SI231b: Matrix Computations

Lecture 17: QR Iteration for Eigenvalue Computations

Yue Qiu

qiuyue@shanghaitech.edu.cn

School of Information Science and Technology

ShanghaiTech University

Nov. 07, 2022

Unnormalized Simultaneous Iteration

Define $\mathbf{V}^{(0)}$ to be the $n \times r$ matrix,

$$\mathbf{V}^{(0)} = \begin{bmatrix} v_1^{(0)} & v_2^{(0)} & \cdots & v_r^{(0)} \end{bmatrix}.$$

After k steps of applying $\mathbf{A}$, we obtain

$$\mathbf{V}^{(k)} = \mathbf{A}^k \mathbf{V}^{(0)} = \begin{bmatrix} v_1^{(k)} & v_2^{(k)} & \cdots & v_r^{(k)} \end{bmatrix}.$$

Assume

1. The leading $r+1$ eigenvalues are distinct in absolute value;

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_r| > |\lambda_{r+1}| \geq |\lambda_{r+2}| \geq \cdots |\lambda_n|$$

2. All the leading principle sub-matrices $\mathbf{Q}^T \mathbf{V}^{(0)}$ are nonsingular.
    - $\mathbf{Q}$ is the matrix with $\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_r$ as columns;
    - $\mathbf{q}_1, \mathbf{q}_2, \cdots, \mathbf{q}_r$ are eigenvectors associated with eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_r$.

```
choose V^(0) with r linear independent columns
for k = 1, 2, ···
      V^(k) = AV^(k-1)
      Q^(k)R^(k) = V^(k)   reduced QR factorization
end
```

Under the assumptions, we have as $k \to \infty$,

- For real symmetric matrix $\mathbf{A}$ ($\mathbf{Q}$ has orthonormal columns)

$$\|\mathbf{q}_j^{(k)} - (\pm q_j)\| = \mathcal{O}(C^k),$$

  for $1 \le j \le r$, where $C < 1$ is the constant

$$C = \max_{1 \le k \le r} \frac{|\lambda_{k+1}|}{|\lambda_k|}$$

- For unsymmetric matrix $\mathbf{A}$ ($\mathbf{Q}$ does not have orthonormal columns)

$$\mathcal{R}(\mathbf{Q}^{(k)}) \to \mathcal{R}(\mathbf{Q})$$

For Unnormalized Simultaneous Iteration, as $k \to \infty$, the vectors $\mathbf{q}^{(1)}, \mathbf{q}^{(2)}, \cdots, \mathbf{q}^{(r)}$ all converge to multiples of the same dominant eigenvector $\mathbf{q}_1$. Therefore, they form an ill-conditioned basis of span $\left\{\mathbf{q}^{(1)}, \mathbf{q}^{(2)}, \cdots, \mathbf{q}^{(r)}\right\}$.

The remedy is simple, we should build orthonormal basis at each iteration $\rightsquigarrow$ **Simultaneous Iteration/Subspace Iteration**

Subspace Iteration:

```
random selection Q^(0) with orthonormal columns
for k = 1, 2, ...
      Z_k = A Q^(k-1)
      Z_k = Q^(k) R^(k)   reduced QR factorization
end
```

- $\mathbf{Z}_k$ and $\mathbf{Q}^{(k)}$ has the same column space
- equal to the column space of $\mathbf{A}^k\mathbf{Q}^{(0)}$

- $\mathcal{R}(\mathbf{Q}^{(k)})$ converge to subspace associated with r largest eigenvalues in magnititude (dominant invariant subspace).
- $\lambda\left(\left(\mathbf{Q}^{(k)}\right)^H \mathbf{A}\mathbf{Q}^{(k)}\right) \rightarrow \{\lambda_1,\ \lambda_2,\ \cdots,\lambda_r\}$
- $\left|\lambda_i^{(k)} - \lambda_i\right| = \mathcal{O}\left(\left|\frac{\lambda_{r+1}}{\lambda_i}\right|^k\right),\ i = 1,\ 2,\ \cdots,\ r$
- also called simultaneously iteration or orthogonal iteration
- when $r = n$, it coincides with QR iteration

Hermitian/real symmetric matrices:

- Simultaneous convergence of eigenvectors

$$\|\mathbf{q}_j^{(k)} - (\pm q_j)\| = \mathcal{O}(C^k),$$

for $1 \leq j \leq r,\ C = \frac{\lambda_{r+1}}{\lambda_r}$

QR Iteration:

$$
\begin{array}{l}
\mathbf{A}^{(0)} = \mathbf{A} \\
\texttt{for } k = 1,\ 2,\ \cdots \\
\qquad \mathbf{Q}^{(k)}\mathbf{R}^{(k)} = \mathbf{A}^{(k-1)} \quad \texttt{QR factorization of } \mathbf{A}^{(k-1)} \\
\qquad \mathbf{A}^{(k)} = \mathbf{R}^{(k)}\mathbf{Q}^{(k)} \\
\texttt{end}
\end{array}
$$

Facts:

- $\mathbf{A}^{(k)}$ is similar to $\mathbf{A}$
- Eigenvalues of $\mathbf{A}^{(k)}$ should be easier to compute than that of $\mathbf{A}$.
- $\mathbf{A}^{(k)}$ should converge fast (expected) to a form whose eigenvalues are easily computed.
  - upper triangular form

The subspace iteration is **equivalent** to QR iteration when applied to a full set of vectors ($r = n$).

Subspace Iteration

$$
\begin{aligned}
\underline{\mathbf{Q}}^{(0)} &= \mathbf{I} \\
\mathbf{Z} &= \mathbf{A}\underline{\mathbf{Q}}^{(k-1)} \\
\mathbf{Z} &= \underline{\mathbf{Q}}^{(k)}\mathbf{R}^{(k)} \\
\mathbf{A}^{(k)} &= (\underline{\mathbf{Q}}^{(k)})^T\mathbf{A}\underline{\mathbf{Q}}^{(k)}
\end{aligned}
$$

QR Iteration

$$
\begin{aligned}
\underline{\mathbf{A}}^{(0)} &= \mathbf{A} \\
\mathbf{A}^{(k-1)} &= \mathbf{Q}^{(k)}\mathbf{R}^{(k)} \\
\mathbf{A}^{(k)} &= \mathbf{R}^{(k)}\mathbf{Q}^{(k)} \\
\underline{\mathbf{Q}}^{(k)} &= \mathbf{Q}^{(1)}\mathbf{Q}^{(2)}\cdots\mathbf{Q}^{(k)}
\end{aligned}
$$

Theorem [Equivalence of Subspace iteration with QR iteration]

The above subspace iteration and QR iteration generate identical sequences of matrices $\mathbf{R}^{(k)}$, $\mathbf{Q}^{(k)}$, and $\mathbf{A}^{(k)}$ defined by the QR factorization of the k-th power of $\mathbf{A}$

$$\mathbf{A}^k = \underline{\mathbf{Q}}^{(k)}\underline{\mathbf{R}}^{(k)},$$

with

$$\mathbf{A}^{(k)} = (\underline{\mathbf{Q}}^{(k)})^T\mathbf{A}\underline{\mathbf{Q}}^{(k)},$$

where

$$\underline{\mathbf{R}}^{(k)} = \mathbf{R}^{(k)}\mathbf{R}^{(k-1)}\cdots\mathbf{R}^{(1)}$$

Challenges of QR Iteration

For an $n \times n$ matrix $\mathbf{A}$, each iteration requires $\mathcal{O}(n^3)$ flops to compute the QR factorization.

- too computationally expensive!

Improvement:

Perform a similarity transform $\mathbf{A}$ to obtain a form $\mathbf{A}^{(0)} = (\mathbf{Q}^{(0)})^H \mathbf{A} \mathbf{Q}^{(0)}$

- the QR decomposition of $\mathbf{A}^{(0)}$ should be computationally cheap
- $\mathbf{A}^{(k)}$ $(k = 1, 2, \cdots)$ should have similar structure with $\mathbf{A}^{(0)}$ so that the QR decomposition at each iteration is computationally cheap

Motivation: perform similarity transform $\mathbf{A}$ to an upper Hessenberg form (zeros below the first subdiagonal), i.e., $\mathbf{Q}^H\mathbf{A}\mathbf{Q} = \mathbf{H}$ where

$$\mathbf{H} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ & \times & \times & \times \\ & & \times & \times \end{bmatrix}$$

Advantage: QR factorization of an upper Hessenberg matrix requires $\mathcal{O}(n^2)$ flops (how?).

- by using Givens rotations

QR Iteration with Hessenberg Reduction:

$\mathbf{A} = \mathbf{Q}^H \mathbf{H} \mathbf{Q}$, $\mathbf{A}^{(0)} = \mathbf{H}$, $\mathbf{H}$ `is upper Hessenberg`
`for` $k = 1, \ 2, \ \cdots$
  $\mathbf{Q}^{(k)} \mathbf{R}^{(k)} = \mathbf{A}^{(k-1)}$ `QR factorization of` $\mathbf{A}^{(k-1)}$
  $\mathbf{A}^{(k)} = \mathbf{R}^{(k)} \mathbf{Q}^{(k)}$
`end`

Key: $\mathbf{A}^{(k)}$ is of upper Hessenberg form (how to preserve?)

- by using Givens rotations to compute the QR factorization (how to prove?)

Benefit: $\mathcal{O}(n^2)$ flops for QR factorization.

For an $n \times n$ matrix $\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 & \mathbf{a}_2 & \cdots & \mathbf{a}_n \end{bmatrix}$.

A Naive Try

Let $\mathbf{Q}_1$ be the Householder reflection matrix that reflects $\mathbf{a}_1$ to $-\text{sign}(\mathbf{a}_1(1))\|\mathbf{a}_1\|_2\mathbf{e}_1$,

$$\mathbf{A} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \rightarrow \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ & \times & \times & \times \\ & \times & \times & \times \\ & \times & \times & \times \end{bmatrix}}_{\mathbf{Q}_1\mathbf{A}} \rightarrow \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ {\color{red}\times} & \times & \times & \times \\ {\color{red}\times} & \times & \times & \times \\ {\color{red}\times} & \times & \times & \times \end{bmatrix}}_{\mathbf{Q}_1\mathbf{A}\mathbf{Q}_1^H}$$

Mission failed!

Less Ambitious Try

Let $\tilde{\mathbf{a}}_1 = \mathbf{A}(2:n,1)$ and $\mathbf{Q}_1$ be the Householder reflection matrix that reflects $\tilde{\mathbf{a}}_1$ to $-\text{sign}(\tilde{\mathbf{a}}_1(1))\|\tilde{\mathbf{a}}_1\|_2\mathbf{e}_1$,

$$\mathbf{A} = \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix} \rightarrow \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ & \times & \times & \times \\ & \times & \times & \times \end{bmatrix}}_{\mathbf{Q}_1\mathbf{A}} \rightarrow \underbrace{\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ & \times & \times & \times \\ & \times & \times & \times \end{bmatrix}}_{\mathbf{Q}_1\mathbf{A}\mathbf{Q}_1^H}$$

Repeat the above procedure to the 2nd column of $\mathbf{Q}_1\mathbf{A}\mathbf{Q}_1^H \cdots$

Given an $n \times n$ matrix $\mathbf{A}$, the following algorithm reduces $\mathbf{A}$ to an upper Hessenberg form.

Hessenberg Reduction:

for $k = 1 : n-2$

$\quad$ $\mathbf{x}$= $\mathbf{A}$(k+1:n, k)

$$\mathbf{v}_k = \mathtt{sign}(\mathbf{x}(1))\|\mathbf{x}\|_2\mathbf{e}_1 + \mathbf{x}$$

$$\mathbf{v}_k = \frac{\mathbf{v}_k}{\|\mathbf{v}_k\|_2}$$

$$\mathbf{A}(k+1:n, k:n) = \mathbf{A}(k+1:n, k:n) - 2\mathbf{v}_k(\mathbf{v}_k^H\mathbf{A}(k+1:n, k:n))$$

$$\mathbf{A}(1:n, k+1:n) = \mathbf{A}(1:n, k+1:n) - 2(\mathbf{A}(1:n, k+1:n)\mathbf{v}_k)\mathbf{v}_k^H$$

end