2

Iterative Ideas

$$\left(\begin{array}{l} A \in \mathbb{R}^{m\times m} \\ A^T = A \end{array} \right.$$

① Power Iteration

seed vector

eigen vectors

→ Start with $x = \sum \alpha_i v_i$

→ $Ax = \sum \alpha_i A v_i$

$= \sum \alpha_i \lambda_i v_i$

→ $A^n x = \sum \alpha_i \lambda_i^n v_i$

→ let $\lambda_1 = \rho(A) = \max_i \{ |\lambda_i| \mid A v_i = \lambda_i v_i \}$

Then

$$\lim_{n\to\infty} A^n x = \sum \alpha_i \lambda_i^n v_i$$

$$= \lim_{n\to\infty} \left(\lambda_1^n\right) \sum \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^n v_i$$

$$= \lim_{n\to\infty} \lambda_1^n \alpha_1 v_1 \equiv X^{(n)}$$

To control the magnitude of $X^{(n)}$,

we normalize :

$\rightarrow$ let $\quad x_n = \dfrac{x^{(n)}}{\| x^{(n)} \|} \approx \dfrac{\lambda_1^n \alpha_1 v_1}{\| \lambda_1^n \alpha_1 v_1 \|}$

$$= \pm v_1$$

To get eig val cor. to v_1 : Consider

$$x_n^T A x_n \qquad (A v_1 = \lambda_1 v_1)$$

$$= (\pm v_1)^T \overbrace{A \, (\pm v_1)}$$

$$= v_1^T \lambda_1 v_1 = \lambda_1 \|v_1\|^2 = \lambda_1$$

Alg <u>Power Iteration</u>

x = arbitrary nonzero vector with $\|x\| = 1$

r = residual vector with $\|r\| =$ large

do while $\|r\| >$ desired accuracy

$$y = \frac{Ax}{\|Ax\|}$$

$$r = y - x$$

$$x = y$$

end while

The rate of

Rmk ① Power Iteration $\simeq \left(\frac{\lambda_i}{\lambda_1}\right)$

$\therefore$ not a fast method

② just reveals one eig vector

③ the direct outcome is the eig vector corr. to the largest eig val

need to be calcuted separately

$\boxed{2.2}$ The inverse iteration

→ will address :

① faster convergence

② cherry - picking is available

Setup : Suppose we want v_J ← eig vector

corr. to one λ_J

with $Av_J = \lambda_J v_J$

$\rightarrow$ Suppose you can find $\mu \approx \lambda_J$
but $\mu \neq \lambda_J$.

$\rightarrow$ $\dfrac{1}{\lambda_J - \mu} \rightarrow$ the largest eig val of

$$B = (A - \mu I)^{-1}$$

pf $A v_J = \lambda_J v_J$

translation

$\rightarrow (A - \mu I) v_J = (\lambda_J - \mu) v_J$

$\rightarrow v_J = (A - \mu I)^{-1} (\lambda_J - \mu) v_J$

$\rightarrow \dfrac{1}{\lambda_J - \mu} v_J = (A - \mu I)^{-1} v_J$

$= B v_J$ ▨

Alg. Inverse Iteration

Initialize $\mu \neq 0$

r = a vector with $\|r\|$ = large

$B = (A - \mu I)^{-1}$

do while $\|r\| >$ dersired accuracy

$$\boxed{y = \frac{Bx}{\|Bx\|}}$$

$r = y - x$

$x = y$

end while

Rmk $(A - \mu I) y = x$

$\Downarrow$

$$y = \frac{y}{\|y\|}$$

Rmk i) One downside is to first approximate $\mu \, (\approx \lambda_J)$

(But we can Gershgorin Thm if possible)

ii) still it's slow in convergence

2.3 Raleigh Quotient Iteration (RQ)

Def. $A \in \mathbb{R}^{n \times m}$,

the Raleigh Quotient is a ftn r

$$r : \mathbb{R}^m \longrightarrow \mathbb{R}$$
$$\quad\ \ \rotatebox{90}{$\in$}$$
$$x \longmapsto \boxed{r(x) = \frac{x^T A x}{x^T x}}$$

Rmk. If $x = v_i$ $\rightarrow$ eigen vector of A
satisfying $Av_i = \lambda_i v_i$,

then

$$r(v_i) = \frac{v_i^T A v_i}{v_i^T v_i} \qquad = \|v_i\|^2 = 1$$

$$= v_i^T \lambda_i v_i = \lambda_i \|v_i\|^2 = \lambda_i$$

The RQ Iteration

	Inverse iter	RQ
Input	eig val	eig vec
Output	eig vec	eig val

not necc. eg vec yet

~~repeat~~

Step 1 : make an initial guess for $x \neq 0$

Step 2 : $r(x) = \mu$

Step 3 : use μ in step 2 to do $(A - \mu I)^{-1}$ to get an update x

Rmk. Note that the eig vectors are critical pts of $r(x)$, i.e, they are local min of $r(x)$,

i.e, $\boxed{\nabla r(v_i) = 0,}$ $\forall i$ $\qquad r(x) = \dfrac{x^T A x}{x^T x}$

pf $\nabla r \equiv \left(\dfrac{\partial r}{\partial x_1}, \dots, \dfrac{\partial r}{\partial x_m} \right)^T$ $\qquad \left(\dfrac{f}{g}\right)'$

For each j,

$$\frac{\partial r}{\partial x_j} = \frac{(x^T x)\frac{\partial}{\partial x_j}(x^T A x) - (x^T A x)\frac{\partial}{\partial x_j}(x^T x)}{(x^T x)^2}$$

$$= \frac{(x^T x)\, 2(Ax)_j - (x^T A x)(2x)_j}{(x^T x)^2}$$

$$= \frac{2(Ax)_j}{x^T x} - r(x)\frac{(2x)_j}{x^T x}$$

$$= \frac{2}{x^T x}\left((Ax)_j - r(x)\, x_j\right)$$

$$\therefore \nabla r(x) = \frac{2}{x^T x}\left(\underbrace{Ax - r(x)x}\right)$$

λ_j

Then $\nabla r(v_j) = \frac{2}{v_j^T v_j}\left(A v_j - r(v_j)\, v_j\right)$

$$= 0 \quad \forall j. \quad \blacksquare$$

Rmk Consider

$$v = v_i + \varepsilon, \quad \|\varepsilon\| : \text{small}$$

$$\to r(v) = r(v_i + \varepsilon)$$

$$= \underbrace{r(v_i)}_{=\lambda_i} + \varepsilon \cdot \nabla r(v_i) + O(\|\varepsilon\|^2)$$

0

$= \lambda_i + O(\|\varepsilon\|^2)$

<u>Rmk</u>. RQ iteration converges to an eigen vec/val pair (λ_i, v_i) for almost any possible initial guess of $v \neq 0$.

<u>Rmk</u> RQ iter is <u>really</u> fast!

i.e, Cubic convergence, that is,

i) $\|v^{(n+1)} - v_i\| = O(\|v^{(n)} - v_i\|^3)$

ii) $|\lambda^{(n+1)} - \lambda_i| = O(|\lambda^{(n)} - \lambda|^3)$

$\boxed{3}$ QR algorithm w/o shifts

$\rightarrow$ can give us "multiple" (λ_i, v_i)s.

Ex <u>failed case</u>

Consider 2×2 system, $A \in \mathbb{R}^{2\times 2}$

let x_1 & x_2 : two initial guesses

Group 1: $X_1 = a_1 v_1 + a_2 v_2$

Group 2: $X_2 = b_1 v_1 + b_2 v_2$

$\longrightarrow$
$$\begin{cases} A^n X_1 = a_1 \lambda_1^n v_1 + a_2 \lambda_2^n v_2 \\ A^n X_2 = b_1 \lambda_1^n v_1 + b_2 \lambda_2^n v_2 \end{cases}$$

$\rightarrow$ Assume $\lambda_1 = \rho(A)$

$\rightarrow$
$$\begin{cases} A^n X_1 \longrightarrow a_1 \lambda_1^n v_1 \\ A^n X_2 \longrightarrow b_1 \lambda_1^n v_1 \end{cases}$$
!!!

Two approaches:

i) QR + Power iter $\boxed{SI}$

ii) QR + RQ iter $\boxed{SII}$

} Simultaneous iteration algorithms

Rmk $A = QR$

"power"

Alg SI (Simulataneous Iter)

$Q = I$

do while error > large

$Z = AQ$

$QR = Z$

enddo

$Q^{(0)} = I$

do while error > large

$Z^{(n)} = AQ^{(n-1)}$

$Q^{(n)} R^{(n)} = Z^{(n)}$

$n = n+1$

compute new error

enddo