QR-method lecture 1

SF2524 - Matrix Computations for Large-scale Systems

So far we have in the course learned about...

Methods suitable for large sparse matrices

- Power method: largest eigenvalue
- Inverse iteration: eigenvalue closest to a target
- Rayleigh Quotient Iteration: One eigenvalue based on starting guess
- Arnoldi method for eigenvalue problems:
  - Outer isolated eigenvalues
  - Only requires matrix vector products Ay
  - Underlying the matlab command: `eigs`

Now: QR-method

- Underlying the matlab command: `eig`
- Computes all eigenvalues
- Suitable for dense problems
- Small matrices in comparison to previous algorithms

Agenda QR-method

1. Decompositions
   - Jordan form
   - Schur decomposition
   - QR-factorization
2. Basic QR-method
3. Improvement 1: Two-phase approach
   - Hessenberg reduction
   - Hessenberg QR-method
4. Improvement 2: Acceleration with shifts
5. Convergence theory

Reading instructions

Point 1: TB Lecture 24 (previous courses)
Points 2-4: Lecture notes PDF
Point 5: Lecture notes PDF (TB Chapter 28)
(Extra reading: TB Chapter 25-26, 28-29)

Similarity transformation

Suppose $A \in \mathbb{C}^{m \times m}$ and $V \in \mathbb{C}^{m \times m}$ is an invertible matrix. Then

$$A$$

and

$$B = VAV^{-1}$$

have the same eigenvalues.

Numerical methods based on similarity transformations

- If B is triangular we can read-off the eigenvalues from the diagonal.
- Idea of numerical method: Compute V such that B is triangular.

First idea: compute the Jordan canonical form

Jordan canonical form (JCF)

Suppose $A \in \mathbb{C}^{m\times m}$. There exists an invertible matrix $V \in \mathbb{C}^{m\times m}$ and a block diagonal matrix such that

$$A = V\Lambda V^{-1}$$

where

$$\Lambda = \begin{pmatrix} J_1 & & \\ & \ddots & \\ & & J_k \end{pmatrix},$$

where

$$J_i = \begin{pmatrix} \lambda_i & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{pmatrix}, \quad i = 1, \ldots, k$$

Common case: distinct eigenvalues

Suppose $\lambda_i \neq \lambda_j$, $i = 1, \ldots, m$. Then,

$$\Lambda = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_m \end{pmatrix}.$$

Common case: symmetric matrix

Suppose $A = A^T \in \mathbb{R}^{m \times m}$. Then,

$$\Lambda = \begin{pmatrix} \lambda_1 & & \\ & \ddots & \\ & & \lambda_m \end{pmatrix}.$$

Example - numerical stability of Jordan form

Consider

$$A = \begin{pmatrix} 2 & 1 & \\ & 2 & 1 \\ \varepsilon & & 2 \end{pmatrix}$$

If $\varepsilon = 0$. Then, the Jordan canonical form (JCF) is

$$\Lambda = \begin{pmatrix} 2 & 1 & \\ & 2 & 1 \\ & & 2 \end{pmatrix}.$$

If $\varepsilon > 0$. Then, the eigenvalues are distinct and

$$\Lambda = \begin{pmatrix} 2 + O(\varepsilon^{1/3}) & & \\ & 2 + O(\varepsilon^{1/3}) & \\ & & 2 + O(\varepsilon^{1/3}) \end{pmatrix}.$$

$\Rightarrow$ JCF not continuous with respect to ε

$\Rightarrow$ JCF is often not numerically stable

Schur decomposition (essentially TB Theorem 24.9)

Suppose $A \in \mathbb{C}^{m\times m}$. There exists an unitary matrix P

$$P^{-1} = P^*$$

and a triangular matrix T such that

$$A = PTP^*.$$

The Schur decomposition is numerically stable.

Goal with QR-method: Numercally compute a Schur factorization

Outline:

1. Decompositions
   - Jordan form
   - Schur decomposition
   - QR-factorization
2. **Basic QR-method**
3. Improvement 1: Two-phase approach
   - Hessenberg reduction
   - Hessenberg QR-method
4. Improvement 2: Acceleration with shifts
5. Convergence theory

QR-factorization

Suppose $A \in \mathbb{C}^{m\times m}$. There exists a unitary matrix $Q \in \mathbb{C}^{m\times m}$ and an upper triangular matrix $R \in \mathbb{C}^{m\times m}$ such that

$$A = QR$$

Note: Very different from Schur factorization

$$A = QTQ^*$$

- QR-factorization can be computed with a finite number of operations
- Schur decomposition directly gives us the eigenvalues

Basic QR-method

Didactic simplifying assumption: All eigenvalues are real

Basic QR-method = basic QR-algorithm

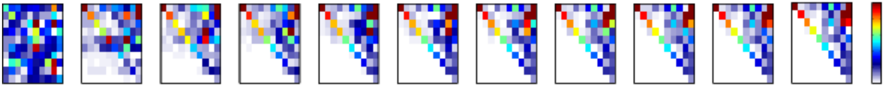
Simple basic idea: Let $A_0 = A$ and iterate:

- Compute QR-factorization of $A_k = QR$
- Set $A_{k+1} = RQ$.

Note:

- $A_1 = RQ = Q^* A_0 Q \Rightarrow A_0, A_1, \ldots$ have the same eigenvalues
- More remarkable: $A_k \to$ triangular matrix (except special cases)

$A_k \rightarrow$ triangular matrix:



* Time for matlab demo *

Elegant and robust but not very efficient:

Disadvantages

- Computing a QR-factorization is quite expensive. One iteration of the basic QR-method

$$\mathcal{O}(m^3).$$

- The method often requires many iterations.

Improvement demo:

`http://www.youtube.com/watch?v=qmgxzsWWsNc`