Welcome to Ma221! Lecture 30, Nov 1

More results on why Rayleigh Quotients are good approx evals

$$\rho(v,A) = \frac{v^T A v}{v^T v}, \quad v \neq 0$$

Thm Given $\|x\|_2 = 1$ and β

Then A has an eval α:

$$|\alpha - \beta| \leq \|Ax - \beta x\|_2$$

Given only x, $\beta = \rho(A,x)$ minimizes $\|Ax - \beta x\|_2$

Given any unit vector x, there is an eval within distance $\|A \cdot x - \rho(x,A) \cdot x\|_2$ of $\rho(x,A)$, and $\rho(x,A)$ minimizes this distance

Now let λ_i be eval of A closest to $\rho(x,A)$, $gap = \min_{j \neq i} |\lambda_j - \rho(x,A)|$

Then $|\lambda_i - \rho(x,A)| \leq \dfrac{\|Ax - \rho(x,A) \cdot x\|_2^2}{gap}$

(later, will get cubic convergence in QR iteration)

proof: $\|x\|_2 = 1 = \|(A - \beta I)^{-1}(A - \beta I)x\|_2$

$$\leq \|(A - \beta I)^{-1}\|_2 \cdot \|Ax - \beta x\|_2$$

$A = Q \Lambda Q^T$

$$= \|(\Lambda - \beta I)^{-1}\|_2 \cdot \|Ax - \beta x\|_2$$

$$= \frac{1}{\min_i |\lambda_i - \beta|} \cdot \|Ax - \beta x\|_2$$

$$\Rightarrow \min_i |\lambda_i - \beta| \leq \|Ax - \beta x\|_2$$

Show $\beta = \rho(x, A)$ minimizes $\|Ax - \beta x\|_2$

$$Ax - \beta x = \underbrace{Ax - \rho(x,A)x}_{y} + \underbrace{\rho(x,A)x - \beta x}_{z}$$

if $y^T z = 0$

$$\|Ax - \beta x\|_2^2 = \|y\|_2^2 + \|z\|_2^2$$

$$\geq \|y\|_2^2 = \|Ax - \rho(x,A)x\|_2^2$$

$$z^T y = \left((\rho(x,A) - \beta)x\right)^T \left(Ax - \rho(x,A)x\right)$$

$$= (\rho(x,A) - \beta)\underbrace{\left(x^T Ax - \rho(x,A)x^T x\right)}$$

$= 0$ by def of $\rho(x,A)$

Last Part: Special case of A 2x2 diagonal, captures all ideas of general case

$$A = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}, \quad x = \begin{bmatrix} c \\ s \end{bmatrix} \qquad c^2 + s^2 = 1$$

$$\rho(x, A) = c^2\lambda_1 + s^2\lambda_2$$

assume $c^2 > s^2 \Rightarrow \rho(x, A)$ closest to λ_1

$$\frac{\|Ax - \rho(x,A)x\|_2^2}{gap} = |\lambda_1 - \rho(x,A)| \quad \begin{matrix} \text{exactly} \\ \geq \text{ in general case} \end{matrix}$$

Algorithms for symeig and SVD

(1) "Usual accuracy":

backward stable: exact evals/evecs for $A+E$, $\|E\|_2 = O(\text{macheps}) \cdot \|A\|_2$

(1.1) all evals (with or without evecs)
(1.2) just evals in $[x,y]$ (w or w/o evecs)
(1.3) just evals $\lambda_i, \lambda_{i+1}, \ldots, \lambda_j$ (w or w/o evecs)

(1.2) and (1.3) can be much cheaper than (1.1)

Note: backward stability makes $\|Ax - \lambda x\| = O(\text{macheps})\|A\|$

but x_i and x_j for λ_i and λ_j may or may not be orthogonal

(2) "High accuracy": get tiny evals with more correct digits

Ex: if A well-conditioned i.e. all sing vals ~ same magnitude

then "usual accuracy" $\Rightarrow$ error bound $\pm O(\varepsilon)\|A\|$ $\Rightarrow$ all sing vals computed with correct leading digits

Consider $B = D \cdot A$, $D = \mathrm{diag}(d_1, \ldots, d_n)$
where some $d_i \gg$ other d_j
$\Rightarrow$ some $\sigma_i \gg$ other σ_j
$\Rightarrow$ usual accuracy $\Rightarrow$ no correct leading digits in tiny σ_j

$\exists$ perturbation theory + algs. to get all σ_i with correct leading digits

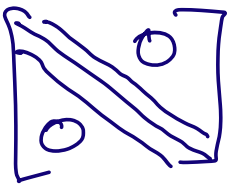
Other structures permitting high accuracy: $A = A^T$ well conditioned
get acc evals for DAD
(see links in class web page)

(3) Updating: given evals and evecs of A, compute them for $A \pm xx^T$
much cheaper than starting from scratch
(basis of default eig(A))

All these options apply to $A = A^T$ and SVD

Algorithms and Costs

(1) Reduce $A = Q T Q^T$ $\quad QQ^T = I$

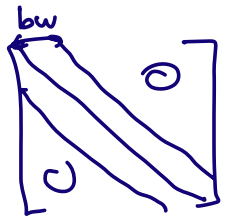
$T = T^T$ tridiagonal 

(Same idea as Hessenberg reduction in Chap 4) (CLAPACK: ssytrd)

all subsequent algs assume $A = T$

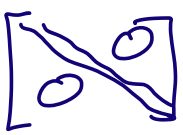
There is an alg for reducing $A \to T$ that attains lower bound

$O\left(\frac{n^3}{\sqrt{\text{cache size}}}\right)$ words moved

if A banded 

cost of reduction drops from $O(n^3)$ to $O(n^2 \cdot bw)$

- SVD analogous: Reduce A to bidiagonal form: $B = U \Sigma V^T$



(1.1) Given T, find all evals (w or w/o evecs).

Many algs, all cost $O(n^2)$ for evals alone

If evecs too, costs range from $O(n^2)$ to $O(n^3)$

(even $O(n \cdot \log n)$ if evecs "implicit")

Varying numerical stability

(1.1.1) Oldest is QR iteration as in Chap 4

Thm (Wilkinson) With right choice of shift σ, tridiagonal QR is globally convergent, usually cubic convergence

Cost: $O(n^2)$ for evals only but $O(n^3)$ for evecs too more expensive than later ones (LAPACK: ssyev)

SVD in LAPACK: sgesvd uses a variant of QR iteration with additional property that all σ_i computed with high relative accuracy, independent of size

(1.1.2) Compute subset of evals (w or w/o evecs)

Improve cost of getting all evecs from $O(n^3)$ to $O(n^2)$, but no more guarantee of orthogonal evecs with λ_i, λ_j close

(1) compute evals $O(n^2)$

(2) compute each evec using inverse iteration (Chap 4)

$$x_{i+1} = (T - \lambda I)^{-1} x_i$$

$x_{i+1} \to$ evec for λ, corresponds to largest eval of $(T - \lambda I)^{-1}$

if λ, and λ' close, no guarantee of orthogonality: worse case $\lambda \neq \lambda'$ but round to same floating point number

Lapack: ssyevx

Long standing Goal: find orthog evecs in $O(n^2)$ – solved by MRRR

(1.1.3) Divide and Conquer:

Lapack ssyevd

(Cuppen, Gu + Eisenstat)

faster than QR, not $O(n^2)$

guarantees orthogonality

Cost $O(n^g)$ $2 < g < 3$

(1.1.4) MRRR: Multiple Relatively Robust Representations:

like inverse iteration but guarantees orthog evecs (Parlett, Dhillon)

Lapack: ssyevr

Extension to SVD by Paul Willems but same counter examples, not in LAPACK, (open problem)

Beating $O(n^2)$ $T = Z \Lambda Z^T$

Thm (Gu): One can compute Z implicitly in $O(n \cdot \log^p n)$ ops (can multiply Zx cheaply) but all evecs would cost $O(n^3)$