Computing Eigenvalues and/or Eigenvectors;Part 2, The Power method and QR-algorithm

Tom Lyche

Centre of Mathematics for Applications,
Department of Informatics,
University of Oslo

November 19, 2010

Today

- The power method to find the dominant eigenvector
- The shifted power method to speed up convergence
- The inverse power method
- The Rayleigh quotient iteration
- The QR-algorithm

The Power Method

- Find the eigenvector corresponding to the dominant (largest in absolute value) eigenvalue.
- With a simple modification we can also find the corresponding eigenvalue

Assumptions

- Let $\mathbf{A} \in \mathbb{C}^{n,n}$ have eigenpairs $(\lambda_j, \mathbf{v}_j)$, $j = 1, \ldots, n$.
- Given $\mathbf{z}_0 \in \mathbb{C}^n$ we assume that

$$
\begin{aligned}
&(i) \quad |\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \cdots \geq |\lambda_n|, \\
&(ii) \quad \mathbf{z}_0^T \mathbf{v}_1 \neq 0. \\
&(iii) \quad \mathbf{A} \text{ has linearly independent eigenvectors .}
\end{aligned}
$$

- The first assumption means that $\mathbf{A}$ has a dominant eigenvalue λ_1 of algebraic multiplicity one.
- The second assumption says that $\mathbf{z}_0$ has a component in the direction $\mathbf{v}_1$.
- The third assumption is not necessary. It is included to simplify the analysis.

Powers

- Given $\mathbf{A} \in \mathbb{C}^{n,n}$, a vector $\mathbf{z}_0 \in \mathbb{C}^n$, and assume that i),ii), iii) hold.
- Define a sequence $\{\mathbf{z}_k\}$ of vectors in $\mathbb{C}^n$ by $\mathbf{z}_k := \mathbf{A}^k \mathbf{z}_0 = \mathbf{A}\mathbf{z}_{k-1}, \quad k = 1, 2, \ldots.$
- $\mathbf{z}_0 = c_1\mathbf{v}_1 + c_2\mathbf{v}_2 + \cdots + c_n\mathbf{v}_n$, with $c_1 \neq 0$.
- $\mathbf{A}^k\mathbf{v}_j = \lambda_j^k\mathbf{v}_j$, $k = 0, 1, 2, \ldots$, $j = 1, \ldots, n$.
- Then $\mathbf{z}_k = c_1\mathbf{A}^k\mathbf{v}_1 + c_2\mathbf{A}^k\mathbf{v}_2 + \cdots + c_n\mathbf{A}^k\mathbf{v}_n, \quad k = 0, 1, 2, \ldots.$
- $\mathbf{z}_k = c_1\lambda_1^k\mathbf{v}_1 + c_2\lambda_2^k\mathbf{v}_2 + \cdots + c_n\lambda_n^k\mathbf{v}_n, \quad k = 0, 1, 2, \ldots.$
- $\frac{\mathbf{z}_k}{\lambda_1^k} = c_1\mathbf{v}_1 + c_2\left(\frac{\lambda_2}{\lambda_1}\right)^k\mathbf{v}_2 + \cdots + c_n\left(\frac{\lambda_n}{\lambda_1}\right)^k\mathbf{v}_n.$
- $\mathbf{z}_k/\lambda_1^k, \to c_1\mathbf{v}_1, \ k \to \infty$

The Power method

Need to normalize the vectors $\mathbf{z}_k$.

- Do not know λ_1.
- Choose a norm on $\mathbb{C}^n$, set $\mathbf{x}_0 = \mathbf{z}_0/\|\mathbf{z}_0\|$ and generate for $k = 1, 2, \ldots$ unit vectors ,$\{\mathbf{x}_k\}$ as follows:

$$\begin{aligned} (i) \quad & \mathbf{y}_k = \mathbf{A}\mathbf{x}_{k-1} \\ (ii) \quad & \mathbf{x}_k = \mathbf{y}_k/\|\mathbf{y}_k\|. \end{aligned} \tag{1}$$

Example1

- $$\mathbf{A} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{z}_0 = [1.0, 1.0]^T, \quad \mathbf{x}_0 = [0.707, 0.707]^T$$
- $\mathbf{x}_1 = [0.39, 0.92]$, $\mathbf{x}_2 = [0.4175, 0.9087]$, $\mathbf{x}_3 = [0.4159, 0.9094], \ldots$
- converges to an eigenvector of $\mathbf{A}$

Example2

The way $\{\mathbf{x}_k\}$ converges to an eigenvector can be more complicated.

- $$\mathbf{A} = \begin{bmatrix} -1 & -2 \\ -3 & -4 \end{bmatrix}, \quad \mathbf{z}_0 = [1.0, 1.0]^T, \quad \mathbf{x}_0 = [0.707, 0.707]^T$$
- $\mathbf{x}_1 = [-0.39, -0.92]$, $\mathbf{x}_2 = [0.4175, 0.9087]$, $\mathbf{x}_3 = [-0.4159, -0.9094], \ldots$
- changes sign in each iteration.

Convergence

Lemma

Suppose (i), (ii), (iii) hold. Then

$$\lim_{k\to\infty} \Big(\frac{|\lambda_1|}{\lambda_1}\Big)^k \mathbf{x}_k = \frac{c_1}{|c_1|}\frac{\mathbf{v}_1}{\|\mathbf{v}_1\|}.$$

In particular, if $\lambda_1 > 0$ and $c_1 > 0$ then the sequence $\{\mathbf{x}_k\}$ will converge to the eigenvector $\mathbf{u}_1 := \mathbf{v}_1/\|\mathbf{v}_1\|$ of unit length.

- In Example 2, $\lambda_1 < 0$ and $c_1 > 0$ if $\mathbf{u}_1 = [0.4159, 0.9094]^T$. For k large:
- $\mathbf{x}_k \approx \big(\frac{\lambda_1}{|\lambda_1|}\big)^k \frac{c_1}{|c_1|}\mathbf{u}_1 = (-1)^k \mathbf{u}_1$.

Eigenvalue

- Suppose we know an approximate eigenvector of $\mathbf{A} \in \mathbb{C}^{n,n}$. How should we estimate the corresponding eigenvalue?
- If $(\lambda, \mathbf{u})$ is an exact eigenpair then $\mathbf{Au} - \lambda\mathbf{u} = \mathbf{0}$.
- If $\mathbf{u}$ is an approximate eigenvector we can minimize the function $\rho : \mathbb{C} \to \mathbb{R}$ given by

$$\rho(\mu) := \|\mathbf{Au} - \mu\mathbf{u}\|_2.$$

Theorem

ρ is minimized when $\mu = \nu := \frac{\mathbf{u}^*\mathbf{Au}}{\mathbf{u}^*\mathbf{u}}$ is the Rayleigh quotient of $\mathbf{u}$.

Proof on blackboard.

Power with Rayleigh

```
function [l,x,it]=powerit(A,z,K,tol)
af=norm(A,'fro'); x=z/norm(z);
for k=1:K
    y=A*x; l=x'*y;
    if norm(y-l*x)/af < tol
      it=k; x=y/norm(y); return
    end
    x=y/norm(y);
end
it=K+1;
```

Example

- $$\mathbf{A}_1 := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \quad \mathbf{A}_2 := \begin{bmatrix} 1.7 & -0.4 \\ 0.15 & 2.2 \end{bmatrix}, \text{ and } \mathbf{A}_3 = \begin{bmatrix} 1 & 2 \\ -3 & 4 \end{bmatrix}.$$
- Start with a random vector and tol$=10^{-6}$.
- Get convergence in 7 iterations for $\mathbf{A}_1$, 174 iterations for $\mathbf{A}_2$ and no convergence for $\mathbf{A}_3$.
- $\mathbf{A}_3$ has two complex eigenvalues so assumption i) is not satisfied
- Rate of convergence depends on $r = |\lambda_2/\lambda_1|$. Faster convergence for smaller r.
- We have $r \approx 0.07$ for $\mathbf{A}_1$ and $r = 0.95$ for $\mathbf{A}_2$.

The shifted power method

- A variant of the power method is the shifted power method.
- In this method we choose a number s and apply the power method to the matrix $\mathbf{A} - s\mathbf{I}$.
- The number s is called a shift since it shifts an eigenvalue λ of $\mathbf{A}$ to $\lambda - s$ of $\mathbf{A} - s\mathbf{I}$.
- Sometimes the convergence can be faster if the shift is chosen intelligently.
- For example, for $\mathbf{A}_2$ with shift $s = 1.8$, we get convergence in 17 iterations instead of 174 without shift.

The inverse power method

- We apply the power method to the inverse matrix $(\mathbf{A} - s\mathbf{I})^{-1}$, where s is a shift.
- If $\mathbf{A}$ has eigenvalues $\lambda_1, \ldots, \lambda_n$ in no particular order then $(\mathbf{A} - s\mathbf{I})^{-1}$ has eigenvalues

$$\mu_1(s) = (\lambda_1 - s)^{-1}, \mu_2(s) = (\lambda_2 - s)^{-1}, \ldots, \mu_n(s) = (\lambda_n - s)^{-1}.$$

- Suppose λ_1 is a simple eigenvalue of $\mathbf{A}$.
- Then $\lim_{s \to \lambda_1} |\mu_1(s)| = \infty$, while $\lim_{s \to \lambda_1} \mu_j(s) = (\lambda_j - \lambda_1)^{-1} < \infty$ for $j = 2, \ldots, n$.
- Hence, by choosing s sufficiently close to λ_1 the inverse power method will converge to that eigenvalue.

For the inverse power method (1) is replaced by.

$$\begin{aligned} (i) \quad & (\mathbf{A} - s\mathbf{I})\mathbf{y}_k = \mathbf{x}_{k-1} \\ (ii) \quad & \mathbf{x}_k = \mathbf{y}_k / \|\mathbf{y}_k\|. \end{aligned} \tag{2}$$

Note that we solve the linear system rather than computing the inverse matrix. Normally the PLU-factorization of $\mathbf{A} - s\mathbf{I}$ is pre-computed in order to speed up the iteration.

Rayleigh quotient iteration

We can combine inverse power with Rayleigh quotient calculation.

$$
\begin{aligned}
(i) \quad & (\mathbf{A} - s_{k-1}\mathbf{I})\mathbf{y}_k = \mathbf{x}_{k-1}, \\
(ii) \quad & \mathbf{x}_k = \mathbf{y}_k / \|\mathbf{y}_k\|, \\
(iii) \quad & \mathbf{s}_k = \mathbf{x}_k^* \mathbf{A}\mathbf{x}_k, \\
(iv) \quad & \mathbf{r}_k = \mathbf{A}\mathbf{x}_k - s_k \mathbf{x}_k.
\end{aligned}
$$

- We can avoid the calculation of $\mathbf{A}\mathbf{x}_k$ in (iii) and (iv).

Example

- $\mathbf{A}_1 := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$.
- Try to find the smallest eigenvalue $\lambda = (5 - \sqrt{33})/2 \approx -0.37$
- Start with $\mathbf{x} = [1, 1]^T$ and $s = 0$

k	1	2	3	4	5
$\|\mathbf{r}\|_2$	1.0e+000	7.7e-002	1.6e-004	8.2e-010	2.0e-020
$\|s_k - \lambda\|$	3.7e-001	-1.2e-002	-2.9e-005	-1.4e-010	-2.2e-016

Table: Quadratic convergence of Rayleigh quotient iteration

Problem with singularity?

- The linear system in i) becomes closer and closer to singular as s_k converges to the eigenvalue.
- Thus the system becomes more and more ill-conditioned and we can expect large errors in the computed $\mathbf{y}_k$.
- This is indeed true, but we are lucky.
- Most of the error occurs in the direction of the eigenvector and this error disappears when we normalize $\mathbf{y}_k$ in ii).
- Miraculously, the normalized eigenvector will be quite accurate.

Discussion

- Since the shift changes from iteration to iteration the computation of $\mathbf{y}$ will require $O(n^3)$ flops for a full matrix.
- For such a matrix it might pay to reduce it to a upper Hessenberg form or tridiagonal form before starting the iteration.
- However, if we have a good approximation to an eigenpair then only a few iterations are necessary to obtain close to machine accuracy.
- If Rayleigh quotient iteration converges the convergence will be quadratic and sometimes even cubic.

The QR Algorithm

- An iterative method to compute all eigenvalues and eigenvectors of a matrix $\mathbf{A} \in \mathbb{C}^{n,n}$.
- The matrix is reduced to triangular or quasitriangular form by a sequence of unitary similarity transformations computed from the QR factorization of $\mathbf{A}$.
- Recall that for a square matrix the QR factorization and the QR decomposition are the same.
- If $\mathbf{A} = \mathbf{QR}$ is a QR factorization then $\mathbf{Q} \in \mathbb{C}^{n,n}$ is unitary, $\mathbf{Q}^*\mathbf{Q} = \mathbf{I}$ and $\mathbf{R} \in \mathbb{C}^{n,n}$ is upper triangular.

Basic QR

$$
\begin{aligned}
&\mathbf{A}_1 = \mathbf{A} \\
&\text{for } k = 1, 2, \ldots \\
&\quad \mathbf{Q}_k \mathbf{R}_k = \mathbf{A}_k \qquad \text{(QR factorization of } \mathbf{A}_k\text{)} \\
&\quad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k. \\
&\text{end}
\end{aligned}
\tag{3}
$$

The determination of the QR factorization of $\mathbf{A}_k$ and the computation of $\mathbf{R}_k \mathbf{Q}_k$ is called a QR step.

Example

- $\mathbf{A}_1 = \mathbf{A} = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$
- $\mathbf{A}_1 = \left(\frac{1}{\sqrt{5}} \begin{bmatrix} 2 & -1 \\ 1 & 2 \end{bmatrix}\right) * \left(\frac{1}{\sqrt{5}} \begin{bmatrix} 5 & 4 \\ 0 & 3 \end{bmatrix}\right) = \mathbf{Q}_1\mathbf{R}_1.$
- $\mathbf{A}_2 = \mathbf{R}_1\mathbf{Q}_1 = \frac{1}{5} \begin{bmatrix} 5 & 4 \\ 0 & 3 \end{bmatrix} * \begin{bmatrix} 2 & -1 \\ 1 & 2 \end{bmatrix} = \frac{1}{5} \begin{bmatrix} 14 & 3 \\ 3 & 6 \end{bmatrix} = \begin{bmatrix} 2.8 & 0.6 \\ 0.6 & 1.2 \end{bmatrix}.$
- $\mathbf{A}_4 \approx \begin{bmatrix} 2.997 & -0.074 \\ -0.074 & 1.0027 \end{bmatrix},$
- $\mathbf{A}_{10} \approx \begin{bmatrix} 3.0000 & -0.0001 \\ -0.0001 & 1.0000 \end{bmatrix}.$
- $\mathbf{A}_{10}$ is almost diagonal and contains approximations to the eigenvalues $\lambda_1 = 3$ and $\lambda_2 = 1$ on the diagonal.

Example 2

$$\mathbf{A}_1 = \mathbf{A} = \begin{bmatrix} 0.9501 & 0.8913 & 0.8214 & 0.9218 \\ 0.2311 & 0.7621 & 0.4447 & 0.7382 \\ 0.6068 & 0.4565 & 0.6154 & 0.1763 \\ 0.4860 & 0.0185 & 0.7919 & 0.4057 \end{bmatrix}$$

we obtain

$$\mathbf{A}_{14} = \left[\begin{array}{r|rr|r} 2.323 & 0.047223 & -0.39232 & -0.65056 \\ \hline -2.1e-10 & 0.13029 & 0.36125 & 0.15946 \\ -4.1e-10 & -0.58622 & 0.052576 & -0.25774 \\ \hline 1.2e-14 & 3.3e-05 & -1.1e-05 & 0.22746 \end{array}\right].$$

$\mathbf{A}_{14}$ is close to quasi-triangular.

Example 2

$$\mathbf{A}_{14} = \left[\begin{array}{r|rr|r} 2.323 & 0.047223 & -0.39232 & -0.65056 \\ \hline -2.1e-10 & 0.13029 & 0.36125 & 0.15946 \\ -4.1e-10 & -0.58622 & 0.052576 & -0.25774 \\ \hline 1.2e-14 & 3.3e-05 & -1.1e-05 & 0.22746 \end{array}\right].$$

- The 1×1 blocks give us two real eigenvalues $\lambda_1 \approx 2.323$ and $\lambda_4 \approx 0.2275$.
- The middle 2×2 block has complex eigenvalues resulting in $\lambda_2 \approx 0.0914 + 0.4586i$ and $\lambda_3 \approx 0.0914 - 0.4586i$.
- From Gerschgorin's circle theorem it follows that the approximations to the real eigenvalues are quite accurate.
- We would also expect the complex eigenvalues to have small absolute errors.

Why QR works

- Since $\mathbf{Q}_k^* \mathbf{A}_k = \mathbf{R}_k$ we obtain

$$\mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k = \mathbf{Q}_k^* \mathbf{A}_k \mathbf{Q}_k. \tag{4}$$

- Thus $\mathbf{A}_{k+1}$ is similar to $\mathbf{A}_k$ and hence to $\mathbf{A}$.
- It combines both the power method and the Rayleigh quotient iteration.
- If $\mathbf{A} \in \mathbb{R}^{n,n}$ has real eigenvalues, then under fairly general conditions, the sequence $\{\mathbf{A}_k\}$ converges to an upper triangular matrix, the Schur form.
- If $\mathbf{A}$ is real, but with some complex eigenvalues, then the convergence will be to the quasi-triangular Schur form

QR factorization of $\mathbf{A}^k$

Theorem

For $k = 1, 2, 3, \ldots$, the QR factorization of $\mathbf{A}^k$ is $\mathbf{A}^k = \tilde{\mathbf{Q}}_k \tilde{\mathbf{R}}_k$, where

$$\tilde{\mathbf{Q}}_k := \mathbf{Q}_1 \cdots \mathbf{Q}_k \text{ and } \tilde{\mathbf{R}}_k := \mathbf{R}_k \cdots \mathbf{R}_1, \tag{5}$$

and $\mathbf{Q}_1, \ldots, \mathbf{Q}_k$, $\mathbf{R}_1, \ldots, \mathbf{R}_k$ are the matrices generated by the basic QR algorithm (3)*. Moreover,*

$$\mathbf{A}_k = \tilde{\mathbf{Q}}_{k-1}^* \mathbf{A} \tilde{\mathbf{Q}}_{k-1}, \quad k \geq 1. \tag{6}$$

Proof on blackboard.

Relation to the Power Method

- Since $\tilde{\mathbf{R}}_k$ is upper triangular its first column is a multiple of $\mathbf{e}_1$
-

$$\mathbf{A}^k\mathbf{e}_1 = \tilde{\mathbf{Q}}_k\tilde{\mathbf{R}}_k\mathbf{e}_1 = \tilde{r}_{11}^{(k)}\tilde{\mathbf{Q}}_k\mathbf{e}_1 \text{ or } \tilde{\mathbf{q}}_1^{(k)} := \tilde{\mathbf{Q}}_k\mathbf{e}_1 = \frac{1}{\tilde{r}_{11}^{(k)}}\mathbf{A}^k\mathbf{e}_1.$$

- Since $\|\tilde{\mathbf{q}}_1^{(k)}\|_2 = 1$ the first column of $\tilde{\mathbf{Q}}_k$ is the result of applying the normalized power iteration to the starting vector $\mathbf{x}_0 = \mathbf{e}_1$.
- If this iteration converges we conclude that the first column of $\tilde{\mathbf{Q}}_k$ must converge to a dominant eigenvector of $\mathbf{A}$.

Initial reduction to Hessenberg form

- One QR step requires $O(n^3)$ flops for a matrix $\mathbf{A}$ of order n.
- By an initial reduction of $\mathbf{A}$ to upper Hessenberg form $\mathbf{H}_1$, the cost of a QR step can be reduced to $O(n^2)$.

Invariance of the Hessenberg form; I

- Consider a QR step on $\mathbf{H}_1$.
- We determine plane rotations $\mathbf{P}_{i,i+1}$, $i = 1, \ldots, n-1$ so that $\mathbf{P}_{n-1,n} \cdots \mathbf{P}_{1,2}\mathbf{H}_1 = \mathbf{R}_1$ is upper triangular.
-

$$\begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{bmatrix} \xrightarrow{\mathbf{P}_{1,2}} \left[\begin{array}{c|ccc} x & x & x & x \\ \hline 0 & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{array}\right] \xrightarrow{\mathbf{P}_{2,3}} \left[\begin{array}{cc|cc} x & x & x & x \\ 0 & x & x & x \\ \hline 0 & 0 & x & x \\ 0 & 0 & x & x \end{array}\right]$$

$$\xrightarrow{\mathbf{P}_{3,4}} \left[\begin{array}{ccc|c} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ \hline 0 & 0 & 0 & x \end{array}\right].$$

Invariance of the Hessenberg form; II

- $\mathbf{H}_1 = \mathbf{Q}_1\mathbf{R}_1$, where $\mathbf{Q}_1 = \mathbf{P}_{1,2}^* \cdots \mathbf{P}_{n-1,n}^*$ is a QR factorization of $\mathbf{H}_1$.
- To finish the QR step we compute $\mathbf{R}_1\mathbf{Q}_1 = \mathbf{R}_1\mathbf{P}_{1,2}^* \cdots \mathbf{P}_{n-1,n}^*$.
- This postmultiplication step is illustrated by the Wilkinson diagram

$$\mathbf{R}_1 = \begin{bmatrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \end{bmatrix} \xrightarrow{\mathbf{P}_{12}^*} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & 0 & x & x \\ 0 & 0 & 0 & x \end{bmatrix} \xrightarrow{\mathbf{P}_{23}^*} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & 0 & x \end{bmatrix} \xrightarrow{\mathbf{P}_{34}^*} \begin{bmatrix} x & x & x & x \\ x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & x & x \end{bmatrix}.$$

Invariance of the Hessenberg form; III

- If $\mathbf{A}_k$ is upper Hessenberg then $\mathbf{A}_{k+1}$ is upper Hessenberg.
- One QR step requires $O(n^2)$ flops.
- If $\mathbf{A}$ is tridiagonal and symmetric then one QR step requires $O(n)$ flops.

Deflation

- If a subdiagonal element $a_{i+1,i}$ of an upper Hessenberg matrix $\mathbf{A}$ is equal to zero, then the eigenvalues of $\mathbf{A}$ are the union of the eigenvalues of the two smaller matrices $A(1:i, 1:i)$ and $A(i+1:n, i+1:n)$.
- Thus if during the iteration the $(i+1, i)$ element of $\mathbf{A}_k$ is sufficiently small then we can continue the iteration on the two smaller submatrices separately.

Effect on $\mathbf{A}$ of deflation

- suppose $|a_{i+1,i}^{(k)}| \leq \epsilon$.
- $\hat{\mathbf{A}}_k := \mathbf{A}_k - a_{i+1,i}^{(k)} \mathbf{e}_{i+1} \mathbf{e}_i^T$
- $\mathbf{A}_k = \tilde{\mathbf{Q}}_{k-1}^* \mathbf{A} \tilde{\mathbf{Q}}_{k-1}$
- $\hat{\mathbf{A}}_k = \tilde{\mathbf{Q}}_{k-1}^* (\mathbf{A} + \mathbf{E}) \tilde{\mathbf{Q}}_{k-1}$,
- $\mathbf{E} = \tilde{\mathbf{Q}}_{k-1} (a_{i+1,i}^{(k)} \mathbf{e}_{i+1} \mathbf{e}_i^T) \tilde{\mathbf{Q}}_{k-1}^*$.
- $\|\mathbf{E}\|_F = \|a_{i+1,i}^{(k)} \mathbf{e}_{i+1} \mathbf{e}_i^T\|_F = |a_{i+1,i}^{(k)}| \leq \epsilon$
- Setting $a_{i+1,i}^{(k)} = 0$ amounts to a perturbation in the original $\mathbf{A}$ of at most ϵ.

The Shifted QR algorithms

Like in the inverse power method it is possible to speed up the convergence by introducing shifts. The explicitly shifted QR algorithm works as follows:

$$
\begin{aligned}
&\mathbf{A}_1 = \mathbf{A} \\
&\text{for } k = 1, 2, \ldots \\
&\quad \text{Choose a shift } s_k \\
&\quad \mathbf{Q}_k \mathbf{R}_k = \mathbf{A}_k - s_k \mathbf{I} \qquad \text{(QR factorization of } \mathbf{A}_k - s\mathbf{I}) \\
&\quad \mathbf{A}_{k+1} = \mathbf{R}_k \mathbf{Q}_k + s_k \mathbf{I}. \\
&\text{end}
\end{aligned}
\tag{7}
$$

Since $\mathbf{R}_k = \mathbf{Q}_k^*(\mathbf{A}_k - s_k\mathbf{I})$

$$\mathbf{A}_{k+1} = \mathbf{Q}_k^*(\mathbf{A}_k - s_k\mathbf{I})\mathbf{Q}_k + s_k\mathbf{I} = \mathbf{Q}_k^*\mathbf{A}_k\mathbf{Q}_k$$

and $\mathbf{A}_{k+1}$ and $\mathbf{A}_k$ are unitary similar.

Relation to the inverse power method

- $\mathbf{A} - s_k \mathbf{I} = \mathbf{Q}_k \mathbf{R}_k \Rightarrow (\mathbf{A} - s_k \mathbf{I})^* = \mathbf{R}_k^* \mathbf{Q}_k^*$
- $(\mathbf{A} - s_k \mathbf{I})^* \mathbf{Q}_k = \mathbf{R}_k^*$
- $(\mathbf{A} - s_k \mathbf{I})^* \mathbf{Q}_k \mathbf{e}_n = \mathbf{R}_k^* \mathbf{e}_n = \overline{r}_{nn}^{(k)} \mathbf{e}_n$.
- $\mathbf{Q}_k \mathbf{e}_n$ is the result of one iteration of the inverse power method to $\mathbf{A}^*$ with shift s_k.

Choice of shifts

1. The shift $s_k := \mathbf{e}_n^T \mathbf{A}_k \mathbf{e}_n$ is called the **Rayleigh quotient shift**.
2. The eigenvalue of the lower right 2×2 corner of $\mathbf{A}_k$ closest to the n, n element of $\mathbf{A}_k$ is called the **Wilkinson shift**. This shift can be used to find complex eigenvalues of a real matrix.
3. The convergence is very fast and at least quadratic both for the Rayleigh quotient shift and the Wilkinson shift.

The Implicitly shifted QR algorithm

1. By doing two QR iterations at a time it is possible to find both real and complex eigenvalues without using complex arithmetic. The corresponding algorithm is called the **implicitly shifted QR algorithm**
2. After having computed the eigenvalues we can compute the eigenvectors in steps. First we find the eigenvectors of the triangular or quasi-triangular matrix. We then compute the eigenvectors of the upper Hessenberg matrix and finally we get the eigenvectors of $\mathbf{A}$.
3. Practical experience indicates that only $O(n)$ iterations are needed to find all eigenvalues of $\mathbf{A}$. Thus both the explicit- and implicit shift QR algorithms are normally $O(n^3)$ algorithms.