QR-method lecture 3

SF2524 - Matrix Computations for Large-scale Systems

Agenda QR-method

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

Agenda QR-method

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

Improvement 1: Two-phase approach (recap)

We will separate the computation into two phases:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \underset{\text{Phase 1}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix} \underset{\text{Phase 2}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Improvement 1: Two-phase approach (recap)

We will separate the computation into two phases:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \underset{\text{Phase 1}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix} \underset{\text{Phase 2}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Phases

- Phase 1: Reduce the matrix to a Hessenberg with similarity transformations (Section 2.2.1 in lecture notes)

Improvement 1: Two-phase approach (recap)

We will separate the computation into two phases:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \underset{\text{Phase 1}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix} \underset{\text{Phase 2}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Phases

- Phase 1: Reduce the matrix to a Hessenberg with similarity transformations (Section 2.2.1 in lecture notes)
- Phase 2: Specialize the QR-method to Hessenberg matrices (Section 2.2.2 in lecture notes)

Improvement 1: Two-phase approach (recap)

We will separate the computation into two phases:

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \end{bmatrix} \underset{\text{Phase 1}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \end{bmatrix} \underset{\text{Phase 2}}{\rightarrow} \begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & & \times & \times & \times \\ & & & \times & \times \\ & & & & \times \end{bmatrix}$$

Phases

- Phase 1: Reduce the matrix to a Hessenberg with similarity transformations (Section 2.2.1 in lecture notes)
- Phase 2: Specialize the QR-method to Hessenberg matrices (Section 2.2.2 in lecture notes) $\leftarrow$ **NOW**

Phase 2: Hessenberg QR-method

A QR-step on a Hessenberg matrix is a Hessenberg matrix:

* Matlab demo showing QR-step: `hessenberg_is_hessenberg.m` *

Phase 2: Hessenberg QR-method

A QR-step on a Hessenberg matrix is a Hessenberg matrix:

* Matlab demo showing QR-step: `hessenberg_is_hessenberg.m` *

Theorem (Theorem 2.2.4)

If the basic QR-method is applied to a Hessenberg matrix, then all iterates A_k are Hessenberg matrices.

Phase 2: Hessenberg QR-method

A QR-step on a Hessenberg matrix is a Hessenberg matrix:

* Matlab demo showing QR-step: `hessenberg_is_hessenberg.m` *

Theorem (Theorem 2.2.4)

If the basic QR-method is applied to a Hessenberg matrix, then all iterates A_k are Hessenberg matrices.

Recall: basic QR-step is $\mathcal{O}(m^3)$.

Phase 2: Hessenberg QR-method

A QR-step on a Hessenberg matrix is a Hessenberg matrix:

* Matlab demo showing QR-step: `hessenberg_is_hessenberg.m` *

Theorem (Theorem 2.2.4)

If the basic QR-method is applied to a Hessenberg matrix, then all iterates A_k are Hessenberg matrices.

Recall: basic QR-step is $\mathcal{O}(m^3)$.

Hessenberg structure can be exploited such that we can carry out a QR-step with less operations.

Definition (Givens rotation)

The matrix $G(i,j,c,s) \in \mathbb{R}^{n\times n}$ where $c^2 + s^2 = 1$ corresponding to a Givens rotation is defined by

$$G(i,j,c,s) := \begin{bmatrix} I & & & & \\ & c & & -s & \\ & & I & & \\ & s & & c & \\ & & & & I \end{bmatrix},$$

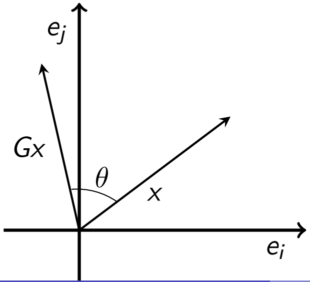
which deviates from identity at row and column i and j.

Definition (Givens rotation)

The matrix $G(i,j,c,s) \in \mathbb{R}^{n\times n}$ where $c^2+s^2=1$ corresponding to a Givens rotation is defined by

$$G(i,j,c,s) := \begin{bmatrix} I & & & & \\ & c & & -s & \\ & & I & & \\ & s & & c & \\ & & & & I \end{bmatrix},$$

which deviates from identity at row and column i and j.

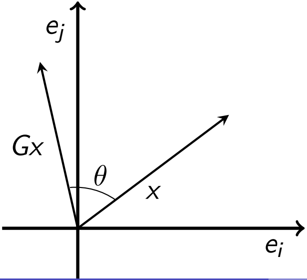


Definition (Givens rotation)

The matrix $G(i,j,c,s) \in \mathbb{R}^{n\times n}$ where $c^2 + s^2 = 1$ corresponding to a Givens rotation is defined by

$$G(i,j,c,s) := \begin{bmatrix} I & & & & \\ & c & & -s & \\ & & I & & \\ & s & & c & \\ & & & & I \end{bmatrix},$$

which deviates from identity at row and column i and j.



Properties

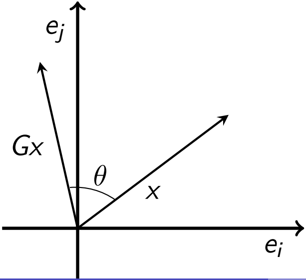
- $G^* = G^{-1}$

Definition (Givens rotation)

The matrix $G(i,j,c,s) \in \mathbb{R}^{n\times n}$ where $c^2+s^2=1$ corresponding to a Givens rotation is defined by

$$G(i,j,c,s) := \begin{bmatrix} I & & & & \\ & c & & -s & \\ & & I & & \\ & s & & c & \\ & & & & I \end{bmatrix},$$

which deviates from identity at row and column i and j.



Properties

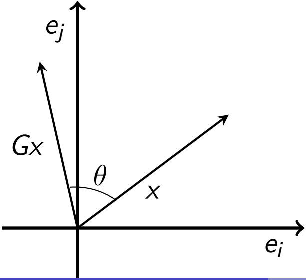
- $G^* = G^{-1}$
- Gz can be computed with $O(1)$ operations

Definition (Givens rotation)

The matrix $G(i,j,c,s) \in \mathbb{R}^{n\times n}$ where $c^2+s^2=1$ corresponding to a Givens rotation is defined by

$$G(i,j,c,s) := \begin{bmatrix} I & & & & \\ & c & & -s & \\ & & I & & \\ & s & & c & \\ & & & & I \end{bmatrix},$$

which deviates from identity at row and column i and j.



Properties

- $G^* = G^{-1}$
- Gz can be computed with $O(1)$ operations
- $\cdots$ (show on white board)

The Q-matrix in the QR-factorization of a Hessenberg matrix can be factorized as a product of $n-1$ Givens rotators.

The Q-matrix in the QR-factorization of a Hessenberg matrix can be factorized as a product of $n-1$ Givens rotators.

Theorem (Theorem 2.2.6)

Suppose $A \in \mathbb{C}^{n\times n}$ *is a Hessenberg matrix. Let* H_i *be generated as follows* $H_1 = A$

$$H_{i+1} = G_i^T H_i, \quad i = 1, \ldots, n-1$$

where

$$G_i = G(i, i+1, (H_i)_{i,i}/r_i, (H_i)_{i+1,i}/r_i)$$

and $r_i = \sqrt{(H_i)_{i,i}^2 + (H_i)_{i+1,i}^2}$ *and we assume* $r_i \neq 0$. *Then,* H_n *is upper triangular and*

$$A = (G_1 G_2 \cdots G_{n-1}) H_n = QR$$

is a QR-factorization of A.

The Q-matrix in the QR-factorization of a Hessenberg matrix can be factorized as a product of $n-1$ Givens rotators.

Theorem (Theorem 2.2.6)

Suppose $A \in \mathbb{C}^{n\times n}$ *is a Hessenberg matrix. Let* H_i *be generated as follows*
$H_1 = A$

$$H_{i+1} = G_i^T H_i, \quad i = 1, \dots, n-1$$

where

$$G_i = G(i, i+1, (H_i)_{i,i}/r_i, (H_i)_{i+1,i}/r_i)$$

and $r_i = \sqrt{(H_i)_{i,i}^2 + (H_i)_{i+1,i}^2}$ *and we assume* $r_i \neq 0$. *Then,* H_n *is upper triangular and*

$$A = (G_1 G_2 \cdots G_{n-1}) H_n = QR$$

is a QR-factorization of A.

* Matlab: Explicit QR-factorization of Hessenberg `qr_givens.m` *

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators: Let

$$A_{k-1} = (G_1 G_2 \cdots G_{n-1}) R_n$$

and

$$A_k = R_n (G_1 G_2 \cdots G_{n-1}) =$$

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators: Let

$$A_{k-1} = (G_1 G_2 \cdots G_{n-1}) R_n$$

and

$$A_k = R_n (G_1 G_2 \cdots G_{n-1}) = (\cdots((R_n G_1) G_2) \cdots) G_{n-1}$$

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators: Let

$$A_{k-1} = (G_1 G_2 \cdots G_{n-1}) R_n$$

and

$$A_k = R_n (G_1 G_2 \cdots G_{n-1}) = (\cdots((R_n G_1) G_2) \cdots) G_{n-1}$$

Complexity of one QR-step for a Hessenberg matrix

We need to apply $2(n-1)$ givens rotators to compute one QR-step.

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators: Let

$$A_{k-1} = (G_1 G_2 \cdots G_{n-1}) R_n$$

and

$$A_k = R_n (G_1 G_2 \cdots G_{n-1}) = (\cdots((R_n G_1) G_2) \cdots) G_{n-1}$$

Complexity of one QR-step for a Hessenberg matrix

We need to apply $2(n-1)$ givens rotators to compute one QR-step.

- One givens rotator applied to a vector can be computed in $O(1)$ operations.

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators: Let

$$A_{k-1} = (G_1 G_2 \cdots G_{n-1}) R_n$$

and

$$A_k = R_n (G_1 G_2 \cdots G_{n-1}) = (\cdots((R_n G_1) G_2) \cdots) G_{n-1}$$

Complexity of one QR-step for a Hessenberg matrix

We need to apply $2(n-1)$ givens rotators to compute one QR-step.

- One givens rotator applied to a vector can be computed in $O(1)$ operations.
- One givens rotator applied to matrix can be computed in $O(n)$ operations.

Idea of Hessenberg QR-method: Do not explicitly compute the Q-matrix but only implicitly apply the Givens rotators: Let

$$A_{k-1} = (G_1 G_2 \cdots G_{n-1}) R_n$$

and

$$A_k = R_n (G_1 G_2 \cdots G_{n-1}) = (\cdots((R_n G_1) G_2) \cdots) G_{n-1}$$

Complexity of one QR-step for a Hessenberg matrix

We need to apply $2(n-1)$ givens rotators to compute one QR-step.

- One givens rotator applied to a vector can be computed in $O(1)$ operations.
- One givens rotator applied to matrix can be computed in $O(n)$ operations.

$\Rightarrow$

$$\text{the complexity of one Hessenberg QR step} = \mathcal{O}(n^2)$$

Givens rotators only modify very few elements.
Several optimizations possible. $\Rightarrow$

Algorithm 3 Hessenberg QR algorithm

Input: A Hessenberg matrix $A \in \mathbb{C}^{n \times n}$

Output: Upper triangular T such that $A = UTU^*$ for an orthogonal matrix U.

Set $A_0 := A$
for $k = 1, \ldots$ **do**
  // One Hessenberg QR step
  $H = A_{k-1}$
  for $i = 1, \ldots, n-1$ **do**
    $[c_i, s_i] = \text{givens}(h_{i,i}, h_{i+1,i})$
    $H_{i:i+1,i:n} = \begin{bmatrix} c_i & s_i \\ -s_i & c_i \end{bmatrix} H_{i:i+1,i:n}$
  end for
  for $i = 1, \ldots, n-1$ **do**
    $H_{1:i+1,i:i+1} = H_{1:i+1,i:i+1} \begin{bmatrix} c_i & -s_i \\ s_i & c_i \end{bmatrix}$
  end for
  $A_k = H$
end for
Return $T = A_\infty$