Eigenvalue algorithms: Rayleigh iteration

MATH 6610 Lecture 19

October 21, 2020

Trefethen & Bau: Lecture 27

Eigenvalues: power iteration

Let $A \in \mathbb{C}^{n \times n}$ be Hermitian with eigenpairs $(\lambda_j, v_j)_{j=1}^n$ that are simple and ordered such that $|\lambda_j| > |\lambda_{j+1}|$.

Power iteration performs the following basic steps:

Initialize with a vector v (e.g., randomly):

1. $v \leftarrow \frac{Av}{\|Av\|_2}$
2. $\lambda \leftarrow R_A(v)$
3. Return to step 1

For a large number of iterations, then (λ, v) converges to (λ_1, v_1).

To make this practically useful, need to supplement with termination criterion, deflation.

Power iteration's drawback

One major drawback of power iteration is that the ratio

$$r = \left| \frac{\lambda_{j+1}}{\lambda_j} \right|,$$

Recall convergence $\mathcal{O}(r^k)$ @ iteration k.

dictates the rate of convergence for computing eigenpair j.

This implies, in particular, that the effectiveness of this algorithm depends heavily on A.

$$A = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

Power iteration's drawback

One major drawback of power iteration is that the ratio

$$\left|\frac{\lambda_{j+1}}{\lambda_j}\right|,$$

dictates the rate of convergence for computing eigenpair j.

This implies, in particular, that the effectiveness of this algorithm depends heavily on A.

But power iteration is simple and the idea is attractive. Is there a way to accelerate power iteration?

Inverse iteration

Inverse iteration "modifies" the spectrum of A so that power iteration will be "more successful."

Spectrum has "larger" spacing.

Inverse iteration

Inverse iteration "modifies" the spectrum of A so that power iteration will be more successful.

First we note that, given some $\mu \in \mathbb{C}$, the eigenvalues of $(A - \mu I)^{-1}$ are $1/(\lambda_j - \mu)$.

Inverse iteration

Inverse iteration "modifies" the spectrum of A so that power iteration will be more successful.

First we note that, given some $\mu \in \mathbb{C}$, the eigenvalues of $(A - \mu I)^{-1}$ are $1/(\lambda_j - \mu)$.

The essential realization is that, if $\mu \approx \lambda_j$, then

$$\frac{1}{|\mu - \lambda_j|} \gg \frac{1}{|\mu - \lambda_k|}, \qquad k \neq j.$$

I.e., if we can somehow get a "reasonable" guess μ, then power iteration on $(A - \mu I)^{-1}$ will be very efficient.

Rayleigh iteration

The idea of Rayleigh iteration is to combine

- inverse iteration, which accelerates identification of eigenvectors, with
- Rayleigh quotient evaluations, which accelerate identification of eigenvalues.

Rayleigh iteration

The idea of Rayleigh iteration is to combine

- inverse iteration, which accelerates identification of eigenvectors, with
- Rayleigh quotient evaluations, which accelerate identification of eigenvalues.

Initialize with a vector v and scalar μ (e.g., randomly):

1. $v \leftarrow (A - \mu I)^{-1} v$ (inverse iteration)
2. $v \leftarrow v / \|v\|_2$
3. ~~λ~~ μ $\leftarrow R_A(v)$ (Rayleigh quotient)
4. Return to step 1

In principle this is more expensive than power iteration: we must solve $(A - \mu I)x = v$ at *every* step.

Rayleigh iteration, II

The rather surprising fact: this is an *extremely* efficient algorithm.

Let $\lambda^{(k)}, v^{(k)}$ be the Rayleigh iteration eigenpair approximation at iteration k.
Let $\lambda^{(k+1)}, v^{(k+1)}$ be the Rayleigh iteration eigenpair approximation at iteration $k+1$.

Rayleigh iteration, II

The rather surprising fact: this is an *extremely* efficient algorithm.

Let $\lambda^{(k)}, v^{(k)}$ be the Rayleigh iteration eigenpair approximation at iteration k.
Let $\lambda^{(k+1)}, v^{(k+1)}$ be the Rayleigh iteration eigenpair approximation at iteration $k+1$.

Theorem

For almost every initialization of Rayleigh iteration, there is some eigenpair (λ_j, v_j) of A such that

$$\left|\lambda_j - \lambda^{(k+1)}\right| \leqslant \left|\lambda_j - \lambda^{(k)}\right|^3, \qquad \left\|v_j - v^{(k+1)}\right\| \leqslant \left\|v_j - v^{(k)}\right\|^3$$

Convergence is <u>cubic</u>!

<u>Proof</u> (-ish): 1 exponent of convergence from power iteration
2 exponents from Rayleigh quotient.

Rayleigh iteration: choose initial vector v.

$v^{(0)} = v$

$\lambda^{(0)} = R_A(v)$

for $k=1,2,\ldots$

$$w = (A - \lambda^{(k-1)} I)^{-1} v^{(k-1)}$$

$$v^{(k)} = w / \|w\|$$

$$\lambda^{(k)} = R_A(v^{(k)})$$

before was μ

For HW: need termination + deflation

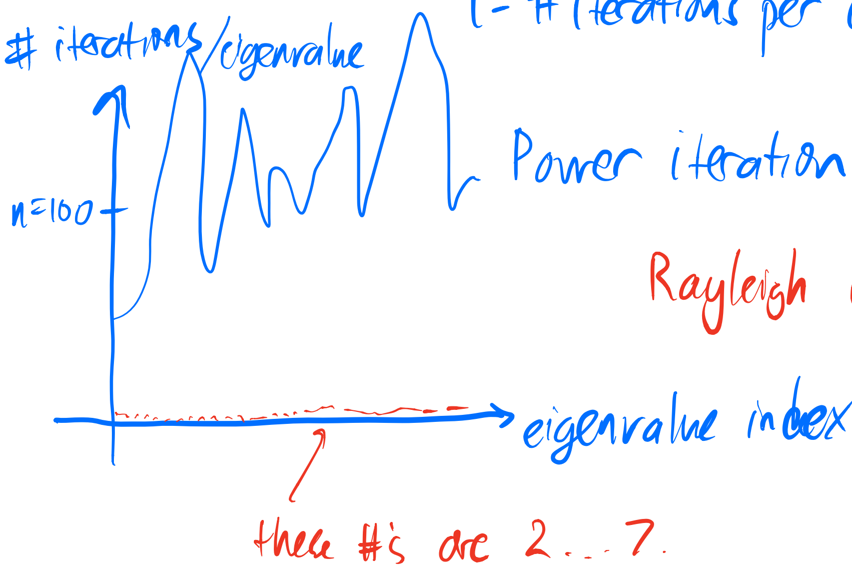
Rayleigh iteration is really efficient.

E.g. $A \in \mathbb{C}^{n \times n}$, $n=100$.

Power iteration: performs some iterations, periodically deflate.

↑ Same idea for Rayleigh iteration

Let's count # of iterations per deflation.



(= # iterations per eigenvalue found).

Power iteration (—)

Rayleigh iteration (- - -)

Hotelling deflation (A Hermitian)

$$A = \sum_{j=1}^{n} \lambda_j v_j v_j^* \quad \text{(E-V decomposition of } A\text{)}$$

Suppose find eigenpair (μ, w). [$= (\lambda_k, v_k)$ for some k]

$$A - \mu w w^* = \sum_{\substack{j=1 \\ j \neq k}}^{n} \lambda_j v_j v_j^*$$

← rank-$(n-1)$ sum