Numerical methods for eigenvalue problems

D. Löchel
Supervisors: M. Hochbruck und M. Tokar

Mathematisches Institut
Heinrich-Heine-Universität Düsseldorf

GRK 1203 seminar february 2008

Outline

Introduction

Linear Algebra:

Definition

$A \in \mathbb{C}^{n\times n}$, eigenvector $\vec{x} \in \mathbb{C}^n \setminus \{0\}$, eigenvalue $\lambda \in \mathbb{C}$

$$A\vec{x} = \lambda\vec{x}$$

Theorem

λ eigenvalue of A $\Longleftrightarrow P_A(\lambda) := det(\lambda I - A) = 0$

- P_A characteristic polynomial
- $\deg(P_A) \geq 5$: no explicit formula $\longrightarrow$ numerical algorithm required
- using $P_A(\lambda)$ numerical instable

Introduction

Definition

Rayleigh quotient: $\rho_A(\vec{x}) = \frac{\vec{x}^H A \vec{x}}{\vec{x}^H \vec{x}}$

Theorem

$\vec{x}$ eigenvector of $A \Longrightarrow \rho_A(\vec{x})$ is eigenvalue of A

Definition

- numbering of eigenvalues: $|\lambda_1| \geq |\lambda_2| \geq \ldots \geq |\lambda_n|$
- $\vec{x}_j$ eigenvector for λ_j

Power Method

Given: $A \in \mathbb{C}^{n\times n},\ \vec{y}_0 \in \mathbb{C}^n \setminus \{0\},\ \|\vec{y}_0\|_2 = 1$
Assume: $\vec{y}_0^H \vec{x}_1 \neq 0$
Idea:

$$\vec{y}_0 = \sum_{j=1}^{n} \beta_j \vec{x}_j$$

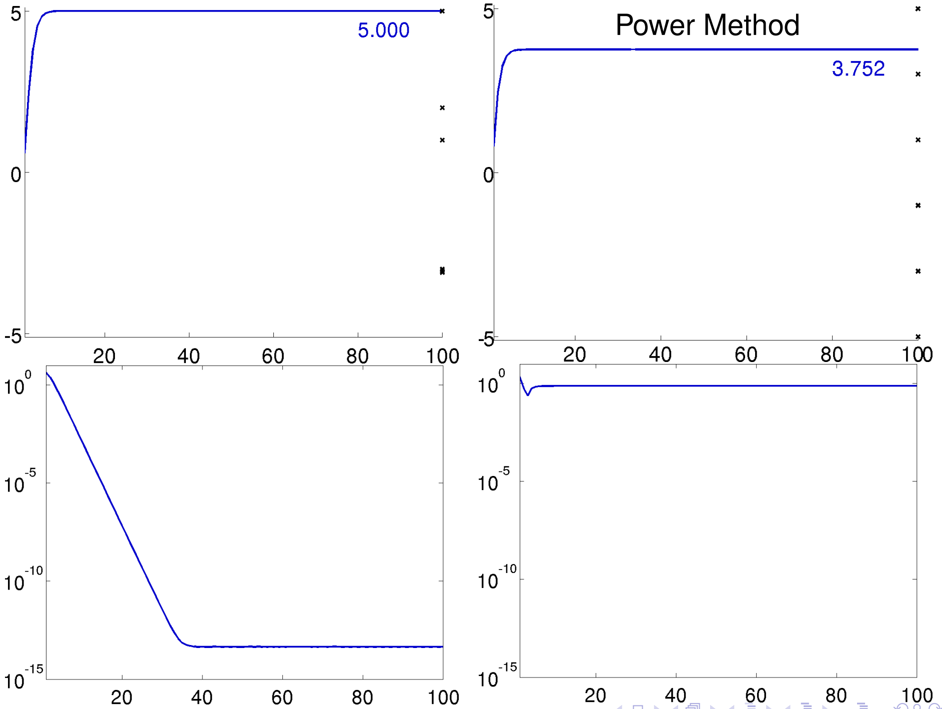
$$\Longrightarrow A\vec{y}_0 = \sum_{j=1}^{n} \beta_j A\vec{x}_j = \sum_{j=1}^{n} \beta_j \lambda_j \vec{x}_j$$

$$\Longrightarrow A^k \vec{y}_0 = \sum_{j=1}^{n} \beta_j \lambda_j^k \vec{x}_j = \beta_1 \lambda_1^k \vec{x}_1 + \mathcal{O}\left(\left| \frac{\lambda_2}{\lambda_1} \right|^k \right)$$

Power Method

Initial guess $\|\vec{y}_0\|_2 = 1$.
for $k = 0, 1, \ldots$ **do**
$\vec{z}_{k+1} = A\vec{y}_k$ // power
$\rho_k = \vec{y}_k^H \vec{z}_{k+1}$ // Rayleigh quotient $\frac{\vec{y}_k^H A \vec{y}_k}{\vec{y}_k^H \vec{y}_k}$
$\vec{y}_{k+1} = \frac{\vec{z}_{k+1}}{\|\vec{z}_{k+1}\|_2}$ // avoid over and underflow
end for

- Convergence if $\eta := \left|\frac{\lambda_2}{\lambda_1}\right| < 1$ and $\vec{x}_1^H \vec{y}_0 \neq 0$.
- Approximation $\rho(\vec{y}_k) = \lambda_1 + \mathcal{O}(\eta^k)$, slow if $\eta \approx 1$.
- Finds λ_1 only.



Power Method
5.000
3.752
5
0
-5
20
40
60
80
100
10^{0}
10^{-5}
10^{-10}
10^{-15}

Interior Eigenvalues

Recall: $\lambda \neq 0$ eigenvalue of A $\iff \frac{1}{\lambda}$ eigenvalue of A^{-1}.

Idea: apply power method to $A^{-1} \rightarrow$ inverse power method.

Definition

LU decomposition: $A = LU$, L lower, U upper triangular matrix

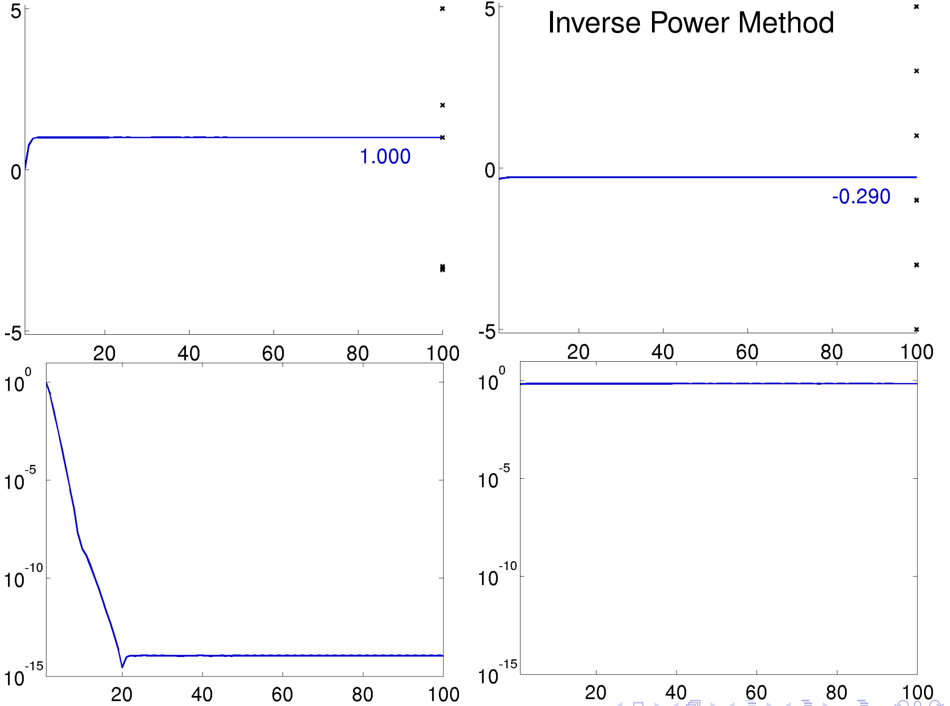
$$\begin{aligned} A\vec{z} &= \vec{y} \\ \iff L(U\vec{z}) &= \vec{y} \\ \iff \quad L\vec{t} &= \vec{y}, \; U\vec{z} = \vec{t} \end{aligned}$$

two triangular linear systems

Inverse Power Method

Initial guess $\|\vec{y}_0\|_2 = 1$.
Calculate LU decomposition of A.
for $k = 0, 1, \ldots$ **do**
  Solve $A\vec{z}_{k+1} = \vec{y}_k$ with LU decomposition. // $\vec{z}_{k+1} = A^{-1}\vec{y}_k$
  $\vec{y}_{k+1} = \frac{\vec{z}_{k+1}}{\|\vec{z}_{k+1}\|_2}$ // avoid over and underflow
  $\rho_k = \rho_A(\vec{y}_{k+1}) = \vec{y}_{k+1}^H A\vec{y}_{k+1}$ // Rayleigh quotient
end for

- Finds eigenvalue closest to zero.
- Convergence if $\eta := \left|\frac{\lambda_n}{\lambda_{n-1}}\right| < 1$ and $\vec{x}_n^H\vec{y}_0 \neq 0$.
- Approximation $\rho(\vec{y}_k) = \lambda_n + \mathcal{O}(\eta^k)$, slow if $\eta \approx 1$.


Inverse Power Method
1.000
-0.290
5
0
-5
20
40
60
80
100
10^{0}
10^{-5}
10^{-10}
10^{-15}

Shifts

What about the remaining eigenvalues?

Smallest eigenvalue of $A - \mu I$ is eigenvalue from A closest to μ.

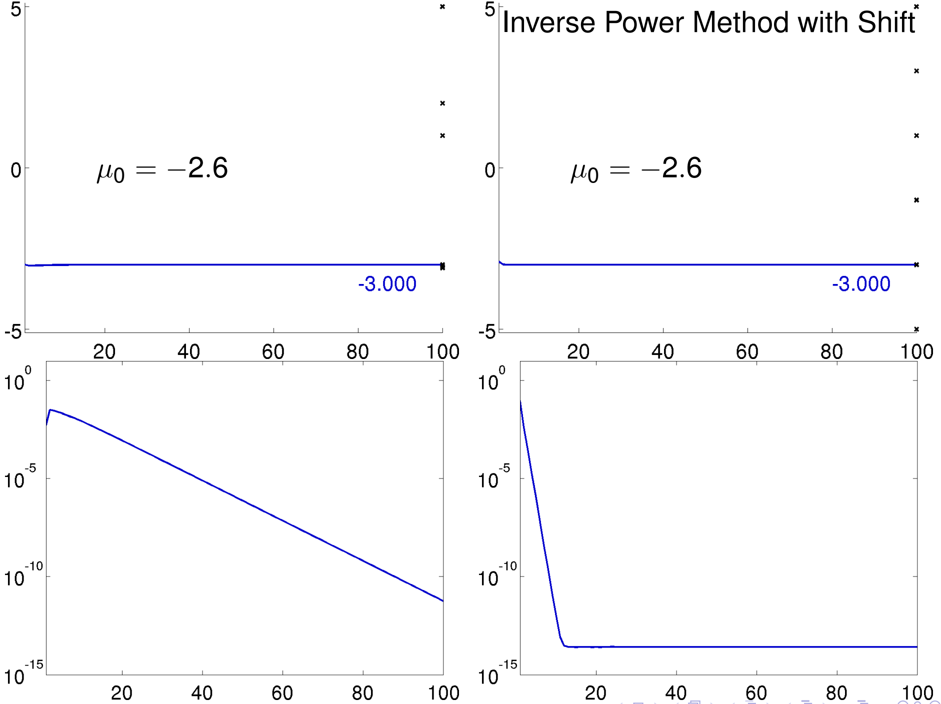
Definition

μ is called Shift and $A - \mu I$ shifted matrix.

Inverse Power Method with Shift

Initial guess $\|\vec{y}_0\|_2 = 1$.
μ_0 initial guess for desired eigenvalue.
Calculate LU decomposition of $(A - \mu_0 I)$.
for $k = 0, 1, \ldots$ **do**
  Solve $(A - \mu_0 I)\vec{z}_{k+1} = \vec{y}_k$ with LU decomposition.
  $\vec{y}_{k+1} = \frac{\vec{z}_{k+1}}{\|\vec{z}_{k+1}\|_2}$ // avoid over and underflow
  $\rho_k = \rho_A(\vec{y}_{k+1}) = \vec{y}_{k+1}^H A \vec{y}_{k+1}$ // Rayleigh quotient
end for

- Finds eigenvalue λ_j closest to μ_0.
- Convergence if $\eta := \max_{m \neq j} \left| \frac{\lambda_j - \mu_0}{\lambda_m - \mu_0} \right| < 1$ and $\vec{x}_j^H \vec{y}_0 \neq 0$.
- Approximation $\rho(\vec{y}_k) = \lambda_j + \mathcal{O}(\eta^k)$, slow if $\eta \approx 1$.

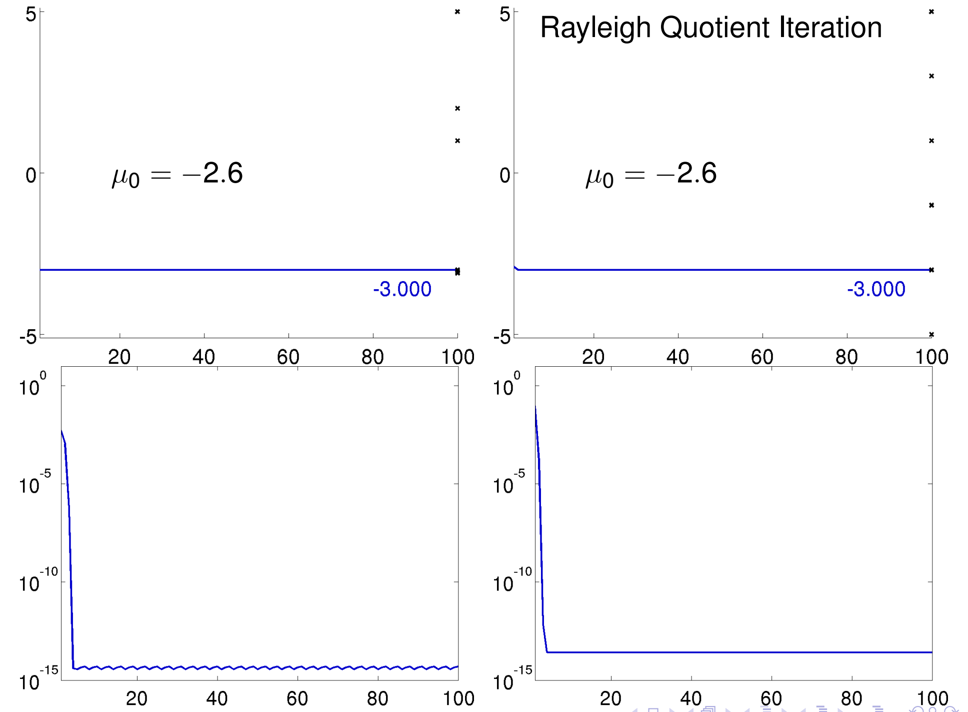


Inverse Power Method with Shift
$\mu_0 = -2.6$
$\mu_0 = -2.6$
-3.000
-3.000
5
0
-5
20
40
60
80
100
10^{0}
10^{-5}
10^{-10}
10^{-15}

Rayleigh Quotient Iteration

Initial guess $\|\vec{y}_0\|_2 = 1$.
μ_0 initial guess for desired eigenvalue.
for $k = 0, 1, \ldots$ **do**
  Calculate LU decomposition of $(A - \mu_k I)$.
  Solve $(A - \mu_k I)\vec{z}_{k+1} = \vec{y}_k$ with LU decomposition.
  $\vec{y}_{k+1} = \frac{\vec{z}_{k+1}}{\|\vec{z}_{k+1}\|_2}$ // avoid over and underflow
  $\rho_k = \rho_A(\vec{y}_{k+1}) = \vec{y}_{k+1}^H A \vec{y}_{k+1}$ // Rayleigh quotient
  $\mu_k := \rho_k$
end for

- Finds eigenvalue λ_j closest to μ_0.
- Convergence if $\eta_0 := \max_{m \neq j} \left| \frac{\lambda_j - \mu_0}{\lambda_m - \mu_0} \right| < 1$ and $\vec{x}_j^H \vec{y}_0 \neq 0$.
- $\eta_k := \max_{m \neq j} \left| \frac{\lambda_j - \mu_k}{\lambda_m - \mu_k} \right| \ll 1$,   if $\eta_0 < 1$: $\lim_{k \to \infty} \eta_k = 0$.
- Approximation $\rho(\vec{y}_k) = \lambda_j + \mathcal{O}(\eta_0^{2k})$.
- LU decompostion in each step expensive
- Convergence fast

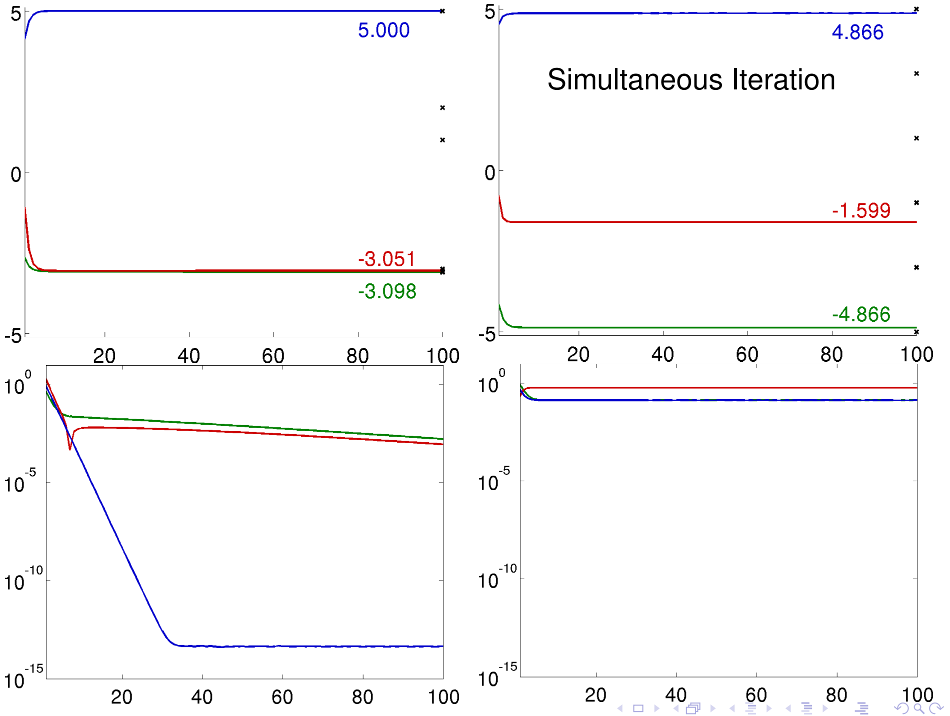

Rayleigh Quotient Iteration
$\mu_0 = -2.6$
-3.000
$\mu_0 = -2.6$
-3.000

Simultaneous Iteration

Until now: calculating only one eigenvalue
Next step: calculating many or all eigenvalues

$m \leq n$, $U_0 \in \mathbb{C}^{n,m}$ unitary, $U_0^H U_0 = I_m$, $A_0 := A$
for $k = 0, 1, \ldots$ **do**
  $Y_{k+1} := A_k U_k$ // power of each column of U_k
  Calculate QR decomposition $Y_{k+1} = U_{k+1} R_{k+1}$.
  $A_{k+1} = U_{k+1}^H A U_{k+1}$ // similarity transformation
end for

- Orthogonalisation is necessary to avoid that all columns converge against x_1.



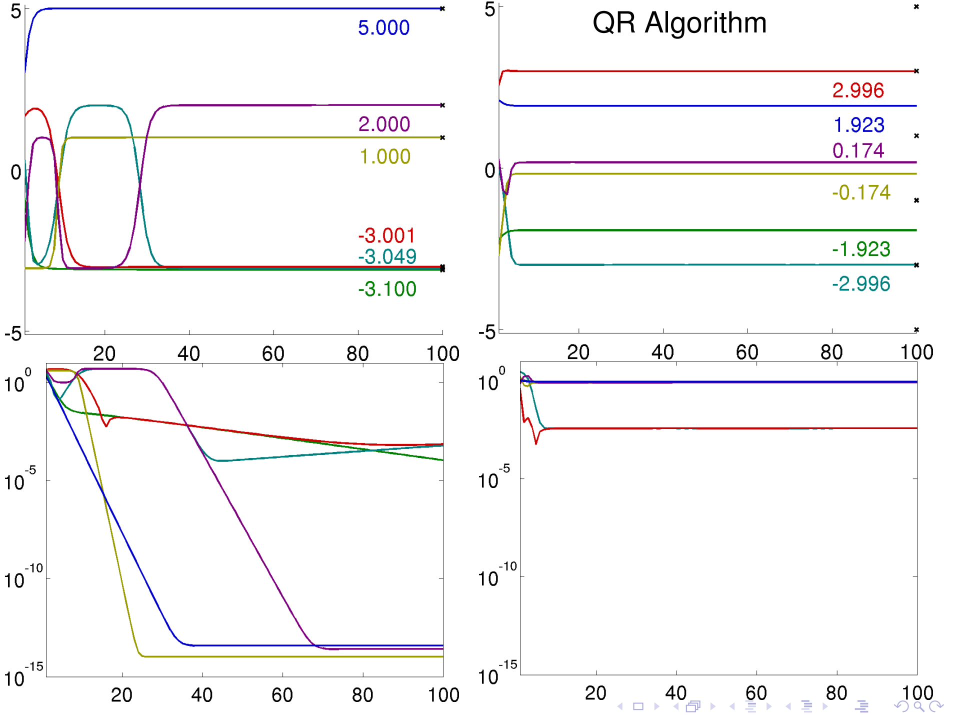
Simultaneous Iteration
5.000
-3.051
-3.098
4.866
-1.599
-4.866

QR Algorithm

$m = n$: QR algorithm (equivalent to simultaneous iteration)

$A_0 := A$
for $k = 0, 1, \ldots$ **do**
  Calculate QR Decomposition $A_k = Q_k R_k$.
  $A_{k+1} := R_k Q_k$
end for

- $A_{k+1} = R_k Q_k = Q_k^H A_k Q_k = (Q_{k-1} Q_k)^H A_{k-1} (Q_{k-1} Q_k) = \widehat{Q}_k^H A_0 \widehat{Q}_k$, unitary similarity transformation.
- $A_k \longrightarrow$ upper triangular matrix, eigenvalues on diagonal (proof: schur decomposition).
- This way QR algorithm can be very slow.

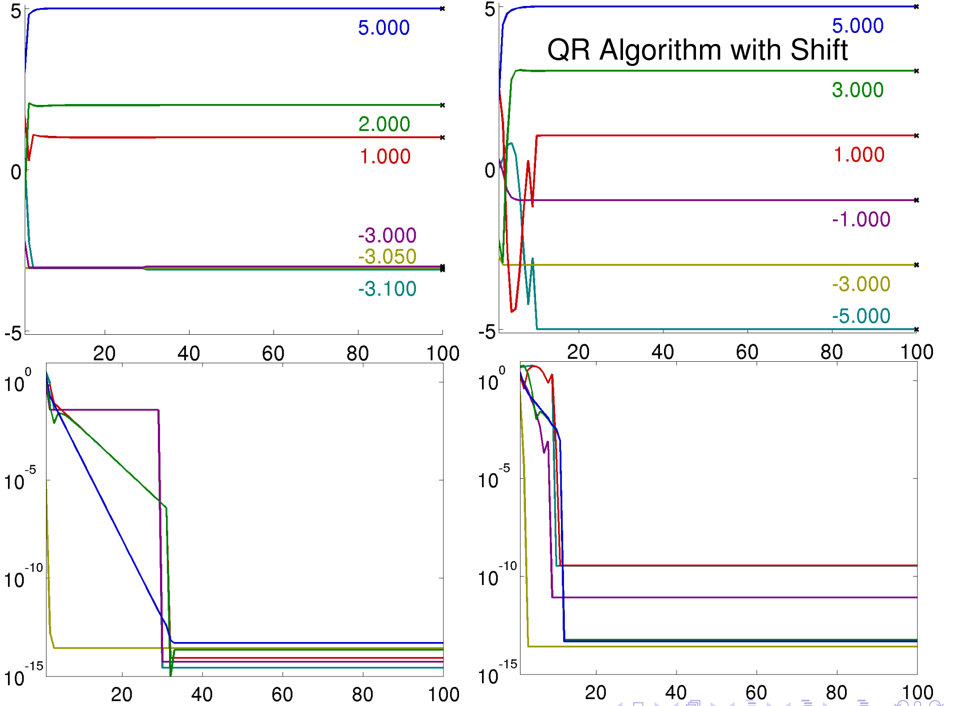


QR Algorithm
5.000
2.000
1.000
-3.001
-3.049
-3.100
2.996
1.923
0.174
-0.174
-1.923
-2.996

QR Algorithm with Shift

$A_0 := A$
for $k = 0, 1, \ldots$ **do**
  Choose a shift μ_k.
  Calculate QR decomposition $A_k - \mu_k I = Q_k R_k$.
  $A_{k+1} := R_k Q_k + \mu_k I$
end for

- this way convergence is fast in general



QR Algorithm with Shift
5.000
2.000
1.000
-3.000
-3.050
-3.100
5.000
3.000
1.000
-1.000
-3.000
-5.000

QR Implementation

- complexity of one QR decomposition: $\frac{2}{3}n^3$ operations
- complexity to calculate all eigenvalues: at least $\mathcal{O}(n^4)$
- reduce complexity for QR decomposition by $A \leftarrow H := U^H A U$ with H upper Hessenberg.

$$U^H A U = H = \begin{bmatrix} h_{1,1} & \dots & h_{1,n-1} & h_{1,n} \\ h_{2,1} & \ddots & \vdots & \vdots \\ & \ddots & h_{n-1,n-1} & h_{n-1,n} \\ 0 & & h_{n,n-1} & h_{n,n} \end{bmatrix}$$

U product of Givens-Rotations

$$G = \begin{bmatrix} \cos(\alpha) & \sin(\alpha) \\ -\sin(\alpha) & \cos(\alpha) \end{bmatrix}$$

Summary

- power method for biggest eigenvalue
- inverse power method for eigenvalue closest to zero.
- inverse power method with shift for desired eigenvalue
- Rayleigh quotient iteration, fast
- simultaneous iteration for many eigenvalues
- QR algorithm (with shift) for all eigenvalues (fast)

Thank you for your attention!