you know $\kappa(A)$ or have good bounds

$\sqrt{\kappa(A)}$

Something like $\log \varepsilon / \log(\sqrt{\kappa(A)})$

to get the # of iters.

If you don't know
$\kappa(A)$,

$\simeq O(n)$ iterations.

Maybe $O(\sqrt{n})$ if very
nice

$A = \underline{w}$. $n \times 1$

what is SVD of A ?

$U : n \times n$

Σ $n \times 1$

$V : 1 \times 1$

$w = U \Sigma$

$=$

$\Rightarrow \Sigma = \begin{bmatrix} \sigma \\ 0 \\ \vdots \\ 0 \end{bmatrix}$

$V = \pm 1$

$U = \begin{bmatrix} \pm w/\sigma & U_2 \end{bmatrix}$

$u^T u = 1 \Rightarrow$

$\| w/\sigma \| = 1 \Rightarrow$

$\sigma = \|w\|$

$\rightarrow U$: Householder Reflector that moves $\underline{w}$ to $\sigma \underline{e}_1$.

If A

I want

$A = Q$

$A = U$

Λ can

$\Sigma =$

$V =$

is ~~diagonalizable~~

normal

ortho vectors

ΛQ^T (λ-vals)

ΣU^T

diag mat

$$Q|\Lambda| \cdot \underbrace{\text{sign}(\Lambda)\, Q^T}_{= U^T}$$

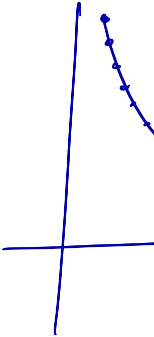
be negative ☹

$|\Lambda|$

$Q \, \text{sign}(\Lambda)$

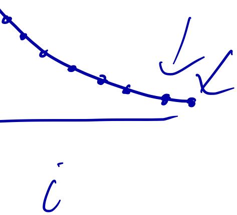
$K(\Lambda)$

Q evals

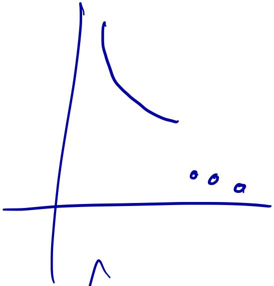


$$= \frac{\sigma_1}{\sigma_n}$$

work ☹

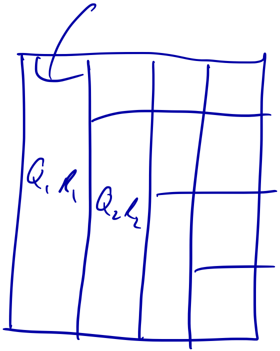


Woah!!



Only a few small σ_i's

Julia + QR



WY alg for QR.

$Q_1 \quad R_1$

Mixed Precision Solve

$A\underline{x} = \underline{b}$

Given $\underline{x}$, $\underline{r} = A\underline{x} - \underline{b}$ in 64-bit.

Solve $A(y) = \underline{r}$ in 32-bit

Update $x \leftarrow \underline{x} + y$

(!) Try this on Hilbert!