Power Iteration

$A x = \lambda x \quad (x_i, \lambda_i)$

What are the eigenvalues of A^{1000}?

Assume $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$ with eigenvectors $\mathbf{x}_1, \ldots, \mathbf{x}_n$.
Further assume $\|\mathbf{x}_i\| = 1$.

$x_0 = \alpha x_1 + \beta x_2 + \gamma x_3 + \cdots$

$x_{k+1} = A x_k$

$x_{k+1} = A^k x_0 = \alpha \lambda_1^k x_1 + \beta \lambda_2^k x_2 + \gamma \lambda_3^k x_3 + \cdots$

$= \lambda_1^k \left(\alpha x_1 + \beta \left(\frac{\lambda_2}{\lambda_1}\right)^k x_2 + \gamma \left(\frac{\lambda_3}{\lambda_1}\right)^k x_3 + \cdots \right)$

$\ll 1 \qquad \ll 1$

$k \to \infty \quad \frac{x_{k+1}}{\lambda_1^k} \to \alpha x_1$ Converges to the eigenpair (x_1, λ_1)

Power Iteration: Issues?

What could go wrong with Power Iteration?

- Overflow $\longrightarrow$ normalize
- x_0 without component along x_1 $(\alpha = 0)$
  - unlike to happen if random start.
  - floating point operations usually introduce such component.
- $|\lambda_1| = |\lambda_2| \longrightarrow$ does not converge to x_1 (method fails)

What about Eigenvalues?

Power Iteration generates eigenvectors. What if we would like to know eigenvalues?

Rayleigh quotient : $\lambda = \frac{x^T A x}{x^T x}$

where x is the converged eigenvector from power iteration

Convergence of Power Iteration

What can you say about the convergence of the power method?
Say $\mathbf{v}_1^{(k)}$ is the kth estimate of the eigenvector $\mathbf{x}_1$, and

$$e_k = \left\| \mathbf{x}_1 - \mathbf{v}_1^{(k)} \right\|.$$

$e_k \sim c \left| \frac{\lambda_2}{\lambda_1} \right|^k \Longrightarrow \frac{\|e_{k+1}\|}{\|e_k\|} \approx \frac{\left(\frac{\lambda_2}{\lambda_1}\right)^{k+1}}{\left(\frac{\lambda_2}{\lambda_1}\right)^k} = \left|\frac{\lambda_2}{\lambda_1}\right|$ = constant

$$\|e_{k+1}\| = \left|\frac{\lambda_2}{\lambda_1}\right| \|e_k\|$$

linear convergence

Cost : $O(n^2)$ mat-vect

Matrix A with eigenvalues $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$

Inverse Power Iteration:

$$Ax = \lambda x \Longrightarrow A^{-1}Ax = A^{-1}\lambda x \Longrightarrow \frac{1}{\lambda}x = A^{-1}x$$

$\left(\frac{1}{\lambda}, x\right)$ is eigenpair of A^{-1}

Power iteration $x_{k+1} = A x_k$ → converges to largest $\lambda \Rightarrow \lambda_1$

Inverse Power iteration $x_{k+1} = A^{-1} x_k$ → converges to largest $\left(\frac{1}{\lambda}\right) \Rightarrow \lambda_n$

Algorithm: x_0 : random

for $k = 0, 1, \cdots$

solve $Ax_{k+1} = x_k$

normalize

Cost = ? $O(n^3)$ → Factorize first

→ solve sequence of triangular solve

Convergence:

$$\frac{\|e_{k+1}\|}{\|e_k\|} = \left|\frac{\lambda_n}{\lambda_{n-1}}\right|$$

Matrix A with eigenvalues $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$ $\boxed{Ax = \lambda x}$

Shifted Inverse:

$$x_{k+1} = (A - \sigma I)^{-1} x_k \longrightarrow \text{converges to largest eigenvalue of } (A-\sigma I)^{-1}$$

$$(A-\sigma I)^{-1} x = \bar{\lambda} x \rightarrow \frac{1}{\bar{\lambda}} x = (A-\sigma I)x = Ax - \sigma I x = \lambda x - \sigma x$$

$$\boxed{\bar{\lambda} = \frac{1}{\lambda - \sigma}}$$

largest $\bar{\lambda}$ $\Rightarrow$ smallest $(\lambda - \sigma)$ $\Rightarrow$ Find λ that is closest to σ

Algorithm:

x_0 = random

for $k = 0, 1, \cdots$

solve $(A - \sigma I) x_{k+1} = x_k$

normalize

- Factorize first $O(n^3)$ once
- solve triangular systems $O(n^2)$ k times
- Convergence:

$$\frac{\|e_{k+1}\|}{\|e_k\|} = \frac{|\lambda_{closest} - \sigma|}{|\lambda_{second\text{-}closest} - \sigma|}$$

Rayleigh Quotient iteration

$$\sigma_k = \frac{x_k^T A x_k}{x_k^T x_k}$$

$$(A - \sigma_k I) x_{k+1} = x_k$$

- at least quadratic convergence
- needs to factorize every iteration (more expensive) } trade-off
- may not converge to desired eigenvector.

GO TO DEMO!

Simultaneous Iteration

Power iteration methods → one eigenvalue at a time

What if we need all of them?

What happens if we carry out power iteration on multiple vectors simultaneously?

- start random $X_0 \in \mathbb{R}^{n \times p}$ $\quad p \leq n$
- iterate $X_{k+1} = A X_k$
- Issues:
  - overflow → rescale
  - X increasingly ill-conditioned (all columns of X converge to x_1)

(x_1, λ_1) : dominant eigenpair of A

$|\lambda_1| > |\lambda_2| \cdots$

Fix: Orthogonalize!

Orthogonal iteration

$A = QR$ → upper triangular

Q: orthogonal

start $X_0 \in \mathbb{R}^{n \times p}$

Factorize: $X_0 = Q_0 R_0$

Update: $X_1 = A Q_0$

Factorize: $X_1 = Q_1 R_1 \longrightarrow X_1 = Q_1 R_1 = A Q_0 \Rightarrow A = Q_1 R_1 Q_0^T$

Update: $X_2 = A Q_1$

Factorize: $X_2 = Q_2 R_2 \longrightarrow X_2 = Q_2 R_2 = A Q_1 \Rightarrow A = Q_2 R_2 Q_1^T$

Update: $X_3 = A Q_2$

⋮

Factorize: $X_k = Q_k R_k \longrightarrow X_k = Q_k R_k = A Q_{k-1} \Rightarrow A = Q_k R_k Q_{k-1}^T$

Update: $X_{k+1} = A Q_k$

when $Q_k \approx Q_{k-1}$ → converges (not a good convergence check!)

Algorithm

start $X_0 \in \mathbb{R}^{n \times p}$

for $k = 0, 1, \ldots$

Factorize $Q_k, R_k = la.qr(X_k)$

Update $X_{k+1} = A Q_k$

- slow convergend
- computationally expensive per iteration

At convergence: $A \simeq Q_k R_k Q_k^T$ → R_k diagonal entries are the eigenvalues of A

QR iteration

$X_0 = I \rightarrow X_0 = Q_0 R_0 \Rightarrow R_0 = Q_0 = I \Longrightarrow X_1 = AI$

start $A_0 = A$

Factorize: $A_0 = Q_0 R_0 \longrightarrow Q_0^T Q_0 R_0 = Q_0^T A_0 \longrightarrow R_0 = Q_0^T A_0$

Update: $A_1 = R_0 Q_0 \longrightarrow A_1 = Q_0^T A_0 Q_0$

Factorize: $A_1 = Q_1 R_1 \longrightarrow Q_1^T Q_1 R_1 = Q_1^T A_1 \longrightarrow R_1 = Q_1^T A_1$

Update: $A_2 = R_1 Q_1 \longrightarrow A_2 = Q_1^T A_1 Q_1$

Factorize: $A_2 = Q_2 R_2 \longrightarrow Q_2^T Q_2 R_2 = Q_2^T A_2 \longrightarrow R_2 = Q_2^T A_2$

Update: $A_3 = R_2 Q_2 \longrightarrow A_3 = Q_2^T A_2 Q_2$

⋮

Factorize: $A_k = Q_k R_k \longrightarrow Q_k^T A_k = Q_k^T Q_k R_k \longrightarrow R_k = Q_k^T A_k$

Update: $A_{k+1} = R_k Q_k \longrightarrow \boxed{A_{k+1} = Q_k^T A_k Q_k}$

Algorithm:

start $X_0 = A$

for $k = 0, 1, \ldots$

Factorize $Q_k, R_k = la.qr(X_k)$

Update $X_{k+1} = R_k Q_k$

$X_{k+1} \approx X_k \rightarrow$ converges

X_k diagonal entries are the eigenvalues of A

QR + Shift

→ Shift is chosen so that it is close to existing eigenvalue.

start $X_0 = A$

for $k = 0, 1, \ldots$

Factorize $Q_k, R_k = la.qr(X_k - \sigma_k I)$

Update $X_{k+1} = R_k Q_k + \sigma_k I$

$X_{k+1} \approx X_k \rightarrow$ converges

$X_k - \sigma_k I = Q_k R_k \Rightarrow Q_k^T (X_k - \sigma_k I) = Q_k^T Q_k R_k = R_k$

Still similarity transform: $X_{k+1} = R_k Q_k + \sigma_k I$

$X_{k+1} = Q^T (X_k - \sigma_k I) Q_k + \sigma_k I$

$X_{k+1} = Q_k^T X_k Q_k - \sigma_k Q_k^T Q_k + \sigma_k I$