Einführung Eigenwerte

Bei der Modellierung von Stabwerken, entstehen folgende Systeme:

$$p = Ax$$

mit $A = ENE^T$, Verschiebungsvektor x und Lastvektor p und Diagonalmatrix N mit den Materialeigenschaften.
Betrachtet man nun das dynamische System:

$$m\ddot{x}(t) = Ax(t)$$

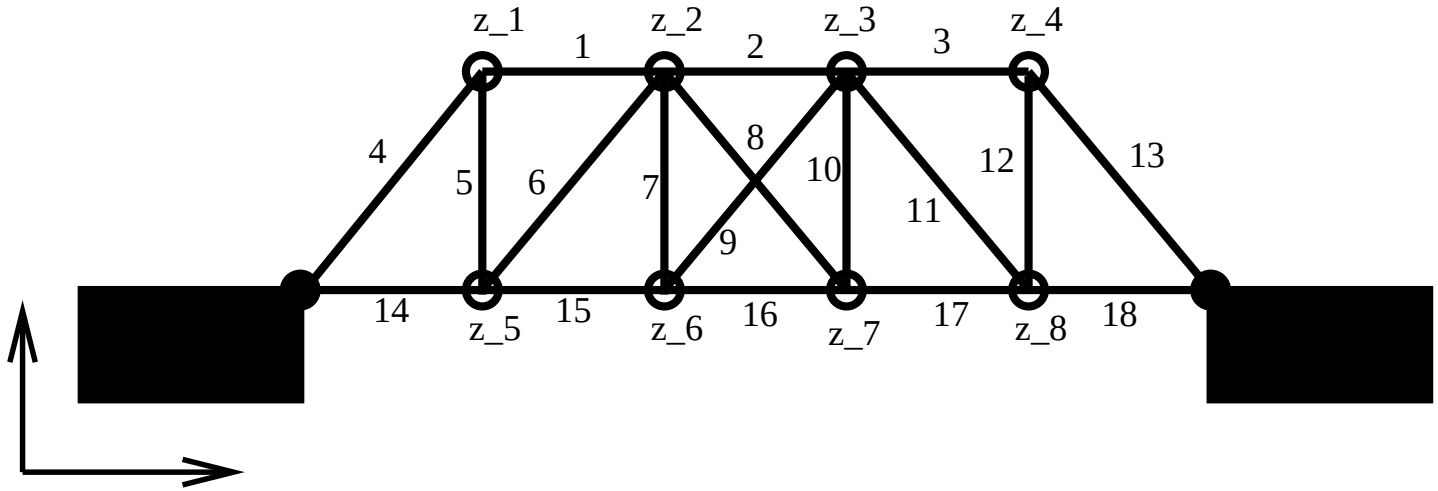
So erhält man zu einem **Eigenvektor** v und zugehörigem **Eigenwert** λ

$$x(t) = \cos(\sqrt{\lambda}t)v$$

als eine Lösung des Systems.
$\Rightarrow$ **Eigenschwingungen** lassen sich direkt aus den **Eigenwerten** ablesen.

Einführung Eigenwerte

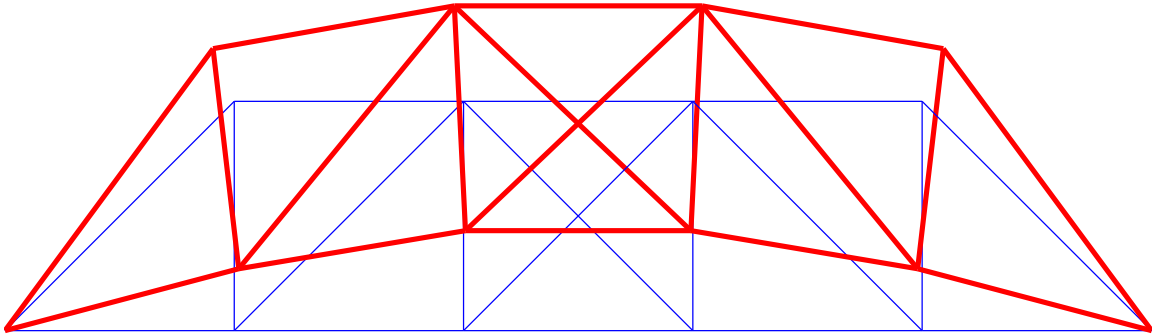
Betrachtet man folgendes Stabwerk



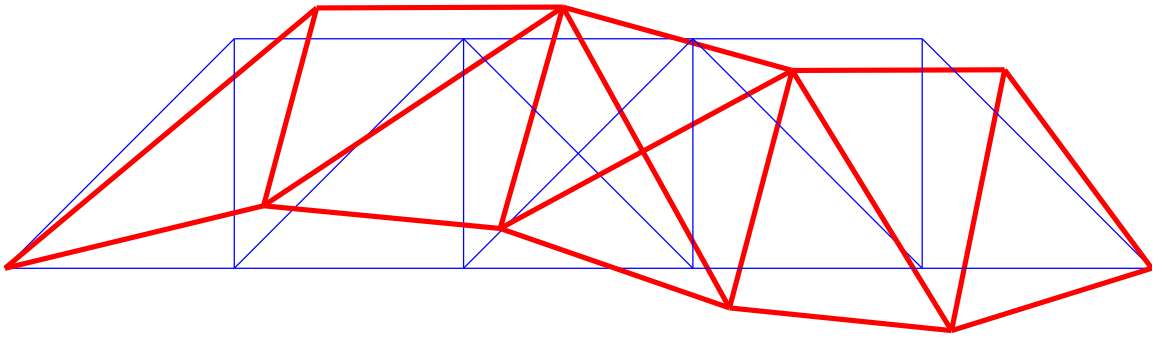
so erhält man bei gegebener Matrix A u.a. folgende Eigenwerte samt Verschiebung.

Einführung Eigenwerte

1. Eigenschwingung: λ=0.35986

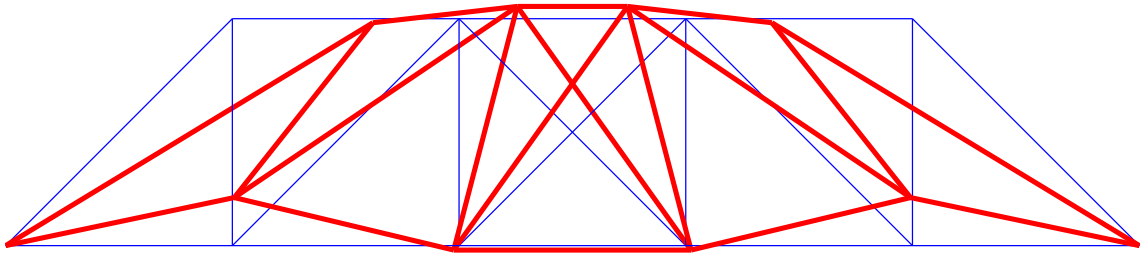


3. Eigenschwingung: λ=2.8027

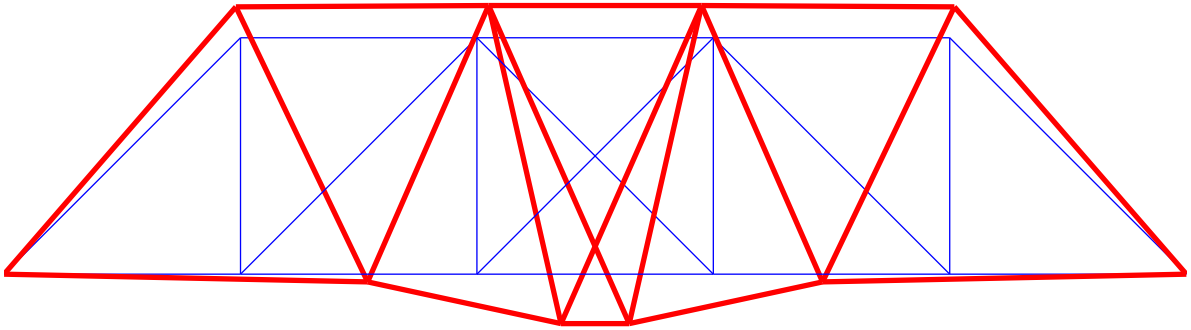


Einführung Eigenwerte

6. Eigenschwingung: λ=9.2956



8. Eigenschwingung: λ=15.4855



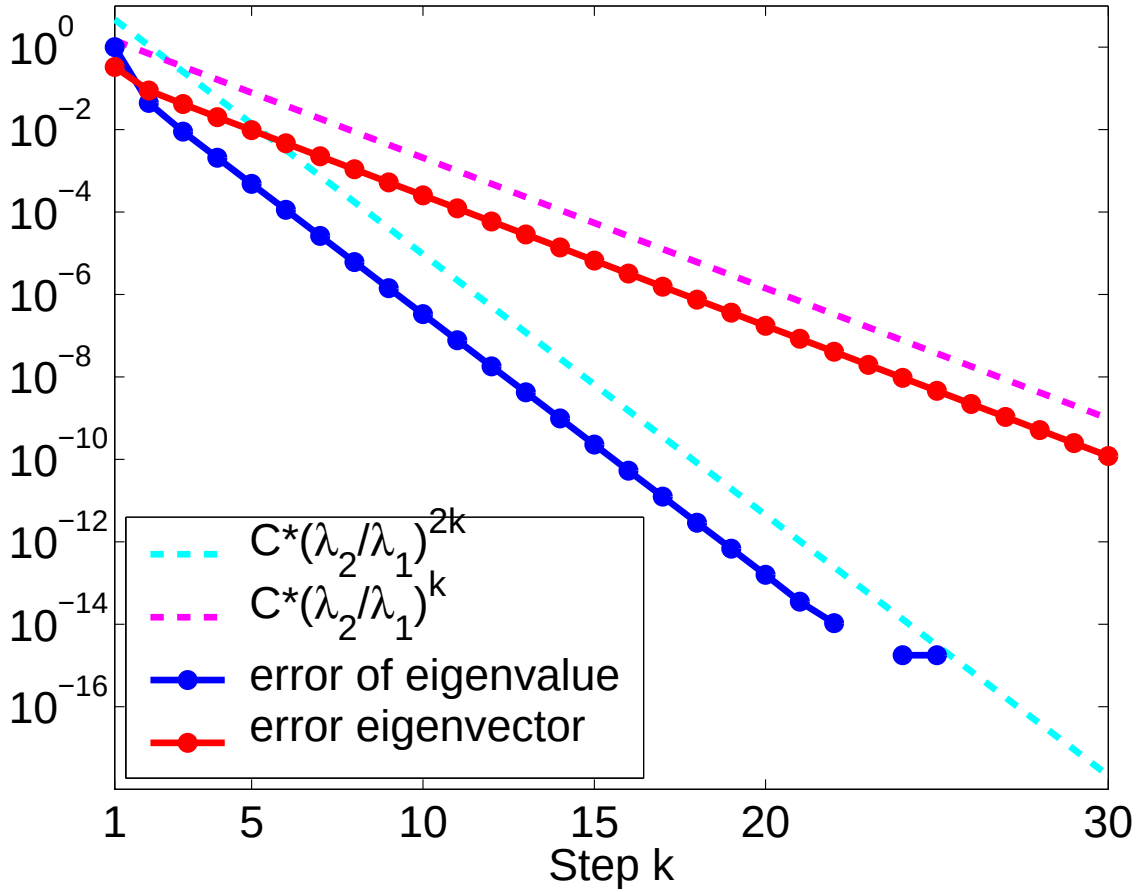
Eigenwerte: Inhalt

Vektoriteration (Bsp. 1): A symmetrisch

$$\begin{pmatrix} 0 & 2 & 2 \\ 2 & 6 & 2 \\ 2 & 2 & 8 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$$

$$\lambda_1 = 10, \qquad \lambda_2 \approx 4.83$$

$$\left|\frac{\lambda_2}{\lambda_1}\right| \approx 0.483$$



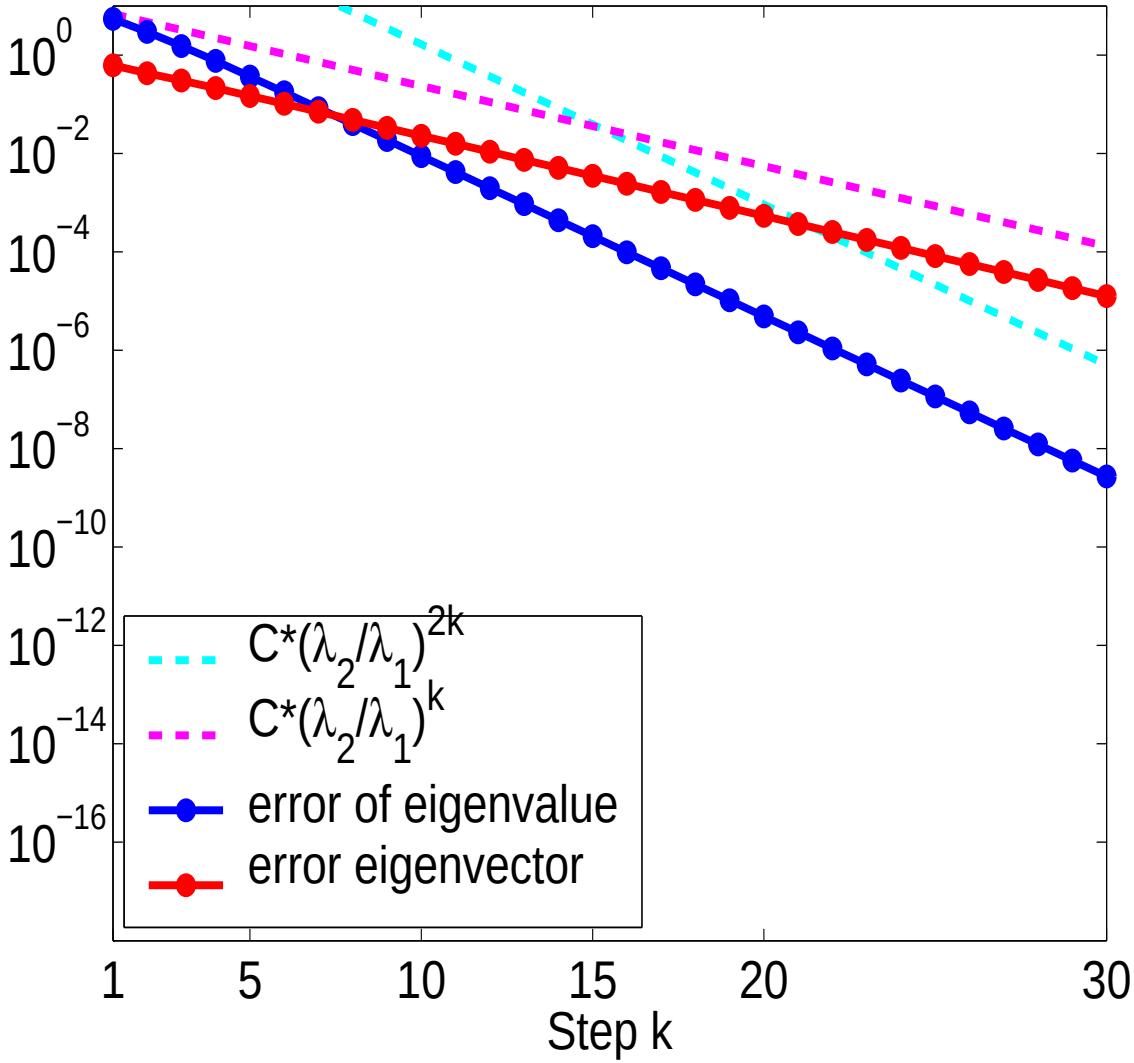
Konvergenzraten symmetrische Matrix EW: $C\left|\frac{\lambda_2}{\lambda_1}\right|^2$ **EV:** $C\left|\frac{\lambda_2}{\lambda_1}\right|$

Vektoriteration (Bsp. 2): A symmetrisch

$$\begin{pmatrix} 1 & 3 & 1 \\ 3 & -2 & 7 \\ 1 & 7 & 5 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 10 \\ 20 \\ 30 \end{pmatrix}$$

$$\lambda_1 = 10, \qquad \lambda_2 \approx -6.87$$

$$\left|\frac{\lambda_2}{\lambda_1}\right| \approx 0.687$$



Konvergenzraten symmetrische Matrix EW: $C\left|\frac{\lambda_2}{\lambda_1}\right|^2$ **EV:** $C\left|\frac{\lambda_2}{\lambda_1}\right|$

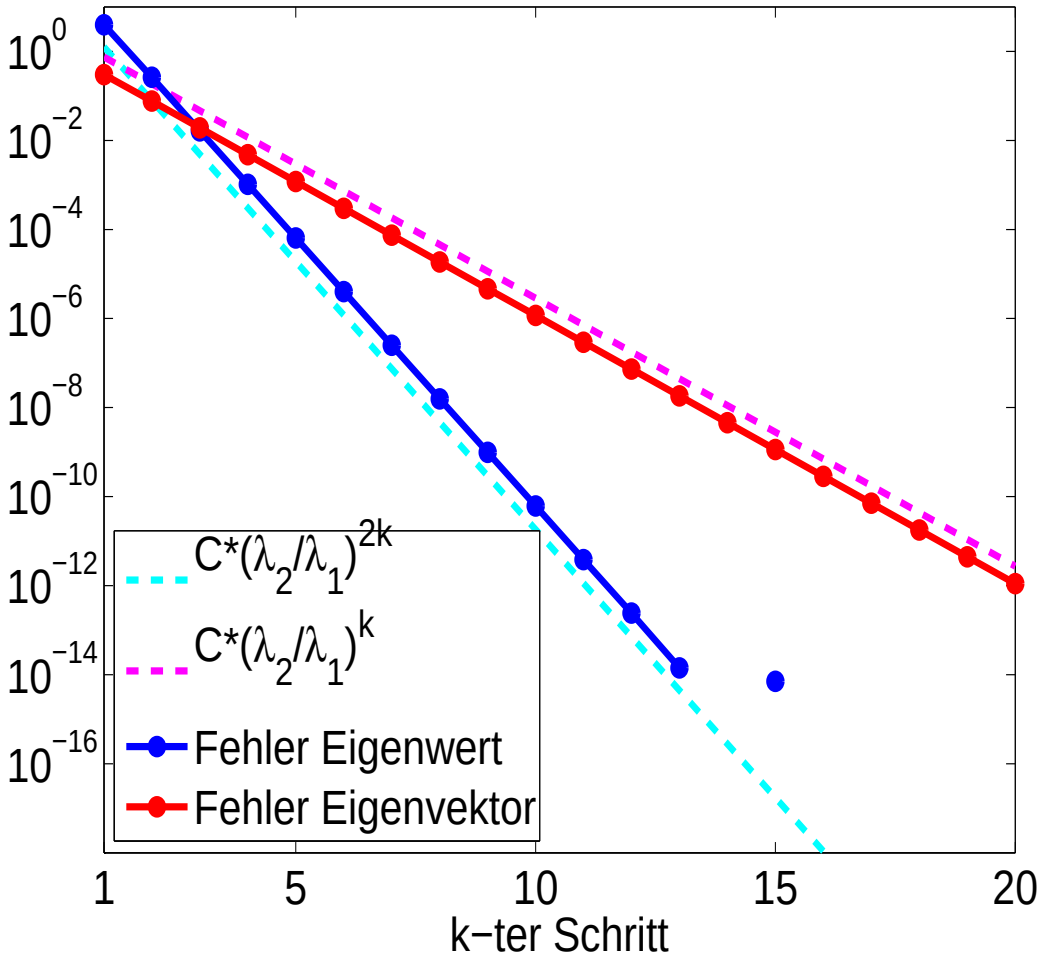
Vektoriteration (Bsp. 3): A symmetrisch

$$A = \begin{pmatrix} -7 & 13 & -16 \\ 13 & -10 & 13 \\ -16 & 13 & 7 \end{pmatrix},$$

$$\lambda_1 = -36, \quad \lambda_2 = 9,$$

$$\left|\frac{\lambda_2}{\lambda_1}\right| = \frac{1}{4},$$

$$x_1 = \frac{1}{\sqrt{3}} \begin{pmatrix} -1 \\ 1 \\ -1 \end{pmatrix}, \quad x^{(0)} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}$$



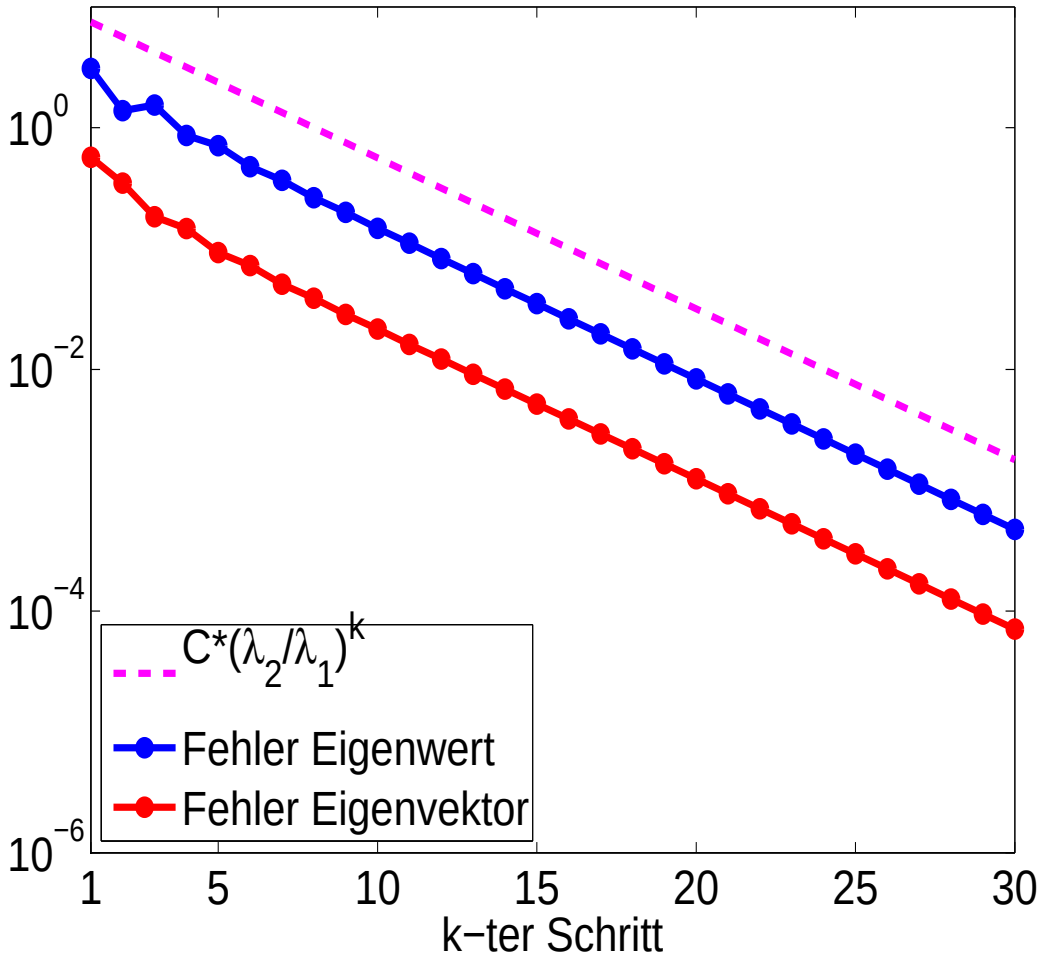
Konvergenzraten symmetrische Matrix EW: $C\left|\frac{\lambda_2}{\lambda_1}\right|^2$ **EV:** $C\left|\frac{\lambda_2}{\lambda_1}\right|$

Vektoriteration (Bsp. 4): A unsymmetrisch

$$A = \begin{pmatrix} 5 & 4 & 4 & 5 & 6 \\ 0 & 8 & 5 & 6 & 7 \\ 0 & 0 & 6 & 7 & 8 \\ 0 & 0 & 0 & -4 & 9 \\ 0 & 0 & 0 & 0 & -2 \end{pmatrix},$$

$$\lambda_1 = 8, \quad \lambda_2 = 6, \quad \left|\frac{\lambda_2}{\lambda_1}\right| = \frac{3}{4},$$

$$x_1 = \frac{1}{5}\begin{pmatrix} 4 \\ 3 \\ 0 \\ 0 \\ 0 \end{pmatrix}, \quad x^{(0)} = \frac{1}{\sqrt{5}}\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$



Konvergenzraten unsymmetrische Matrix EW: $C\left|\frac{\lambda_2}{\lambda_1}\right|$ **EV:** $C\left|\frac{\lambda_2}{\lambda_1}\right|$

Inverse Vektoriteration (Bsp. 5)

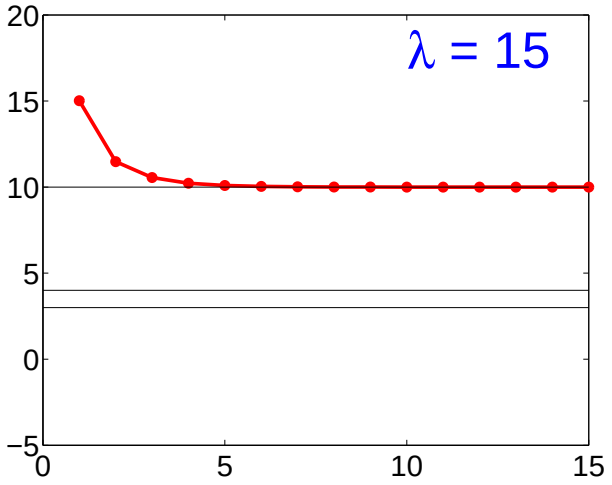
Als Beispiel betrachten wir die Matrix

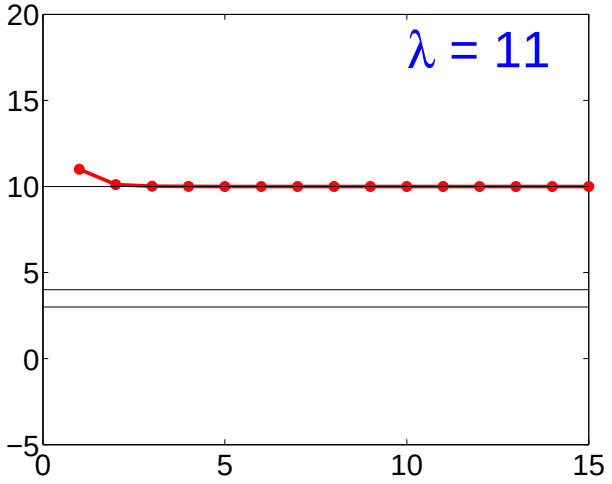
$$A = \begin{pmatrix} -261 & 209 & -49 \\ -530 & 422 & -98 \\ -800 & 631 & -144 \end{pmatrix}$$

mit den Eigenwerten $\lambda_1 = 10,\ \lambda_2 = 4,\ \lambda_3 = 3.$
Ausgehend von verschiedenen Werten für den Shift μ führen wir 15 Schritte der inversen Vektoriteration mit dem Startvektor $x^{(0)} = (1,\ 0,\ 0)^\top$ aus.
Beachte dabei die Anordnung von $|\mu - \lambda_i|$:

Für $\mu = 15$: $|15 - 11| < |15 - 4| < |15 - 3|$

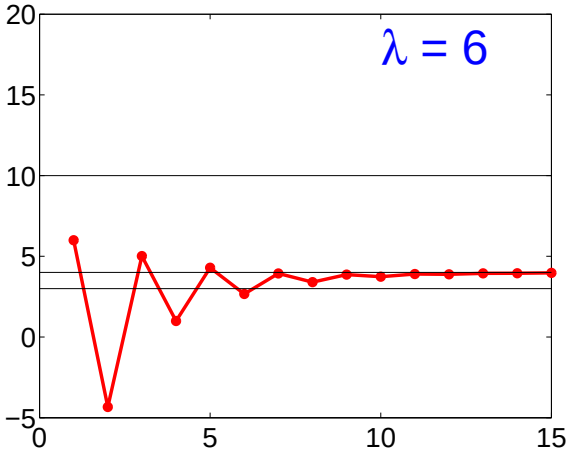
Für $\mu = 11$: $|11 - 11| < |11 - 4| < |11 - 3|$

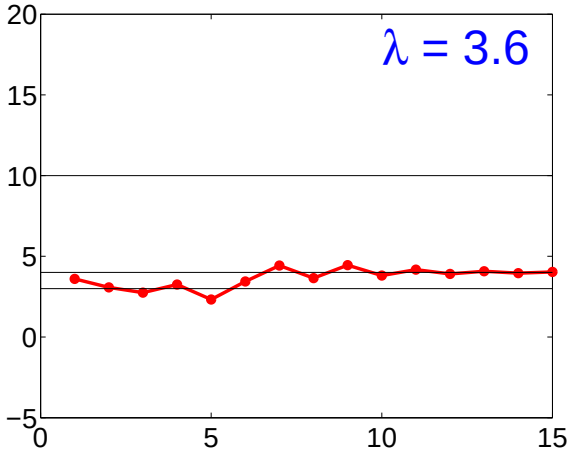




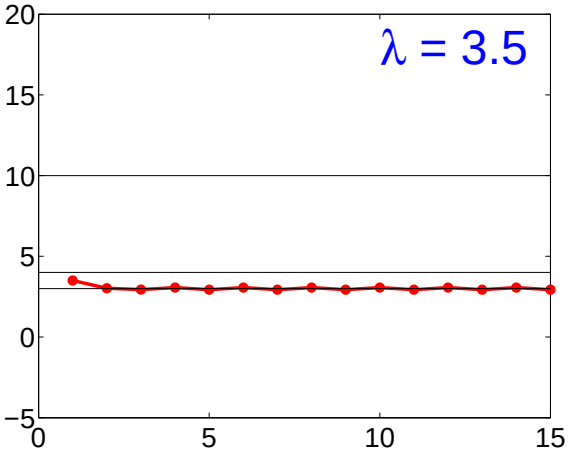
Inverse Vektoriteration (Bsp. 5)

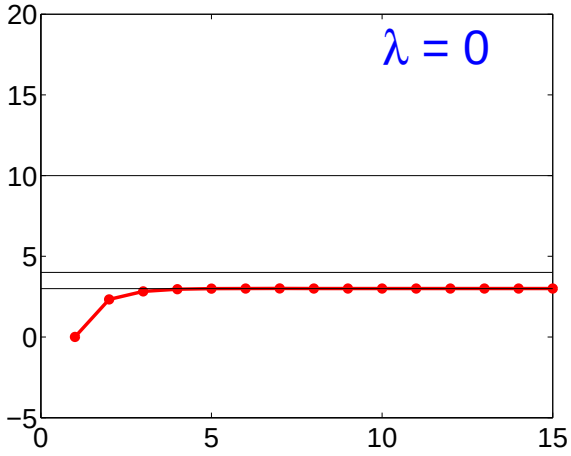
Für $\mu = 6$:
$|6-4| < |6-3| < |6-11|$
Für $\mu = 3.6$:
$|3.6-4| < |3.6-3| < |3.6-11|$





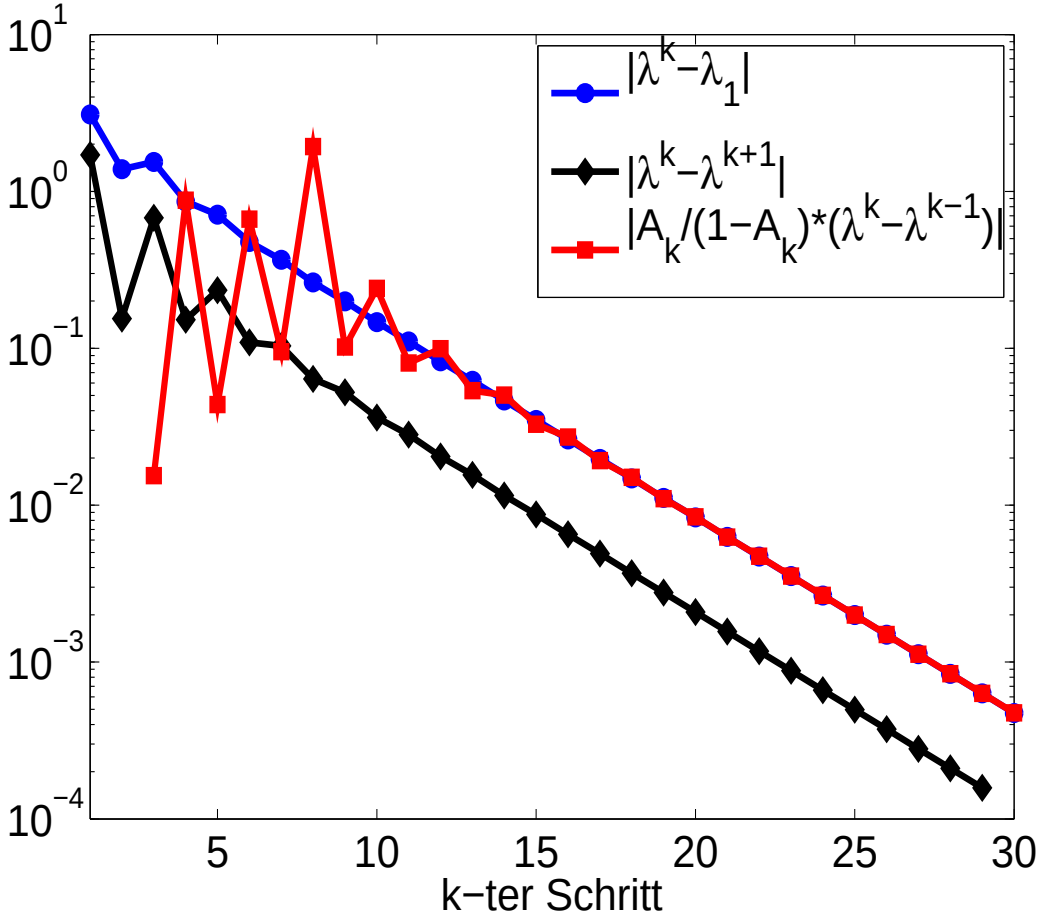
Für $\mu = 3.5$:
$|3.5-3| < |3.5-4| < |3.5-11|$
Für $\mu = 0$:
$|0-3| < |0-4| < |0-11|$





Vektoriteration : Konvergenzaussagen

A unsymmetrisch



Problem: Fehler $\left|\lambda^{(k)} - \lambda_1\right|$ unbekannt.

$\Rightarrow$ $\left|\lambda^{(k+1)} - \lambda^{(k)}\right|$ unterschätzt Fehler $\left|\lambda^{(k)} - \lambda_1\right|$.

$\Rightarrow$ $\left|\frac{A_k}{1-A_k}\left(\lambda^{(k)} - \lambda^{(k-1)}\right)\right|$ ist gutes Maß für $\left|\lambda^{(k)} - \lambda_1\right|$ für k groß genug.

$$A_k := \frac{\lambda^{(k)} - \lambda^{(k-1)}}{\lambda^{(k-1)} - \lambda^{(k-2)}}$$

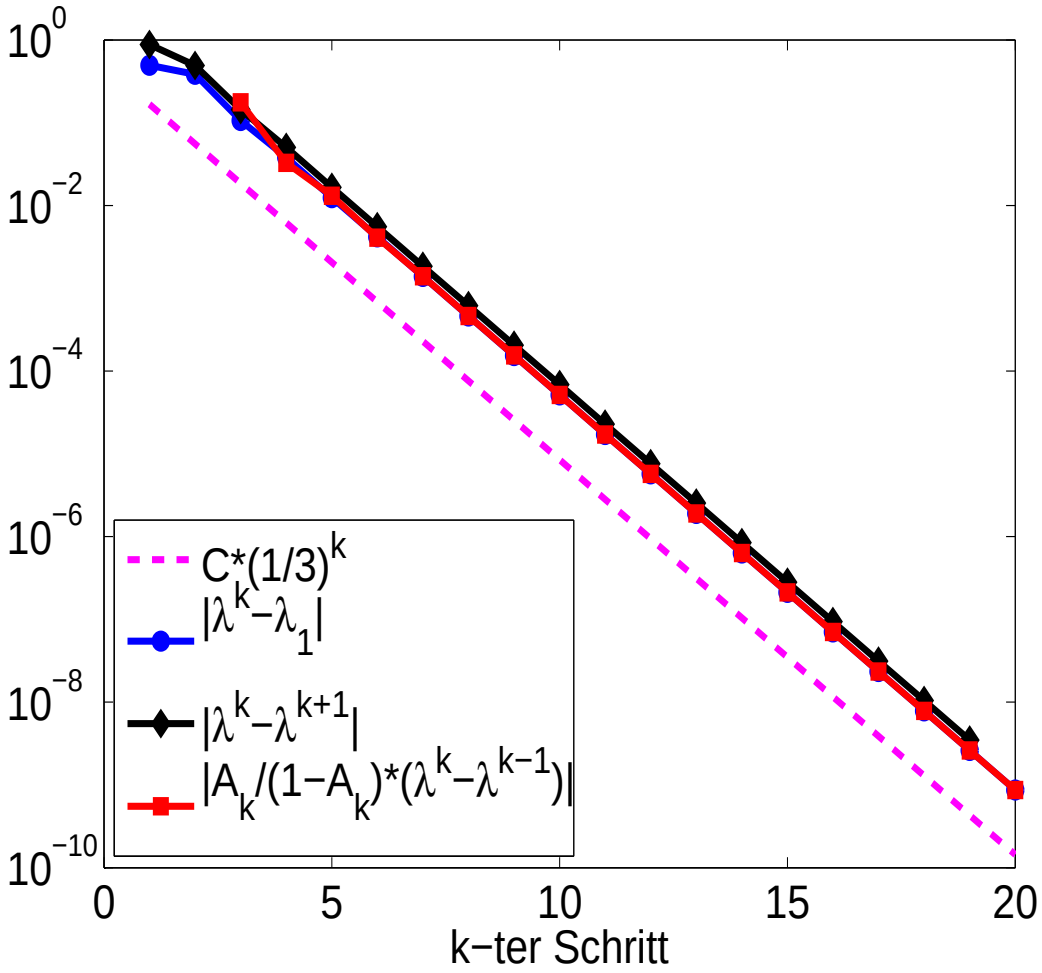
Inverse Vektoriteration: μ fest

$$A = \begin{pmatrix} 5 & 4 & 4 & 5 & 6 \\ 0 & 8 & 5 & 6 & 7 \\ 0 & 0 & 6 & 7 & 8 \\ 0 & 0 & 0 & -4 & 9 \\ 0 & 0 & 0 & 0 & -2 \end{pmatrix},$$

$$\mu = -3.5, \quad \min_j |\lambda_j - \mu| = 0.5,$$

$$\lambda_4 = -4, \quad \min_{j \neq 4} |\lambda_j - \mu| = 1.5,$$

$$\frac{|\lambda_4 - \mu|}{\min_{j \neq 4} |\lambda_j - \mu|} = \frac{1}{3}$$



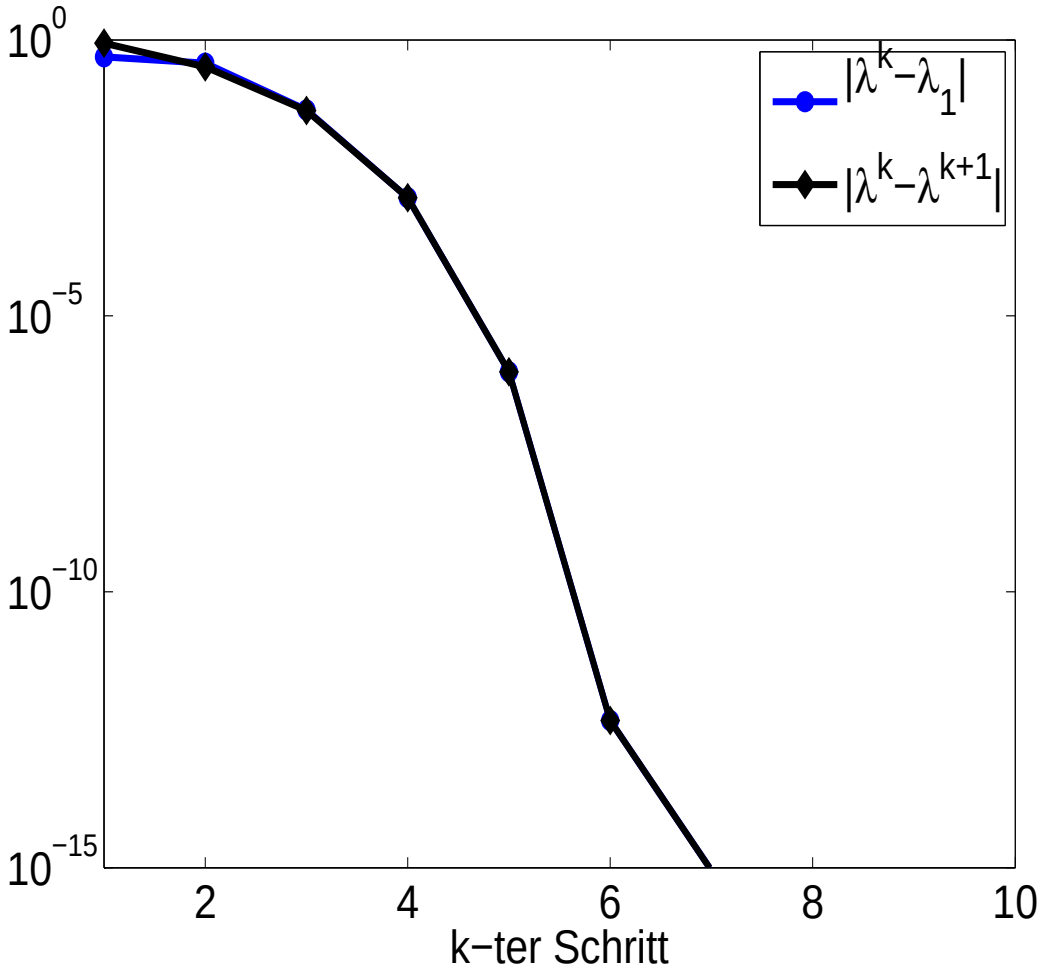
Inverse Vektoriteration: μ adaptiv

$$A = \begin{pmatrix} 5 & 4 & 4 & 5 & 6 \\ 0 & 8 & 5 & 6 & 7 \\ 0 & 0 & 6 & 7 & 8 \\ 0 & 0 & 0 & -4 & 9 \\ 0 & 0 & 0 & 0 & -2 \end{pmatrix},$$

$$\mu^{(0)} = -3.5, \quad \mu^{(k)} = \lambda^{(k-1)}$$

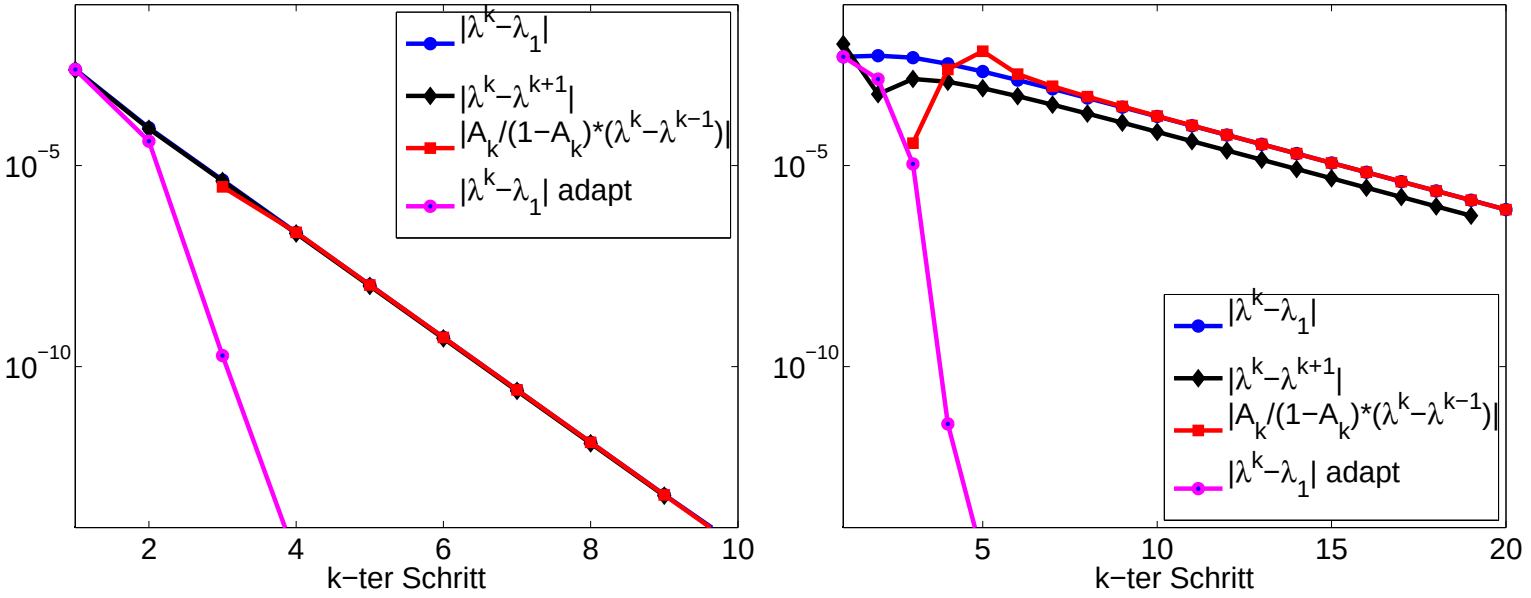
$\Rightarrow$ **quadratische** Konvergenz

$$\left|\lambda^{(k+1)} - \lambda^{(k)}\right| \approx \left|\lambda^{(k)} - \lambda_1\right|$$



Inverse Vektoriteration: Poisson-Matrix

$$A \in \mathbb{R}^{16^2 \times 16^2},\ \lambda_{10,10} = 8\sin^2\left(\frac{10\pi k}{2\cdot 17}\right) \approx 5.0947$$



$$\mu = 5.1,\ \frac{|\lambda_{10,10} - \mu|}{\min|\lambda_{kl} - \mu|} \approx 0.1 \qquad \mu = 5.12,\ \frac{|\lambda_{10,10} - \mu|}{\min|\lambda_{kl} - \mu|} \approx 0.82$$

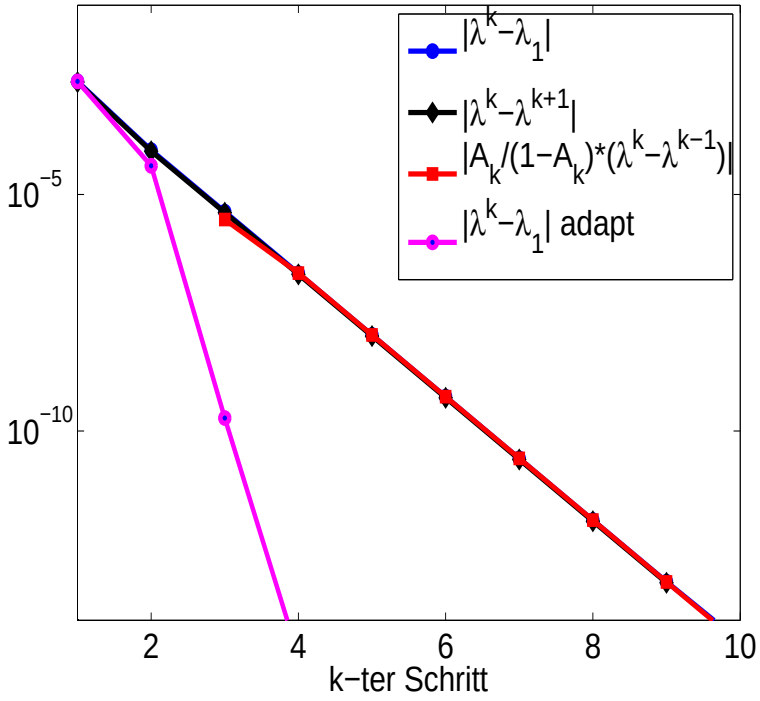
Inverse Vektoriteration (3): Poisson-Matrix

$$A = \begin{pmatrix} B & -\mathrm{Id} & & \\ -\mathrm{Id} & B & \ddots & \\ & \ddots & \ddots & -\mathrm{Id} \\ & & -\mathrm{Id} & B \end{pmatrix} \in \mathbb{R}^{n^2 \times n^2},$$

$$B = \begin{pmatrix} 4 & -1 & & \\ -1 & 4 & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{pmatrix} \in \mathbb{R}^{n \times n},$$

$$\lambda_{kl} = 4\left[\sin^2\left(\frac{\pi k h}{2}\right) + \sin^2\left(\frac{\pi l h}{2}\right)\right],$$

$$1 \le k, l \le n, \quad h = \frac{1}{n+1}$$



QR-Iteration

Vorüberlegung:
Die Multiplikation mit einer orthonormalen Matrix Q heißt Ähnlichkeitstransformation. Die Eigenwerte einer Matrix A bleiben unter einer Ähnlichkeitstransformation erhalten: Sei λ ein Eigenwert der Matrix A, dann folgt

$$Ax = \lambda x \Leftrightarrow Q^T A Q Q^T x = \lambda Q^T x.$$

Außerdem bleibt bei der Multiplikation mit einer orthonormalen Matrix die Kondition erhalten, so dass keine numerischen Stabilitätsprobleme auftreten.
Idee der QR-Iteration:
Durch eine Multiplikation mit einer orthonormalen Matrix kann die Matrix A auf eine Form gebracht werden, für welche die Berechnung der Eigenwerte leichter ist. Dies ist zum Beispiel bei einer oberen Dreiecksmatrix der Fall.

QR-Iteration: Algorithmus

Algorithmus: QR-Iteration
Voraussetzung: Die Matrix A besitzt nur reelle Eigenwerte.

for $k = 0, 1, \ldots$ **do**

$$A_k := Q_k R_k \quad (\Rightarrow QR\text{-Zerlegung})$$

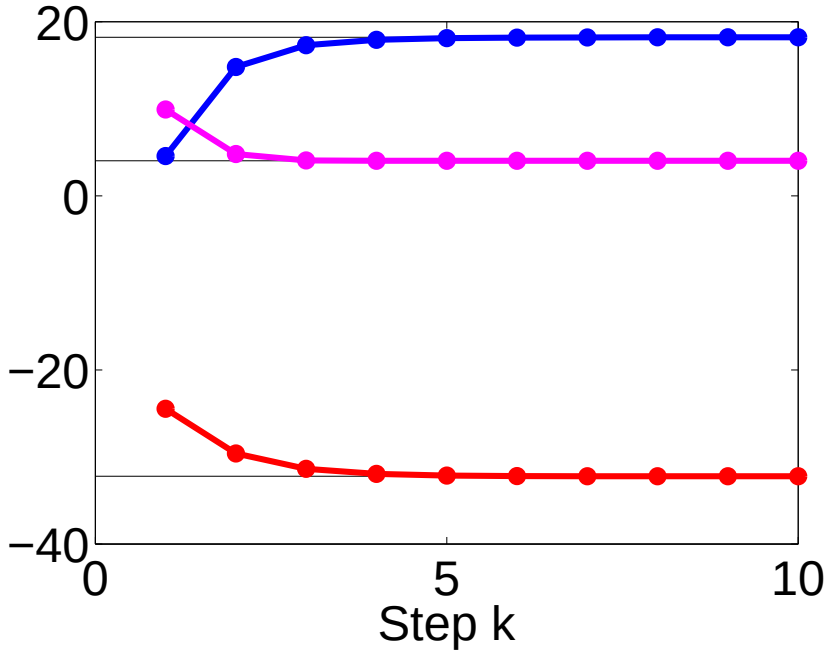
$$A_{k+1} := R_k Q_k \quad \Rightarrow (\lim A_{k+1} = \text{ obere Dreiecksmatrix})$$

end for

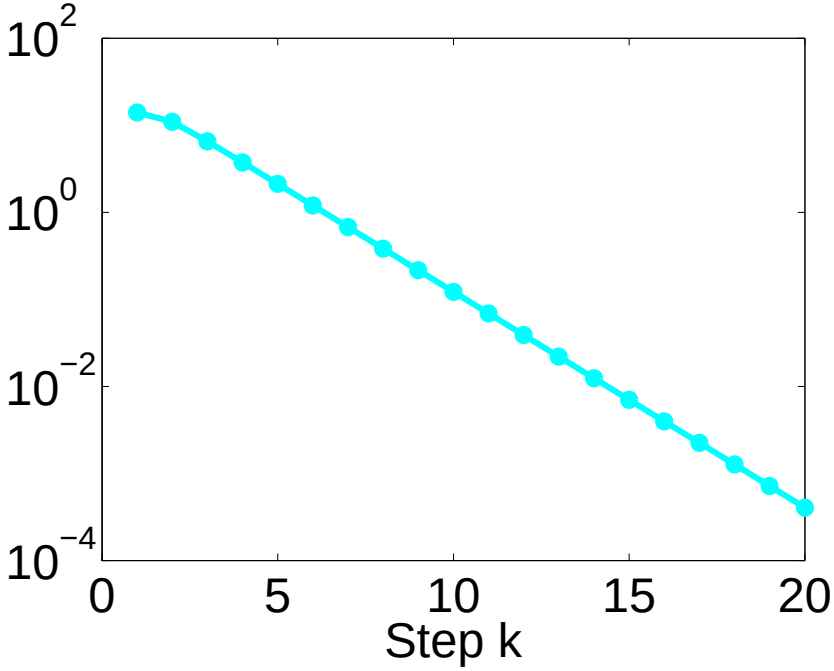
Satz: Die mit dem Algorithmus erzeugten Matrizen A_{k+1} sind ähnlich zu A.

QR-Iteration - A symmetrisch

Diagonaleinträge von $A^{(k)}$



Betrag des größten Eintrages der unteren Dreiecksmatrix von $A^{(k)}$



QR-Iteration - Beispiel: Berechnung aller EW, A symmetrisch

$k = 0, 1, \ldots$

$A^{(k)} = Q^{(k)} R^{(k)}$

$\Rightarrow$ QR-Zerlegung

$A^{(k+1)} = R^{(k)} Q^{(k)}$

$\Rightarrow$ Matrixmultiplikation

$$A^{(0)} = \begin{pmatrix} -7 & 13 & -16 \\ 13 & -10 & 13 \\ -16 & 13 & 7 \end{pmatrix}$$

$$A^{(1)} = \begin{pmatrix} -24.4705 & -14.0115 & -11.2872 \\ -14.0115 & 4.5667 & 3.2256 \\ -11.2872 & 3.2256 & 9.9038 \end{pmatrix}$$

$$A^{(2)} = \begin{pmatrix} -29.6119 & 10.9418 & -2.2539 \\ 10.9418 & 14.8061 & -2.9340 \\ -2.2539 & -2.9340 & 4.8058 \end{pmatrix}$$

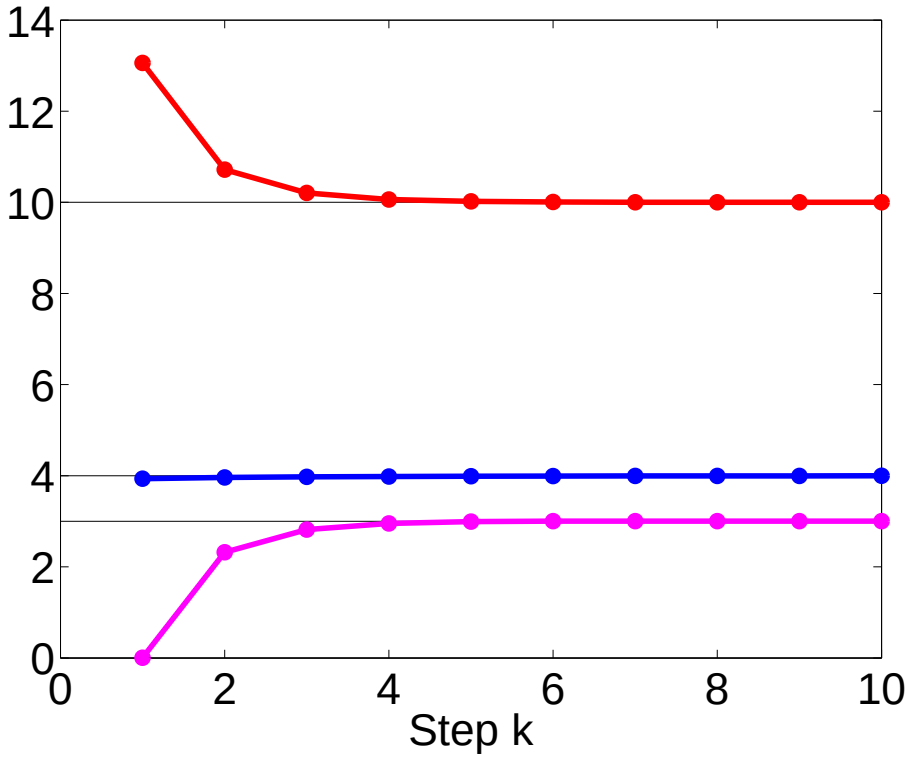
$$A^{(5)} = \begin{pmatrix} -32.1354 & -2.1182 & -0.0046 \\ -2.1182 & 18.1158 & 0.0384 \\ -0.0046 & 0.0384 & 4.0195 \end{pmatrix}$$

$$A^{(10)} = \begin{pmatrix} -32.2242 & 0.1221 & 0.0000 \\ 0.1221 & 18.2048 & 0.0000 \\ 0.0000 & 0.0000 & 4.0194 \end{pmatrix}$$

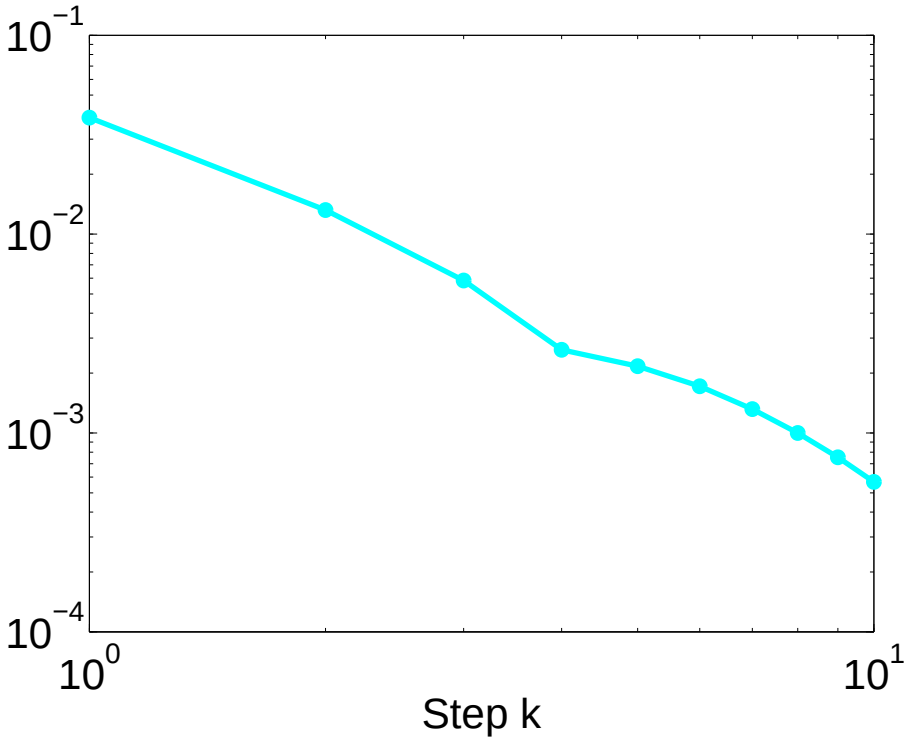
$$A^{(20)} = \begin{pmatrix} -32.2245 & 0.0004 & 0.0000 \\ 0.0004 & 18.2051 & 0.0000 \\ 0.0000 & 0.0000 & 4.0194 \end{pmatrix}$$

QR-Iteration - A unsymmetrisch

Diagonaleinträge von $A^{(k)}$



Betrag des größten Eintrages der unteren Dreiecksmatrix von $A^{(k)}$



QR-Iteration - Beispiel: Berechnung aller EW, A unsymmetrisch

$$k = 0, 1, \ldots$$

$$A^{(k)} = Q^{(k)} R^{(k)}$$

$$\Rightarrow \; QR\text{-Zerlegung}$$

$$A^{(k+1)} = R^{(k)} Q^{(k)}$$

$$\Rightarrow \; \text{Matrixmultiplikation}$$

$$A^{(0)} = \begin{pmatrix} -261 & 209 & -49 \\ -530 & 422 & -98 \\ -800 & 631 & -144 \end{pmatrix}$$

$$A^{(1)} = \begin{pmatrix} 13.061 & -9.108 & -1281.2 \\ 0.039 & 3.935 & -2.934 \\ 0.024 & -0.017 & 0.005 \end{pmatrix}$$

$$A^{(2)} = \begin{pmatrix} 10.719 & -9.005 & 1281.2 \\ 0.013 & 3.962 & -0.845 \\ -0.004 & 0.000 & 2.319 \end{pmatrix}$$

$$A^{(3)} = \begin{pmatrix} 10.207 & -10.073 & -1281.2 \\ 0.006 & 3.975 & 2.423 \\ 0.001 & 0.002 & 2.818 \end{pmatrix}$$

$$A^{(5)} = \begin{pmatrix} 10.012 & -12.106 & -1281.2 \\ 0.001 & 3.989 & 3.489 \\ 0.000 & 0.002 & 2.991 \end{pmatrix}$$

$$A^{(10)} = \begin{pmatrix} 10.000 & -14.393 & 1281.1 \\ 0.000 & 3.998 & -3.739 \\ 0.000 & 0.000 & 3.002 \end{pmatrix}$$

QR-Iteration: Hessenberg-Matrix

Aufwand der QR-Iteration:
Bei vollbesetzten Matrizen beträgt der Aufwand der QR-Iteration pro Schritt $\mathbb{O}(n^3)$.

Idee:
Überführe A durch Ähnlichkeitstransformationen in eine obere Hessenberg-Matrix H mit

$$H = \begin{pmatrix} * & \cdots & \cdots & * \\ * & \ddots & & \vdots \\ & \ddots & \ddots & \vdots \\ 0 & & * & * \end{pmatrix},$$

also ist $H_{ij} = 0$ für $i > j + 1$.
Dann beträgt der Aufwand der QR-Iteration pro Schritt nur noch $\mathbb{O}(n^2)$. Für symmetrische Matrizen beträgt der Aufwand pro Schritt sogar nur noch $\mathbb{O}(n)$.

QR-Iteration: Konvergenz

Problem:
Langsame Konvergenz der QR-Iteration im Fall betragsmäßig nahe beieinander liegender Eigenwerte.

Ausweg:
Einfache Shift-Strategie zusammen mit Deflation.

Bemerkung:
Wenn A komplexe Eigenwerte hat, so ist eine doppelte Shift-Strategie notwendig.

QR-Iteration: Algorithmus mit Shift und Deflation

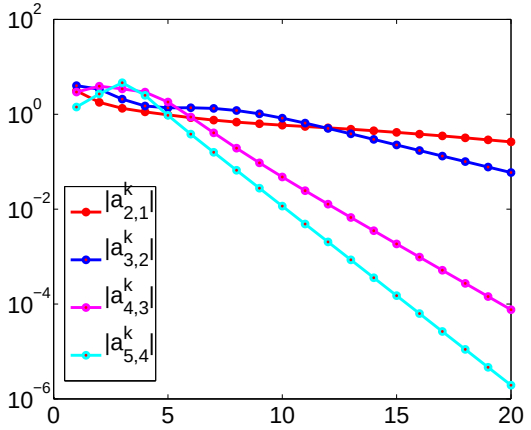
function $QRrayleigh(A)$
$A \in \mathbb{R}^{n\times n}$, $A_0 = A$, A in Hessenbergform
if $n = 1$ **then**
  return $A(1,1)$
end if
Define $l = 0$, TOL, $l_{\max}$
while $|A_l(n,n-1)| > TOL \cdot (|A_l(n,n)| + |A_l(n-1,n-1)|)$ **and** $l \leq l_{\max}$ **do**
  $\mu^{(l)} := A_l(n,n)$
  $A_l - \mu^{(l)} Id =: Q_{l+1} R_{l+1}$
  $A_{l+1} := R_{l+1} Q_{l+1} + \mu^{(l)} Id$
  $l := l + 1$
end while
if $n = 2$ **then**
  return $A_l(2,2) \cup A_l(1,1)$
else
  $A_{deflation} = A_l(1 : n-1, 1 : n-1)$
  return $A_l(n,n) \cup QRrayleigh(A_{deflation})$
end if

QR-Iteration ohne Shift, A symmetrisch

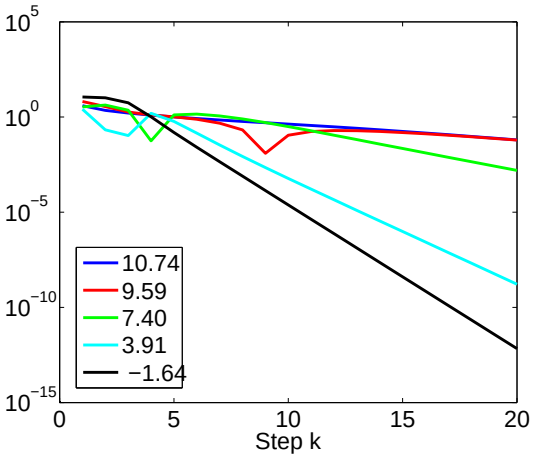
$$A = \begin{pmatrix} 2 & 4 & 0 & 0 & 0 \\ 4 & 4 & 3 & 0 & 0 \\ 0 & 3 & 6 & 2 & 0 \\ 0 & 0 & 2 & 8 & 1 \\ 0 & 0 & 0 & 1 & 10 \end{pmatrix}$$

Durch Anwendung des QR-Algorithmus erhält man **lineare** Konvergenz in der Subdiagonalen gegen die Null.

Subdiagonaleinträge



Konvergenz im EW

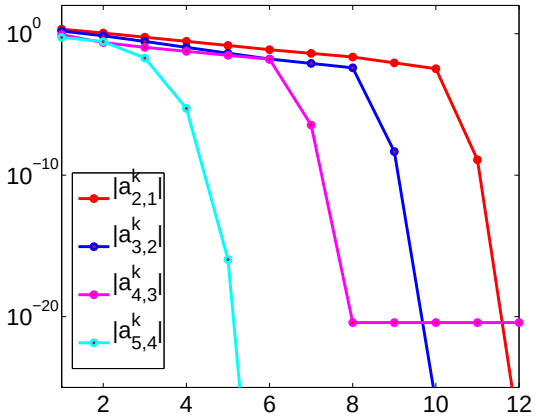


QR-Iteration mit Shift und Deflation, A symmetrisch

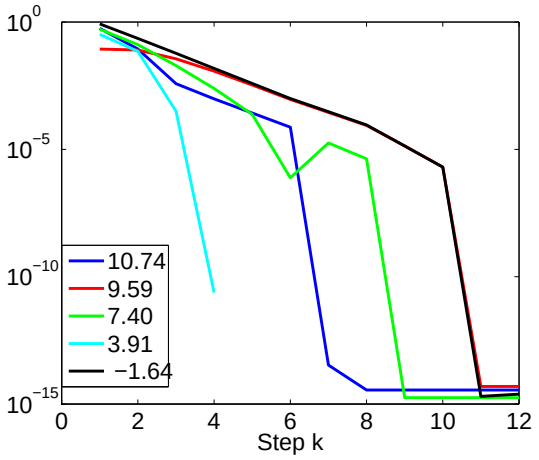
$$A = \begin{pmatrix} 2 & 4 & 0 & 0 & 0 \\ 4 & 4 & 3 & 0 & 0 \\ 0 & 3 & 6 & 2 & 0 \\ 0 & 0 & 2 & 8 & 1 \\ 0 & 0 & 0 & 1 & 10 \end{pmatrix}$$

Durch Anwendung des QR-Algorithmus mit Shift und Deflation erhält man **kubische** Konvergenz in der Subdiagonalen gegen die Null.

Subdiagonaleinträge



Konvergenz der EW



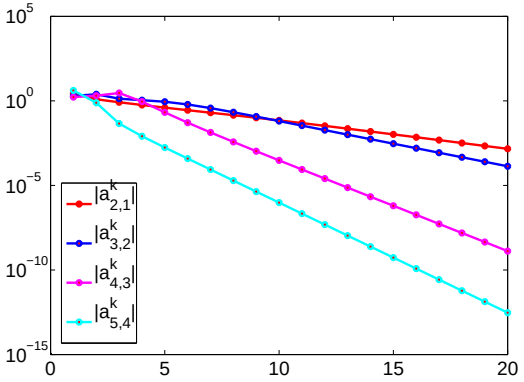
QR-Iteration ohne Shift, A unsymmetrisch

$$A = \begin{pmatrix} 2 & 3 & 4 & 5 & 6 \\ 4 & 4 & 5 & 6 & 7 \\ 0 & 3 & 6 & 7 & 8 \\ 0 & 0 & 2 & 8 & 9 \\ 0 & 0 & 0 & 1 & 10 \end{pmatrix}$$

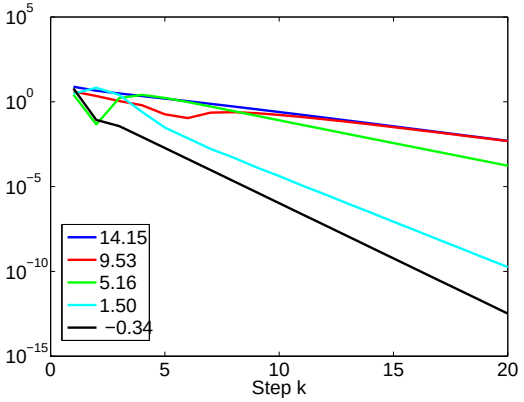
Durch Anwendung des QR-Algorithmus erhält man **lineare** Konvergenz in der Subdiagonalen gegen die Null.

Dahmen/Reusken: Numerik für Ingenieure und Naturwissenschaftler, 2.korrigierte Auflage, Springer, Berlin 2008, Seite 257ff.

Subdiagonaleinträge



Konvergenz der EW



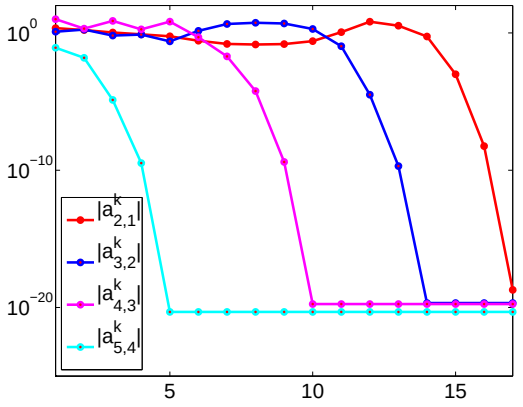
QR-Iteration mit Shift und Deflation, A unsymmetrisch

$$A = \begin{pmatrix} 2 & 3 & 4 & 5 & 6 \\ 4 & 4 & 5 & 6 & 7 \\ 0 & 3 & 6 & 7 & 8 \\ 0 & 0 & 2 & 8 & 9 \\ 0 & 0 & 0 & 1 & 10 \end{pmatrix}$$

Durch Anwendung des QR-Algorithmus mit Shift und Deflation erhält man **quadratische** Konvergenz in der Subdiagonalen gegen die Null.

Dahmen/Reusken: Numerik für Ingenieure und Naturwissenschaftler, 2.korrigierte Auflage, Springer, Berlin 2008, Seite 257ff.

Subdiagonaleinträge



Konvergenz der EW