Matrix Computations
Chapter 4: Eigenvalues, Eigenvectors, and Eigendecomposition

Section 4.4 Power Iteration and QR Iteration

Jie Lu
ShanghaiTech University

The Power Method

- A method for numerically computing an eigenvector of a given matrix

- Simple, though not the best in convergence speed
    - A comprehensive coverage of various computational methods for the eigenvalue problem can be found in Chapter 7 of textbook

- Suitable for large-scale sparse problems, e.g., PageRank

The Power Method/Power Iteration

Suppose $\mathbf{A} \in \mathbb{C}^{n\times n}$ admits an eigendecomposition $\mathbf{A} = \mathbf{V}\Lambda\mathbf{V}^{-1}$

The eigenvalues of $\mathbf{A}$ are ordered as $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$

Algorithm: Power Method
input: $\mathbf{A} \in \mathbb{C}^{n\times n}$ and an initial guess $\mathbf{v}^{(0)} \in \mathbb{C}^n$
for $k = 1, 2, \ldots$ (until a termination criterion is satisfied)
$\quad \tilde{\mathbf{v}}^{(k)} = \mathbf{A}\mathbf{v}^{(k-1)}$
$\quad \mathbf{v}^{(k)} = \tilde{\mathbf{v}}^{(k)} / \|\tilde{\mathbf{v}}^{(k)}\|_2$
$\quad \lambda^{(k)} = [\mathbf{v}^{(k)}]^H \mathbf{A}\mathbf{v}^{(k)}$
end
output: $\mathbf{v}^{(k)}, \lambda^{(k)}$

Complexity per iteration: $\mathcal{O}(n^2)$, or $\mathcal{O}(nzz(\mathbf{A}))$ for sparse $\mathbf{A}$

Result: $\mathrm{dist}(\mathrm{span}\{\mathbf{v}^{(k)}\}, \mathrm{span}\{\mathbf{v}_1\}) \to 0$ and $\lambda^{(k)} \to \lambda_1$ as $k \to \infty$

The convergence rates depend on $|\lambda_2|/|\lambda_1|$

Analysis of The Power Method

Let the initial guess

$$\mathbf{v}^{(0)} = \alpha_1 \mathbf{v}_1 + \alpha_2 \mathbf{v}_2 + \cdots + \alpha_n \mathbf{v}_n = \mathbf{V}\boldsymbol{\alpha}$$

We require $\alpha_1 \neq 0$ (random guess essentially works). Then,

$$\mathbf{A}^k \mathbf{v}^{(0)} = \mathbf{V}\Lambda^k \mathbf{V}^{-1}\mathbf{v}^{(0)} = \sum_{i=1}^{n} \alpha_i \lambda_i^k \mathbf{v}_i = \alpha_1 \lambda_1^k \Bigg(\mathbf{v}_1 + \underbrace{\sum_{i=2}^{n} \frac{\alpha_i}{\alpha_1} \left(\frac{\lambda_i}{\lambda_1}\right)^k \mathbf{v}_i}_{=\mathbf{r}^{(k)}} \Bigg)$$

where $\mathbf{r}^{(k)}$ is a residual satisfying

$$\|\mathbf{r}^{(k)}\|_2 \leq \sum_{i=2}^{n} \left|\frac{\alpha_i}{\alpha_1}\right| \left|\frac{\lambda_i}{\lambda_1}\right|^k \|\mathbf{v}_i\|_2 \leq \left|\frac{\lambda_2}{\lambda_1}\right|^k \sum_{i=2}^{n} \left|\frac{\alpha_i}{\alpha_1}\right| \|\mathbf{v}_i\|_2 \to 0 \text{ as } k \to \infty$$

Analysis of The Power Method (cont'd)

Note from the algorithm that $\mathbf{v}^{(k)} \in \mathrm{span}\{\mathbf{A}^k \mathbf{v}^{(0)}\}$

In fact, it can be verified that $\mathbf{v}^{(k)} = \dfrac{\mathbf{A}^k \mathbf{v}^{(0)}}{\|\mathbf{A}^k \mathbf{v}^{(0)}\|_2}$

Hence, $\mathbf{v}^{(k)}$ converges to an eigenvector associated with λ_1, i.e.,

$$\mathsf{dist}(\mathrm{span}\{\mathbf{v}^{(k)}\}, \mathrm{span}\{\mathbf{v}_1\}) = \mathcal{O}(|\tfrac{\lambda_2}{\lambda_1}|^k)$$

Accordingly,

$$\|\lambda^{(k)} - \lambda_1\| = \mathcal{O}(|\frac{\lambda_2}{\lambda_1}|^k)$$

The convergence is slow if $|\lambda_2|$ is closer to $|\lambda_1|$

The two conditions in red require that

- λ_1 is a dominant eigenvalue (i.e., $>$ all the other eigenvalues in modulus)
- The initial guess has a component in the direction of the corresponding dominant eigenvector

Without these conditions, the power method does not necessarily converge

Deflation

- The power method only computes the dominant eigenvalue and eigenvector
- How can we compute all the eigenvalues with the corresponding eigenvectors?

Consider a Hermitian matrix $\mathbf{A}$ with $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|$
Express $\mathbf{A}$ using the outer-product representation

$$\mathbf{A} = \sum_{i=1}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^H$$

Deflation: Use the power method to obtain $\mathbf{v}_1$, λ_1. Then, do the subtraction

$$\mathbf{A} - \lambda_1 \mathbf{v}_1 \mathbf{v}_1^H = \sum_{i=2}^{n} \lambda_i \mathbf{v}_i \mathbf{v}_i^H$$

Apply the power method to the above matrix and obtain $\mathbf{v}_2$, λ_2
Repeat until all the eigenvalues and eigenvectors are found

- Stop when completing the kth iteration gives the first k eigen-pairs

Orthogonal Iteration

- A generalization of the power method for computing higher-dimensional invariant subspaces

Algorithm: Method of Orthogonal Iteration
input: $\mathbf{A} \in \mathbb{C}^{n\times n}$, $r \in \{1, \ldots, n\}$, $\mathbf{Q}^{(0)} \in \mathbb{C}^{n\times r}$ semi-unitary
for $k = 1, 2, \ldots$ (until a termination criterion is satisfied)
    $\mathbf{Z}^{(k)} = \mathbf{A}\mathbf{Q}^{(k-1)}$
    Find (thin) QR decomposition of $\mathbf{Z}^{(k)}$: $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{Z}^{(k)}$, $\mathbf{Q}^{(k)} \in \mathbb{C}^{n\times r}$
    Find the eigenvalues $\lambda_1^{(k)}, \ldots, \lambda_r^{(k)}$ of $(\mathbf{Q}^{(k)})^H \mathbf{A}\mathbf{Q}^{(k)}$
end
output: $\mathbf{Q}^{(k)}$, $\lambda_1^{(k)}, \ldots, \lambda_r^{(k)}$

- $\mathbf{Q}^{(k)}\mathbf{e}_1$ is the vector generated by the power method starting from $\mathbf{v}^{(0)} = \mathbf{Q}^{(0)}\mathbf{e}_1$
- When $r = 1$, the algorithm reduces to the power method

Analysis of Orthogonal Iteration

Recall the Schur decomposition of $\mathbf{A} \in \mathbb{C}^{n\times n}$

$$\mathbf{U}^H\mathbf{A}\mathbf{U} = \mathbf{T}, \quad \mathbf{U} = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_n \end{bmatrix} \text{ unitary, } \mathbf{T} \text{ uppertriangular, } t_{ii} = \lambda_i$$

Fact: Let $\mathbf{U}_i = \begin{bmatrix} \mathbf{u}_1 & \cdots & \mathbf{u}_i \end{bmatrix}$, $i = 1, \ldots, n$. Then, $\mathcal{R}(\mathbf{U}_i)$ is an invariant subspace for $\mathbf{A}$ and the eigenvalues of $\mathbf{U}_i^H\mathbf{A}\mathbf{U}_i$ are $\lambda_1, \ldots, \lambda_i$

Analysis of Orthogonal Iteration (cont'd)

Analysis of Orthogonal Iteration (cont'd)

Suppose the eigenvalues of $\mathbf{A}$ are ordered as

$$|\lambda_1| \geq \cdots \geq |\lambda_r| > |\lambda_{r+1}| \geq \cdots \geq |\lambda_n|, \quad t_{ii} = \lambda_i$$

Partition $\mathbf{U}$ and $\mathbf{T}$ as

$$\mathbf{U} = \begin{bmatrix} \mathbf{U}_r & \mathbf{U}_\beta \end{bmatrix}, \quad \mathbf{U}_r \in \mathbb{C}^{n\times r},\ \mathbf{U}_\beta \in \mathbb{C}^{n\times(n-r)}$$
$$\mathbf{T} = \begin{bmatrix} \mathbf{T}_{11} & \mathbf{T}_{12} \\ \mathbf{0} & \mathbf{T}_{22} \end{bmatrix}, \quad \mathbf{T}_{11} \in \mathbb{C}^{r\times r},\ \mathbf{T}_{22} \in \mathbb{C}^{(n-r)\times(n-r)}$$

With $|\lambda_r| > |\lambda_{r+1}|$, $D_r(\mathbf{A}) := \mathcal{R}(\mathbf{U}_r)$ is called the dominant invariant subspace, which is the unique invariant subspace associated with the eigenvalues $\lambda_1, \ldots, \lambda_r$ of $\mathbf{A}$

Convergence: With proper assumptions,[1]

$$\operatorname{dist}(D_r(\mathbf{A}), \mathcal{R}(\mathbf{Q}^{(k)})) = \mathcal{O}(|\tfrac{\lambda_{r+1}}{\lambda_r}|^k)$$
$$|\lambda_i^{(k)} - \lambda_i| = \mathcal{O}(|\tfrac{\lambda_{i+1}}{\lambda_i}|^k), \quad i = 1, \ldots, r$$

[1] See Theorem 7.3.1 of textbook for details

Orthogonal Iteration (cont'd)

Let $\mathbf{A} \in \mathbb{C}^{n \times n}$ with Schur Decomposition $\mathbf{U}^H \mathbf{A} \mathbf{U} = \mathbf{T}$, where $\mathbf{U} = \left[\mathbf{u}_1, \ldots, \mathbf{u}_n\right]$ is unitary and $\mathbf{T}$ is uppertriangular with $t_{ii} = \lambda_i$ s.t.

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n|, \quad t_{ii} = \lambda_i$$

Let $\mathbf{Q}^{(k)} = \begin{bmatrix} \mathbf{q}_1^{(k)} & \cdots & \mathbf{q}_n^{(k)} \end{bmatrix}$ be generated by the method of orthogonal iteration with $r = n$

It can be shown that with a proper $\mathbf{Q}^{(0)}$,

$$\mathrm{dist}(\mathrm{span}\{\mathbf{q}_1^{(k)}, \ldots, \mathbf{q}_i^{(k)}\}, \mathrm{span}\{\mathbf{u}_1, \ldots, \mathbf{u}_i\}) \to 0 \text{ as } k \to \infty, \quad \forall i = 1, \ldots, n$$

This implies that $\mathbf{T}^k = (\mathbf{Q}^{(k)})^H \mathbf{A} \mathbf{Q}^{(k)}$ converges to an upper triangular matrix, so that the algorithm leads to a Schur decomposition

Orthogonal Iteration (cont'd)

Compute $\mathbf{T}^k$ more efficiently via its predecessor $\mathbf{T}^{(k-1)}$

$$\mathbf{T}^{(k-1)} = (\mathbf{Q}^{(k-1)})^H \mathbf{A} \mathbf{Q}^{(k-1)} = (\mathbf{Q}^{(k-1)})^H \mathbf{Z}^{(k)} = (\mathbf{Q}^{(k-1)})^H \mathbf{Q}^{(k)} \mathbf{R}^{(k)}$$

$$\begin{aligned} \mathbf{T}^{(k)} &= (\mathbf{Q}^{(k)})^H \mathbf{A} \mathbf{Q}^{(k)} = (\mathbf{Q}^{(k)})^H \mathbf{A} \mathbf{Q}^{(k-1)} \cdot (\mathbf{Q}^{(k-1)})^H \mathbf{Q}^{(k)} \\ &= (\mathbf{Q}^{(k)})^H \mathbf{Z}^{(k)} \cdot (\mathbf{Q}^{(k-1)})^H \mathbf{Q}^{(k)} = \mathbf{R}^{(k)} (\mathbf{Q}^{(k-1)})^H \mathbf{Q}^{(k)} \end{aligned}$$

This suggests that we may find $\mathbf{T}^{(k)}$ by computing the QR decomposition of $\mathbf{T}^{(k-1)}$ and then multiplying the factors in reverse order

The QR Algorithm/QR Iteration

The above computation of $\mathbf{T}^{(k)}$ motivates the QR algorithm

Algorithm: QR algorithm
input: $\mathbf{A} \in \mathbb{C}^{n \times n}$
$\mathbf{A}^{(0)} = \mathbf{A}$
for $k = 1, 2, \ldots$ (until a termination criterion is satisfied)
  Find QR decomposition of $\mathbf{A}^{(k-1)}$: $\mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{A}^{(k-1)}$
  $\mathbf{A}^{(k)} = \mathbf{R}^{(k)}\mathbf{Q}^{(k)}$
end
output: $\mathbf{A}^{(k)}$

- $\mathbf{A}^{(k)}$ $\forall k$ are similar matrices and thus have the same set of eigenvalues
- If the Schur decomposition of $\mathbf{A}$ is $\mathbf{A} = \mathbf{U}\mathbf{T}\mathbf{U}^H$, then under some mild assumptions, $\mathbf{A}^{(k)}$ converges to $\mathbf{T}$
  - The diagonal elements of $\mathbf{A}^{(k)}$ for a sufficiently large k would give all the eigenvalues of $\mathbf{A}$
- Complexity of each iteration: $\mathcal{O}(n^3)$
- Improved algorithms can be found in Sections 7.4 and 7.5 of textbook, including the practical QR algorithm (same main idea)