Welcome back to Ma 221! Lecture 31, Nov 3

1) "Usual accuracy" : backward stable

Reduce A to T = tridiagonal

all evals only : QR or bisection

all evals and evecs: QR if T small enough
else divide-and-conquer or MRRR

some evals only: bisection

some evals + evecs: bisection + inverse iteration
or MRRR

2) High accuracy:

Jacobi's method (1864) works on A, not T

LAPACK (SVD only) sgesvj

3) Updating $A = Q \Lambda Q^T$ for $A \pm vv^T$

QR iteration: (matlab demo for showing cubic convergence, code in typed notes)

Cubic convergence follows from analysis of Rayleigh Quotient Iter.

$i = 0$, choose unit vector x_0

repeat

$s_i = \rho(x_i, A) = x_i^T A x_i$

$y = (A - s_i I)^{-1} x_i$

$x_{i+1} = y / \|y\|_2$

until convergence

Inverse iteration using Rayleigh Quotient as shift, best available approx eval. given x_i

Suppose $Aq = \lambda q$ $\|q\|_2 = 1$ $\|x_i - q\|_2 = e \ll 1$

Want to show $\|x_{i+1} - q\|_2 = O(e^3)$

Bound $|s_i - \lambda|$:

$$s_i = x_i^T A x_i = (x_i - q + q)^T A (x_i - q + q)$$

$$= (x_i - q)^T A (x_i - q) + q^T A (x_i - q)$$

$$+ (x_i - q)^T A q + \underbrace{q^T A q}_{\lambda}$$

$$s_i - \lambda = (x_i - q)^T A (x_i - q) + 2\lambda (x_i - q)^T q$$

$$|s_i - \lambda| \leq O(\|x_i - q\|_2^2) + O(\|x_i - q\|_2)$$

$$= O(e^2) + O(e) = O(e)$$

want tighter bound

$$|s_i - \lambda| \leq \|A x_i - s_i x_i\|_2^2 / gap$$

... gap = distance from s_i to second closest eval

... Thm 5.5 in text, sketch last time

$$= \|A(x_i - q + q) - s_i (x_i - q + q)\|_2^2 / gap$$

$$= \|(A - s_i I)(x_i - q) + (\lambda - s_i) q\|_2^2 / gap$$

$$\leq (\|(A - s_i I)(x_i - q)\|_2 + \|\underbrace{(\lambda - s_i)}_{O(e)} q\|_2)^2 / gap$$

$\underbrace{\quad}_{O(e)}$

$= O(e^2)/gap$

Use analysis from Chap 4 of one step of inverse iteration:

$$\|x_{i+1} - q\|_2 \le \underbrace{\|x_i - q\|_2}_{O(e)} \cdot \underbrace{|s_i - \lambda|}_{O(e^2)} / gap$$

$= O(e^3)$ if gap not too small

Show that QR iteration doing Rayleigh quotient iteration implicitly

$T - s_i I = QR \longrightarrow$ new $T = RQ + s_i I$

$$(T - s_i I)^{-1} = R^{-1} Q^{-1}$$
$$= R^{-1} Q^T \quad \text{symmetric}$$
$$= Q R^{-T}$$

$\Rightarrow (T - s_i I)^{-1} R^T = Q$

last column: $(T - s_i I)^{-1} e_n \cdot R_{nn}$

$s_i = T(n,n) = e_n^T T e_n = \rho(e_n, T)$

$\Rightarrow q_n =$ last col of Q = result of one step of Rayleigh Quotient Iteration starting with e_n

new $T = RQ + s_i I = Q^T T Q$

$(\text{new } T)(n,n) = q_n^T T q_n = \rho(q_n, T)$

$\Rightarrow$ QR iteration doing Rayleigh Quotient Iteration

Actual implementation does "bulge chasing" as in Chap 4

$\Rightarrow$ cost $= O(n)$ per iteration
$= O(n)$ per eval
$= O(n^2)$ for all evals

But cost for evecs $= O(n^3)$ by taking product of $O(n^2)$ Givens rotations, want to be faster

Beating $O(n^3)$ by Divide & conquer only for all evals + evecs

Main ingredient: cheaply updating $A = Q \Lambda Q^T$ for $A + \alpha v v^T$

$$A + \alpha v v^T = Q \Lambda Q^T + \alpha v v^T$$
$$= Q(\Lambda + \alpha (Q^T v)(v^T Q)) Q^T$$
$$= Q(\Lambda + \alpha \cdot z \cdot z^T) Q^T \qquad z = Q^T v$$

needs are evals + evecs of $\Lambda + \alpha \cdot z \cdot z^T$
$=$ diagonal + rank -1

use: characteristic polynomial

Lemma (HW5.14) $\det(I + xy^T) = 1 + y^T x$

$$\det(\Lambda + \alpha z z^T - \lambda I)$$
$$= \det((\Lambda - \lambda I)(I + \alpha \underbrace{(\Lambda - \lambda I)^{-1} z z^T}_{\text{rank-1}}))$$
$$= \prod_{i=1}^{n} (\lambda_i - \lambda) \cdot \left(1 + \alpha \sum_{i=1}^{n} \frac{z_i^2}{\lambda_i - \lambda}\right)$$

solve $f(\lambda)=0$ "secular equation"

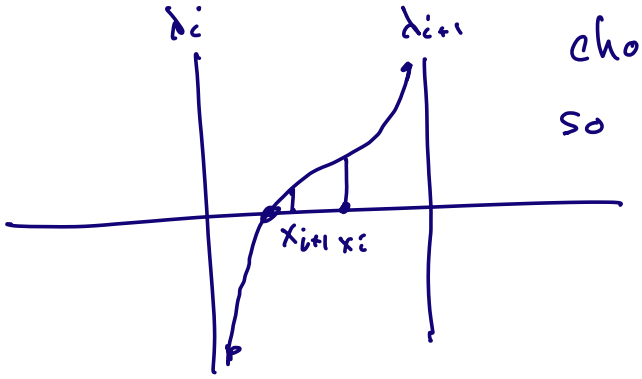
Fig 5.2: $f(\lambda) = 1 + \frac{.5}{1-\lambda} + \frac{.5}{2-\lambda} + \frac{.5}{3-\lambda} + \frac{.5}{4-\lambda}$

Fig 5.3 $f(\lambda) = 1 + \frac{.001}{1-\lambda} + \frac{.001}{2-\lambda} + \frac{.001}{3-\lambda} + \frac{.001}{4-\lambda}$

Lesson: Newton would fail, use different approx than tangent line

approx $f(\lambda)$ by $g(\lambda)$ that has poles at λ_i and λ_{i+1} and matches f and f' at last iteration

$$g(\lambda) = c_1 + \frac{c_2}{\lambda_i - \lambda} + \frac{c_3}{\lambda_{i+1} - \lambda}$$



choose c_1, c_2, c_3 so $g(x_i) = f(x_i)$
$g'(x_i) = f'(x_i)$

Also need evecs

Lemma: if λ eval of $\Lambda + \alpha v v^T$ then its evec is $(\Lambda - \lambda I)^{-1} v$

proof: $(\Lambda + \alpha v v^T)(\Lambda - \lambda I)^{-1} v$

$= (\Lambda - \lambda I + \lambda I + \alpha v v^T)(\Lambda - \lambda I)^{-1} v$

$$= v + \lambda (\Lambda - \lambda I)^{-1} v + v (\underbrace{\alpha v^T (\Lambda - \lambda I)^{-1} v}_{\cdot = -1 \text{ since } f(\lambda) = 0})$$

— cancel

$$= \lambda (\Lambda - \lambda I)^{-1} v$$

Unfortunately not numerically stable because of cancellation in $(d_i - \lambda)^{-1}$

clever fix in text (Gu, Eisenstat)

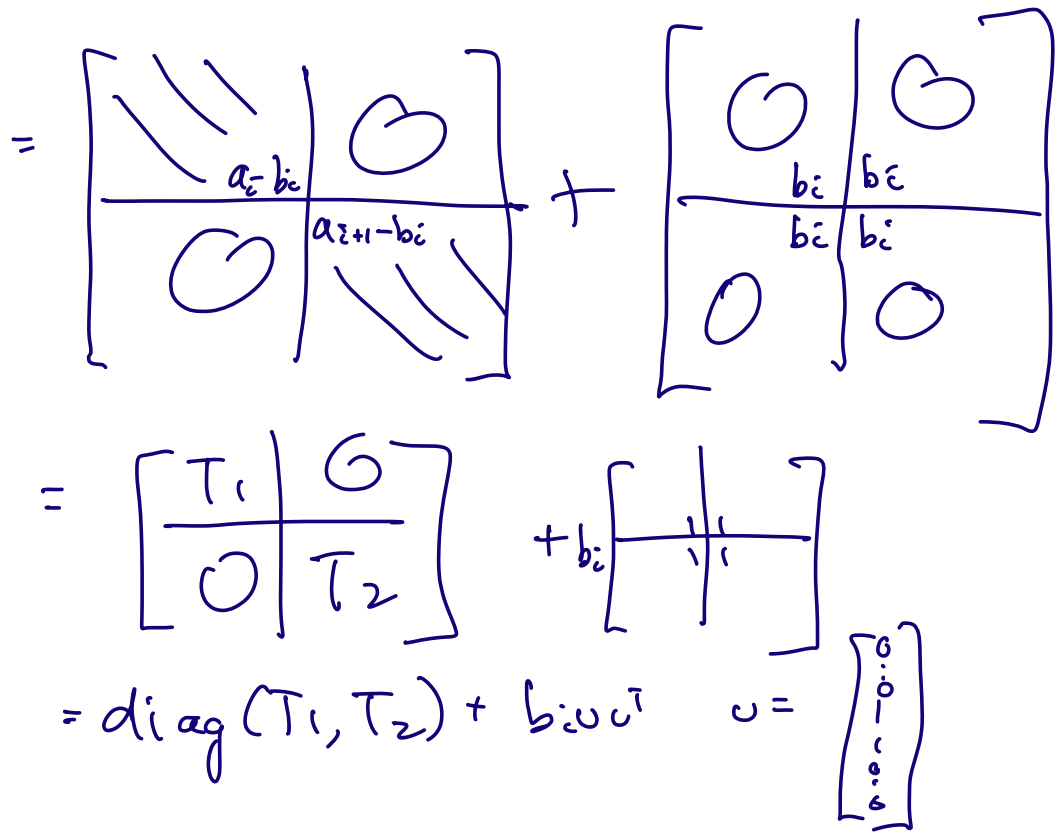
For general eigenproblem, How to do Divide & Conquer?

Reduce dense A to tridiagonal T

Write alg just described as

$$[Q', \Lambda'] = \text{Eig-update}[Q, \Lambda, \alpha, v]$$

How to divide tridiagonal T into 2 half-sized problems?



$$= \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} + b_i \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

$$= \mathrm{diag}(T_1, T_2) + b_i u u^T \qquad u = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

```
function [Q, Λ] = DC_eig(T)
  if n small enough
      use QR
  else
      i = [n/2]
      [Q1, Λ1] = DC_eig(T1)
      [Q2, Λ2] = DC_eig(T2)
      [Q, Λ] = Fig_update(diag(Q1, Q2),
                          diag(Λ1, Λ2), bi, u)
  endif
return
```

$$\text{cost} = O(n^{\beta}) \qquad 2 < \beta < 3$$