Lecture 16
The QR Algorithm II

MIT 18.335J / 6.337J

Introduction to Numerical Methods

Per-Olof Persson

November 2, 2006

Simultaneous *Inverse* Iteration $\Longleftrightarrow$ QR Algorithm

- Last lecture we showed that "pure" QR $\Longleftrightarrow$ simultaneous iteration applied to I, and the first column evolves as in power iteration

- But it is also equivalent to simultaneous *inverse* iteration applied to a "flipped" I, and the last column evolves as in inverse iteration

- To see this, recall that $A^k = \underline{Q}^{(k)}\underline{R}^{(k)}$ with

$$\underline{Q}^{(k)} = \prod_{j=1}^{k} Q^{(j)} = \left[\begin{array}{c|c|c|c} q_1^{(k)} & q_2^{(k)} & \cdots & q_m^{(k)} \end{array} \right]$$

- Invert and use that A^{-1} is symmetric:

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

Simultaneous *Inverse* Iteration $\Longleftrightarrow$ QR Algorithm

- Introduce the "flipping" permutation matrix

$$P = \begin{bmatrix} & & & 1 \\ & & 1 & \\ & \cdots & & \\ 1 & & & \end{bmatrix}$$

and rewrite that last expression as

$$A^{-k}P = [\underline{Q}^{(k)}P][P(\underline{R}^{(k)})^{-T}P]$$

- This is a QR factorization of $A^{-k}P$, and the algorithm is equivalent to simultaneous iteration on A^{-1}
- In particular, the last column of $\underline{Q}^{(k)}$ evolves as in inverse iteration

The Shifted QR Algorithm

- Since the QR algorithm behaves like inverse iteration, introduce shifts $\mu^{(k)}$ to accelerate the convergence:

$$A^{(k-1)} - \mu^{(k)} I = Q^{(k)} R^{(k)}$$
$$A^{(k)} = R^{(k)} Q^{(k)} + \mu^{(k)} I$$

- We then get (same as before):

$$A^{(k)} = (Q^{(k)})^T A^{(k-1)} Q^{(k)} = (\underline{Q}^{(k)})^T A \underline{Q}^{(k)}$$

and (different from before):

$$(A - \mu^{(k)} I)(A - \mu^{(k-1)} I) \cdots (A - \mu^{(1)} I) = \underline{Q}^{(k)} \underline{R}^{(k)}$$

- Shifted simultaneous iteration – last column of $\underline{Q}^{(k)}$ converges quickly

Choosing $\mu^{(k)}$: The Rayleigh Quotient Shift

- Natural choice of $\mu^{(k)}$: Rayleigh quotient for last column of $\underline{Q}^{(k)}$

$$\mu^{(k)} = \frac{(q_m^{(k)})^T A q_m^{(k)}}{(q_m^{(k)})^T q_m^{(k)}} = (q_m^{(k)})^T A q_m^{(k)}$$

- Rayleigh quotient iteration, last column $q_m^{(k)}$ converges cubically
- Convenient fact: This Rayleigh quotient appears as m, m entry of $A^{(k)}$ since $A^{(k)} = (\underline{Q}^{(k)})^T A \underline{Q}^{(k)}$
- The *Rayleigh quotient shift* corresponds to setting $\mu^{(k)} = A_{mm}^{(k)}$

Choosing $\mu^{(k)}$: The Wilkinson Shift

- The QR algorithm with Rayleigh quotient shift might fail, e.g. with two symmetric eigenvalues
- Break symmetry by the *Wilkinson shift*

$$\mu = a_m - \mathrm{sign}(\delta) b_{m-1}^2 \Big/ \left(|\delta| + \sqrt{\delta^2 + b_{m-1}^2} \right)$$

where $\delta = (a_{m-1} - a_m)/2$ and $B = \begin{bmatrix} a_{m-1} & b_{m-1} \\ b_{m-1} & a_m \end{bmatrix}$ is the lower-right submatrix of $A^{(k)}$

- Always convergence with this shift, in worst case quadratically

A Practical Shifted QR Algorithm

Algorithm: “Practical” QR Algorithm

$(Q^{(0)})^T A^{(0)} Q^{(0)} = A$ — $A^{(0)}$ is a tridiagonalization of A

for $k = 1, 2, \ldots$

Pick a shift $\mu^{(k)}$ — e.g., choose $\mu^{(k)} = A_{mm}^{(k-1)}$

$Q^{(k)} R^{(k)} = A^{(k-1)} - \mu^{(k)} I$ — QR factorization of $A^{(k-1)} - \mu^{(k)} I$

$A^{(k)} = R^{(k)} Q^{(k)} + \mu^{(k)} I$ — Recombine factors in reverse order

If any off-diagonal element $A_{j,j+1}^{(k)}$ is sufficiently close to zero,

set $A_{j,j+1} = A_{j+1,j} = 0$ to obtain

$$\begin{bmatrix} A_1 & 0 \\ 0 & A_2 \end{bmatrix} = A^{(k)}$$

and now apply the QR algorithm to A_1 and A_2

Stability and Accuracy

- The QR algorithm is backward stable:

$$\tilde{Q}\tilde{\Lambda}\tilde{Q}^T = A + \delta A, \qquad \frac{\|\delta A\|}{\|A\|} = O(\epsilon_{\text{machine}})$$

where $\tilde{\Lambda}$ is the computed Λ and $\tilde{Q}$ is an exactly orthogonal matrix

- The combination with Hessenberg reduction is also backward stable
- Can be shown (for normal matrices) that $|\tilde{\lambda}_j - \lambda_j| \leq \|\delta A\|_2$, which gives

$$\frac{|\tilde{\lambda}_j - \lambda_j|}{\|A\|} = O(\epsilon_{\text{machine}})$$

where $\tilde{\lambda}_j$ are the computed eigenvalues