

# AMS526: Numerical Analysis I (Numerical Linear Algebra)

## Lecture 18: Rayleigh Quotient Iteration and QR Algorithm

Xiangmin Jiao

SUNY Stony Brook

November 13, 2008

# Solving Eigenvalue Problems

- All eigenvalue solvers must be iterative
- Iterative algorithms have multiple facets:
  - 1 Basic idea behind the algorithms
  - 2 Convergence and techniques to speed-up convergence
  - 3 Efficiency of implementation
  - 4 Termination criteria
- We will focus on first two aspects

## Simplification: Real Symmetric Matrices

- We will consider eigenvalue problems for real symmetric matrices, i.e.  $\mathbf{A} = \mathbf{A}^T \in \mathbb{R}^{m \times m}$ , and  $\mathbf{A}\mathbf{x} = \lambda\mathbf{x}$  for  $\mathbf{x} \in \mathbb{R}^m$ 
  - ▶ Note:  $\mathbf{x}^* = \mathbf{x}^T$ , and  $\|\mathbf{x}\| = \sqrt{\mathbf{x}^T \mathbf{x}}$
- $\mathbf{A}$  has real eigenvalues  $\lambda_1, \lambda_2, \dots, \lambda_m$  and orthonormal eigenvectors  $\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_m$ , where  $\|\mathbf{q}_j\| = 1$
- Eigenvalues are often also ordered in a particular way (e.g., ordered from large to small in magnitude)
- In addition, we focus on symmetric tridiagonal form
  - ▶ Why? Because phase 1 of two-phase algorithm reduces matrix into tridiagonal form

# Rayleigh Quotient

- The Rayleigh quotient of  $\mathbf{x} \in \mathbb{R}^m$  is the scalar

$$r(\mathbf{x}) = \frac{\mathbf{x}^T \mathbf{A} \mathbf{x}}{\mathbf{x}^T \mathbf{x}}$$

- For an eigenvector  $\mathbf{x}$ , its Rayleigh quotient is  $r(\mathbf{x}) = \mathbf{x}^T \lambda \mathbf{x} / \mathbf{x}^T \mathbf{x} = \lambda$ , the corresponding eigenvalue of  $\mathbf{x}$
- For general  $\mathbf{x}$ ,  $r(\mathbf{x}) = \alpha$  that minimizes  $\|\mathbf{A}\mathbf{x} - \alpha\mathbf{x}\|_2$ .
- $\mathbf{x}$  is eigenvector of  $\mathbf{A} \iff \nabla r(\mathbf{x}) = \frac{2}{\mathbf{x}^T \mathbf{x}} (\mathbf{A}\mathbf{x} - r(\mathbf{x})\mathbf{x}) = 0$  with  $\mathbf{x} \neq 0$
- $r(\mathbf{x})$  is smooth and  $\nabla r(\mathbf{q}_j) = 0$  for any  $j$ , and therefore is quadratically accurate:

$$r(\mathbf{x}) - r(\mathbf{q}_j) = O(\|\mathbf{x} - \mathbf{q}_j\|^2) \text{ as } \mathbf{x} \rightarrow \mathbf{q}_j \text{ for some } j$$

# Power Iteration

- Simple power iteration for largest eigenvalue

Algorithm: Power Iteration

$\mathbf{v}^{(0)}$  = some unit-length vector

for  $k = 1, 2, \dots$

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

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

$$\lambda^{(k)} = r(\mathbf{v}^{(k)}) = (\mathbf{v}^{(k)})^T \mathbf{A}\mathbf{v}^{(k)}$$

- Termination condition is omitted for simplicity

## Convergence of Power Iteration

- Expand initial  $\mathbf{v}^{(0)}$  in orthonormal eigenvectors  $\mathbf{q}_i$ , and apply  $\mathbf{A}^k$ :

$$\mathbf{v}^{(0)} = a_1 \mathbf{q}_1 + a_2 \mathbf{q}_2 + \cdots + a_m \mathbf{q}_m$$

$$\mathbf{v}^{(k)} = c_k \mathbf{A}^k \mathbf{v}^{(0)}$$

$$= c_k (a_1 \lambda_1^k \mathbf{q}_1 + a_2 \lambda_2^k \mathbf{q}_2 + \cdots + a_m \lambda_m^k \mathbf{q}_m)$$

$$= c_k \lambda_1^k (a_1 \mathbf{q}_1 + a_2 (\lambda_2/\lambda_1)^k \mathbf{q}_2 + \cdots + a_m (\lambda_m/\lambda_1)^k \mathbf{q}_m)$$

- If  $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_m| \geq 0$  and  $\mathbf{q}_1^T \mathbf{v}^{(0)} \neq 0$ , this gives

$$\|\mathbf{v}^{(k)} - (\pm \mathbf{q}_1)\| = O(|\lambda_2/\lambda_1|^k), \quad |\lambda^{(k)} - \lambda_1| = O(|\lambda_2/\lambda_1|^{2k})$$

where  $\pm$  sign is chosen to be sign of  $\mathbf{q}_1^T \mathbf{v}^{(k)}$

- It finds the largest eigenvalue (unless eigenvector orthogonal to  $\mathbf{v}^{(0)}$ )
- Error reduces by only a constant factor ( $\approx |\lambda_2/\lambda_1|$ ) each step, and very slowly especially when  $|\lambda_2| \approx |\lambda_1|$

## Inverse Iteration

- Apply power iteration on  $(\mathbf{A} - \mu \mathbf{I})^{-1}$ , with eigenvalues  $\{(\lambda_j - \mu)^{-1}\}$
- If  $\mu \approx \lambda_J$  for some  $J$ , then  $(\lambda_J - \mu)^{-1}$  may be far larger than  $(\lambda_j - \mu)^{-1}$ ,  $j \neq J$ , so power iteration may converge rapidly

Algorithm: Inverse Iteration

$\mathbf{v}^{(0)}$  = some unit-length vector

for  $k = 1, 2, \dots$

Solve  $(\mathbf{A} - \mu \mathbf{I})\mathbf{w} = \mathbf{v}^{(k-1)}$  for  $\mathbf{w}$

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

$\lambda^{(k)} = r(\mathbf{v}^{(k)}) = (\mathbf{v}^{(k)})^T \mathbf{A} \mathbf{v}^{(k)}$

- Converges to eigenvector  $\mathbf{q}_J$  if parameter  $\mu$  is close to  $\lambda_J$

$$\|\mathbf{v}^{(k)} - (\pm \mathbf{q}_J)\| = O\left(\left|\frac{\mu - \lambda_J}{\mu - \lambda_K}\right|^k\right), \quad |\lambda^{(k)} - \lambda_J| = O\left(\left|\frac{\mu - \lambda_J}{\mu - \lambda_K}\right|^{2k}\right)$$

where  $\lambda_J$  and  $\lambda_K$  are closest and second closest eigenvalues to  $\mu$

- Standard method for determining eigenvector given eigenvalue

# Rayleigh Quotient Iteration

- Parameter  $\mu$  is constant in inverse iteration, but convergence is better for  $\mu$  close to the eigenvalue
- Improvement: At each iteration, set  $\mu$  to last computed Rayleigh quotient

Algorithm: Rayleigh Quotient Iteration

$\mathbf{v}^{(0)}$  = some unit-length vector

$$\lambda^{(0)} = r(\mathbf{v}^{(0)}) = (\mathbf{v}^{(0)})^T \mathbf{A} \mathbf{v}^{(0)}$$

for  $k = 1, 2, \dots$

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

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

$$\lambda^{(k)} = r(\mathbf{v}^{(k)}) = (\mathbf{v}^{(k)})^T \mathbf{A} \mathbf{v}^{(k)}$$

- Cost per iteration is linear for tridiagonal matrix

## Convergence of Rayleigh Quotient Iteration

- Cubic convergence in Rayleigh quotient iteration

$$\|\mathbf{v}^{(k+1)} - (\pm \mathbf{q}_J)\| = O(\|\mathbf{v}^{(k)} - (\pm \mathbf{q}_J)\|^3)$$

and

$$|\lambda^{(k+1)} - \lambda_J| = O(|\lambda^{(k)} - \lambda_J|^3)$$

- In other words, each iteration triples number of digits of accuracy
- Proof idea: If  $\mathbf{v}^{(k)}$  is close to an eigenvector,  $\|\mathbf{v}^{(k)} - (\pm \mathbf{q}_J)\| \leq \epsilon$ , then accuracy of Rayleigh quotient estimate  $\lambda^{(k)}$  is  $|\lambda^{(k)} - \lambda_J| = O(\epsilon^2)$ . One step of inverse iteration then gives

$$\|\mathbf{v}^{(k+1)} - \mathbf{q}_J\| = O(|\lambda^{(k)} - \lambda_J| \|\mathbf{v}^{(k)} - \mathbf{q}_J\|) = O(\epsilon^3)$$

- Rayleigh quotient is great in finding largest (or smallest) eigenvalue and its corresponding eigenvector. What if we want to find all eigenvalues?

# QR Algorithm

- Most basic version of QR algorithm is remarkably simple:

Algorithm: “Pure” QR Algorithm

$$\mathbf{A}^{(0)} = \mathbf{A}$$

for  $k = 1, 2, \dots$

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

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

- With some suitable assumptions,  $\mathbf{A}^{(k)}$  converge to Schur form of  $\mathbf{A}$  (diagonal if  $\mathbf{A}$  is symmetric)
- Similarity transformation of  $\mathbf{A}$ :

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

# Unnormalized Simultaneous Iteration

- To understand QR algorithm, first consider simple algorithm
- Simultaneous Iteration is power iteration applied to several vectors
- Start with linearly independent  $\mathbf{v}_1^{(0)}, \dots, \mathbf{v}_n^{(0)}$
- We know from power iteration that  $\mathbf{A}^k \mathbf{v}_1^{(0)}$  converge to  $\mathbf{q}_1$
- With some assumptions, the space  $\langle \mathbf{A}^k \mathbf{v}_1^{(0)}, \dots, \mathbf{A}^k \mathbf{v}_n^{(0)} \rangle$  should converge to  $\langle \mathbf{q}_1, \dots, \mathbf{q}_n \rangle$
- Notation: Define initial matrix  $\mathbf{V}^{(0)}$  and matrix  $\mathbf{V}^{(k)}$  at step  $k$ :

$$\mathbf{V}^{(0)} = \left[ \mathbf{v}_1^{(0)} \mid \dots \mid \mathbf{v}_n^{(0)} \right], \quad \mathbf{V}^{(k)} = \mathbf{A}^k \mathbf{V}^{(0)} = \left[ \mathbf{v}_1^{(k)} \mid \dots \mid \mathbf{v}_n^{(k)} \right]$$