Numerische Lineare Algebra

Eigenwertproblem: Shifts und Eigenvektorberechnung

Ausarbeitung von Kristina Erbert

Praktische Durchführung des QR-Algorithmus

In meinem Vortrag werden die zwei wichtigsten Aspekte diskutiert, die bei einer effizienten Implementierung des QR-Verfahrens zu berücksichtigen sind:

-eine Trasformation auf Hessenbergform

-die Technik der Spektralverschiebung

Transformation auf Hessenbergform

Eine Matrix ($B \in R^{nxn}$ (C^{nxn})) heißt obere Hessenberg-Matrix, falls B die Gestalt hat

$B=\times\times\times\times\times\times\times\times 0\times 00\times\times\times\times$

Ist B eine symmetrische Matrix und hat sie eine Hessenberg-Gestalt wie oben , dann muss B eine Tridiagonalmatrix sein. Es sei noch bemerkt, dass man die Transformation auf obere Hessenberg-Form anstatt über Housholder- Transformationen, auch mit Givens-Rotationen durchführen kann.

Beispiel 1 (Im Beispiel wird gezeigt, wie man eine Matrix A über eine orthogonale Ähnlichkeitstransformation, d.h. Q^TAQ mit Q orthogonal, auf obere Hessenbergform bringen kann)

Sei die Matrix A=115-601731222-7-28-30153gegeben. Um A auf obere Hessenbergform zu bringen, wird zum Verschwinden der Einträge a_{31}und a_{41} eine Householder-Transformation verwendet. Man setze dazu $v_1 := 122+3100=422$, $Q_{v1} := I-2v_1v_1^Tv_1^Tv_1 \in R^{3\times 3}$

Und $Q_1 := 1000000Q_{v1}$ (1.1)

Dann ergibt sich Q_1A=115-60-321-9-6000-21-9-99-6.

Da Q_1 die Form wie in (1.1) hat, bleiben bei der Multiplikation von Q_1A mit Q_1 *die Null-Einträge in der ersten Spalte erhalten:*

$A := Q_1AQ_1$ =1-1-14-8-33-18-1500125-3-3227

Da für die Housholder-Transformation $Q_1 = Q_1^T = Q_1^{-1}$ gilt, sind die Matrizen A und A ähnlich, also $\sigma A = \sigma(A)$.

Im nächsten Schritt wird der Eintrag a_{42} mit Hilfe einer geeigneten Householder-Transformation eliminiert. Sei dazu $v_2 := 125+1310=255$, $Q_{v2} := I-2v_2v_2^Tv_2^Tv_2 \in R^{2\times 2}$,

und $Q_2 := 100001000000Q_{v2}$ (1.2)

Dann ergibt sich

Q_2A=1-1-14-8-33-18-1500-130-5.6920.076921.4627.615,

und wiederrum bleiben wegen der Form von Q_2 in (1.2) die Null-Einträge in den ersten zwei Spalten bei der Multiplikation vor rechts erhalten:

$A := Q_2AQ_2$=1-116-2-3322.385-6.92300-1305.2252.260-22.7401.225
(1.3)

Sei Qdie orthogonale Matrix $Q=Q_1Q_2$ also $Q^T=Q_2^TQ_1^T=Q_2Q_1$, dann gilt

Q-1AQ=QTAQ=Q2Q1AQ1Q2=A,

mit A wie in (1.3). Die Matrix A ist also über eine *orthogonale* Transformation *ähnlich zur oberen Hessenberg-Matrix A.* Also gilt:

Man kann eine Matrix A ∈ Rn×n durch Householder-Transformation auf eine zu A ähnliche Matrix mit oberer Hessenberggestalt bringen.

Zur Berechnung der Eigenwerte der oberen Hessenberg-Matrix A benutzen wir den QR-Algorithmus. Man kann zeigen, dass die Hessenberggestalt der Matrix mehrere Vorteile bringt:

-wenn A eine nicht-reduzierbare obere Hessenbergmatrix ist, kann man die Identität als Anfangsmatrix beim QR-Algorithmus nehmen.

-dadurch, dass man beim QR-Algorithmus in einer Vorbereitungsphase die Matrix auf obere Hessenbergform bringt, braucht man nur die QR-Zerlegung einer Hessenberg-Matrix Ak-1 zu berechnen. Falls man dazu Givens-Rotationen verwendet, ist der Aufwand für die Berechnung Ak-1=:QR, Ak:=RQ nur O(n2) Operationen. Falls A symmetrisch ist, ist dieser Aufwand nur O(n) Operationen.

Der QR-Algorithmus

Basis für diesen Zusammenhang, der letztendlich im QR-Algorithmus mündet, ist der folgende Satz von Schur (1908), der hier in der Formulierung für reelle Matrizen wiedergegeben wird und schon in der Einführung zur Gesamtproblematik zitiert wurde.

Satz (Satz von Schur**).** Zu jeder reellen Matrix A der Ordnung N existiert eine orthogonale Matrix Q der Ordnung N, so dass die zu A ähnliche Matrix R=QTAQ die Quasidreiecksgestalt
R= QTAQ=R11……R1m0R22R2m ⋮ 0⋱⋯⋱0 ⋮ Rmm
(1.4)
Besitzt.
(ohne Beweis).

Auf diesem Satz beruht nun *die Idee des QR-Algorithmus*: Man konstruiere obige Quasidreiecksmatrix R (deren Existenz aufgrund des Satzes gesichert ist) auf möglichst effektive Weise. Die Eigenwerte sind dann entweder direkt ablesbar oder aus den 2×2 Untermatrizen einfach zu berechnen. Diese Konstruktion versucht man, über eine Folge von QR-Transformationen (bezüglich Hessenberg-Matrizen) H,

Hk=QkRk, Hk+1=RkQk, k=1,2,…..

zu erreichen. Dann gilt folgende Konvergenzaussage

Satz (Francis, 1961).

(i) Die Matrix H habe Eigenwerte λi mit der Eigenschaft, dass

$$\lambda 1 > \lambda 2 > \ldots > \lambda N,$$

d.h. die Eigenwerte sind reell und nicht entartet. Die dazugehörigen Eigenvektoren xi seien als Spalten in der regulären Matrix $X \in RN \times N$(der Reihe nach) zusammengefasst.
Falls nun für X-1 die LU-Zerlegung existiert, dann konvergiert die Folge der QR-Transformationen Hk für $k \to \infty$ gegen eine Rechtsdreiecksmatrix und es gilt

$$\lim k \to \infty hiik = \lambda i, \qquad i=1,\ldots,N , \qquad (1.5)$$

d.h. in der erzeugten Matrix sind die Eigenwerte betragsmäßig geordnet, mit dem absolut größten Eigenwert „links oben“

(ii) Hat die Matrix H Paare von konjugiert komplexen Eigenwerten derart, dass ihre Beträge und die Beträge der reellen Eigenwerte paarweise verschieden sind, und existiert die komplexe LU-Zerlegung von X-1, dann konvergiert die Folge Hk, $k \to \infty$ gegen die Quasidreiecksmatrix A

Problem. Obwohl aufgrund dieses Satzes die Konvergenz (und damit die Lösung des Eigenwertproblems) gesichert ist (zumindest unter der gegebenen Einschränkung, die allerdings fast immer erfüllt ist), kann diese Konvergenz sehr langsam sein. Es lässt sich nämlich zeigen, dass für große k die Subdiagonalelemente wie

$$hi+1,ik \approx \lambda i+1 \, \lambda ik, \qquad i=1,\ldots..,N-1 \qquad (1.6)$$

konvergieren. Damit gilt, dass

(a) im reellen Fall, $\lambda i+1/\lambda i<1$ die Konvergenz nur *linear* ist und vom Verhältnis ($\lambda i+1/\lambda i$) abhängt, wobei dieses Verhältnis nahe bei „1" liegen kann und somit die Konvergenz „schleichend" ist.

(b) Für konjugiert komplexe Paare tritt keine Konvergenz ein, da $\lambda i+1/\lambda i=1$ (wie es auch der Satz behauptet; man beachte das Auftreten von entsprechenden 2×2 Untermatrizen).

Aufgrund dieser Problematik ist der „reine" QR-Algorithmus nur schlecht geeignet, das Eigenwertproblem zu lösen, da die erforderliche Rechenzeit teilweise unannehmbar wäre. Es gibt allerdings eine Möglichkeit, die Konvergenz drastisch zu beschleunigen, die auf der sog. „Spektralverschiebung" beruht und auf zwei Arten in den Algorithmus „eingebaut" werden kann. Im weiteren werden wir uns auf die sog. „explizite" Spektralverschiebung konzentrieren und die entsprechenden Algorithmen (sowohl für rein reelle als auch für reelle und komplexe Eigenwerte) vorstellen. Bezüglich der zweiten Methode, der sog. „impliziten" Spektralverschiebung, sei auf die Literatur verwiesen z.B. Schwarz, „Numerische Mathematik, Kap. 6.4.3 und „Numerical Recipes", Kap. 11.3/4.

Der QR-Algorithmus mit expliziter Spektralverschiebung für reelle Eigenwerte

Die oben erwähnte *Spektralverschiebung* beruht darauf, die Eigenwerte einer Matrix durch Subtraktion einer Diagonalmatrix mit identischen Elementen $\sigma \in R$ zu modifizieren. Die modifizierte Matrix

$$H=H-\sigma 1$$

hat dann die Eigenwerte $\lambda i-\sigma$, $i=1,\ldots,N$. Diese seien jetzt so indiziert, dass

$$\lambda 1-\sigma > \lambda 2-\sigma > \ldots > \lambda N-\sigma$$

Damit konvergieren die Subdiagonalelemente der Folge

$$Hk \text{mit } H1=H-\sigma 1$$

mit

$$h_{i+1,i} \approx \lambda_{i+1} - \sigma \;\; \lambda_i - \sigma_k, \qquad i=1,\ldots,N-1 \qquad (1.7)$$

Sei nun σ eine gute Näherung für λ_N, so dass

$$\lambda_N - \sigma \ll \lambda_i - \sigma, \qquad i=1,\ldots,N-1$$

gilt. Dann konvergiert das „letzte" Subdiagonalelement $h_{N,N-1}^k$ sehr schnell gegen Null (man beachte, dass der Quotient in Gl. (1.7) mit dieser Wahl sehr klein ist), und die Matrix zerfällt nach wenigen Iterationsschritten in die folgende Struktur (nachdem wir die Spektralverschiebung rückgängig gemacht haben, s.u.)

$$\begin{pmatrix} \times & \times & \times & \times & | & \times \\ \times & \times & \times & \times & | & \times \\ 0 & \times & \times & \times & | & \times \\ 0 & 0 & \times & \times & | & \times \\ \cdots & \cdots & \cdots & \cdots & | & \cdots \\ 0 & 0 & 0 & \underbrace{\tilde{h}_{N,N-1}}_{\to 0} & | & \to \lambda_N \end{pmatrix} \Rightarrow \begin{pmatrix} \hat{H} & | & \hat{\boldsymbol{h}} \\ \cdots & | & \cdots \\ \mathbf{0}^T & | & \lambda_N \end{pmatrix}$$

Nachdem die Matrix zerfallen ist und wir somit den ersten Eigenwert λ_N bestimmt haben, sind die restlichen Eigenwerte diejenigen der Matrix $H \in R^{N-1 \times (N-1)}$, die wir genauso wie eben beschrieben behandeln.

Die *Spektralverschiebung* wird nicht nur einmal, sondern vor jedem QR- Schritt angewandt, wobei σ den Veränderungen angepasst wird. Am Ende des Schrittes wird die jeweilige Verschiebung wieder rückgängig gemacht. Damit lautet der QR-Algorithmus mit expliziter Spektralverschiebung

$$\begin{aligned} H_k - \sigma_k 1 &= Q_k' R_k' \quad (\textit{Zerlegung}) \\ \Rightarrow R_k' &= Q_k'^T (H_k - \sigma_k 1) \\ \Rightarrow \\ H_{k+1} &= R_k' Q_k' + \sigma_k 1 \quad (\textit{Transformation, Spektralverschiebung rückgängig gemacht}) \\ &= Q_k'^T H_k - \sigma_k 1 Q_k' + \sigma_k 1 \\ &= Q_k'^T H_k Q_k' \end{aligned} \qquad (1.8)$$

Unter Verwendung der Spektralverschiebung haben wir also wiederum eine Matrix H_{k+1} erzeugt, die orthogonal ähnlich zu H_k ist (wie im originalen Algorithmus, allerdings jetzt mit einer veränderten Transformationsmatrix Q') , aber die Konvergenz beschleunigt. Man beachte allerdings, dass die Eigenwerte jetzt nicht mehr bezüglich ihrer (absoluten) Größe λ_i angeordnet sind.
Nachzutragen bleibt nun noch, welchen Wert man für das jeweilige σ_k annimmt. Optimal wäre natürlich $\sigma = \lambda_N$, aber λ_N wird ja gerade gesucht. In Praxis werden **zwei Möglichkeiten** zur Wahl von σ betrachtet.

(a) Man verwendet einfach das aktuelle „letzte" Diagonalelement

$$\sigma_k = h_{NN}^k, \qquad k=1,2,\ldots\ldots \;, \qquad (1.9)$$

(b) Eine bessere Konvergenz (und eine Verallgemeinerung auf den komplexen Fall, siehe nächstes Kapitel) ergibt sich, wenn man σ über die Eigenwerte der aktuellen, letzten 2×2 -Untermatrix nähert. Diese berechnen sich über

$$C_k = \begin{vmatrix} h_{N-1,N-1} & h_{N-1,N} \\ h_{N,N-1} & h_{NN} \end{vmatrix} = \begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

$$a - \lambda \; d - \lambda - bc = 0$$

$$ad-\lambda a+d+\lambda 2-bc=0$$
$$\Rightarrow \lambda 2-\lambda a+d+\lambda 2-bc=0$$

$$\lambda 1,2=a+d2-+\ a+d22-(ad-bc)$$

(2.0)

Falls die Diskriminante < 0 ist, liegen (höchstwahrscheinlich) konjugiert komplexe Lösungen vor, dann wählt man für σk denjenigen Eigenwert, der näher an hNN liegt!

$$\sigma k=\lambda i,\ \text{so dass}\ \lambda i-hNN=Min! \qquad (2.1)$$

Mit der Vorgehensweisen erzielt man letztendlich eine quadratische Konvergenz (gegenüber der linearen ohne Spektralverschiebung) für das Verschwinden des (jeweils letzten) Subdiagonalelementes, d.h. $hN,N-1\rightarrow 0$ nach nur (sehr) wenigen Iterationen. Als vernünftiges Konvergenzkriterium fordert man, dass

$$hi+1,i(k)<\delta\ maxhii(k),hi+1,i+1(k) \qquad (2.2)$$

wenn δ die Maschinengenauigkeit ist. Dieses Kriterium erweist sich auch für betragsmäßig kleine Eigenwerte als geeignet.

Nach Konvergenz, d.h. wenn $hN,N-1\rightarrow 0$, gilt

$$\Rightarrow \sigma k=\lambda N=hNN(k) \qquad (\text{da „c“} = 0)$$

und die Matrix entkoppelt. Die restlichen $N-1$ Eigenwerte der Untermatrix $H\in RN-1\times(N-1)$ werden, wie schon oben erwähnt, analog ermittelt, wobei sich σkwiederum aus den Eigenwerten der letzten 2×2 Untermatrix (jetzt bzgl. der Indizes $N-2,N-1$) ergibt.

Aus dem bisher Ausgeführten ist der wesentlicher Vorteil des QR-Algorithmus sofort offensichtlich. Mit jedem berechneten Eigenwert reduziert sich die Ordnung der noch zu bearbeitenden Matrix, und man gewinnt einen zusätzlichen Zeitvorteil.

Nachdem wir nun σk spezifiziert haben, lautet der QR-Algorithmus mit expliziter Spektralverschiebung pro Schritt *im Prinzip* folgendermaßen

$$Hk+1=UN-1T\ldots U2TU1TQk'THk-\sigma k1U1U2\ldots\ldots UN-1Qk'+\sigma k1 \qquad (2.3)$$

wobei Ui die entsprechenden Rotationsmatrizen $Ui=Ui,i+1$ sind.

Ein dieser Gleichung immanentes Problem scheint es nun zu sein, dass pro Schritt die Drehwinkel aller Einzelrotationen Ui abgespeichert werden müssen, was einen zusätzlichen Zeit- und Verwaltungsaufwand erfordern würde. Auf Grund der Assoziativität von Matrixmultiplikationen lässt sich die Reihenfolge der Multiplikationen allerdings so wählen, dass dieses Problem erst gar nicht auftritt.

Wir nehmen dazu an, dass für die $A\in R6\times 6$ Operationen

$$U2TU1THk-\sigma k1$$

schon durchgeführt wurden. Daraus resultiert folgende Matrixstruktur

$$\times\times 0\times\times\times\times\times\times\times\times\times 0000\times\times\times\times\times\times\times 00000\times 00\times\times\times\times$$

Die ersten beiden Zeilen wurden dabei durch die Transformationen U1T=UT1,2 und U2T=UT2,3 modifiziert. Die folgenden Rotationen U3T,U4T,…. betreffen nur noch die dritte und folgende Zeilen, d.h. Zeile 1 und 2 und *insbesondere Spalte 1 und 2 dieser Zeilen* werden durch die weitergehende Zerlegung nicht mehr verändert. Deshalb kann nun die Operation ·U1 (Rechtsmultiplikation) durchgeführt werden, da sie „nur" Linearkombinationen der Spalten 1 und 2 bildet. Den nächsten Schritt bildet dann die Linksmultiplikation von U3T , danach die Rechtmultiplikation ·U2 usw. Mit dieser Vorgehensweise müssen die jeweiligen Drehwinkel nicht abgespeichert werden, und das gesamte gestaffelte Verfahren lässt sich wie folgt beschreiben.

- Schritt 1 : U1T von links, merke Drehwinkel (alt)
- Schritt i:i≤2≤N-1

a) UiT von links, neuer Drehwinkel: QR-Zerlegung in i, (i+1) ter Zeile
   b) Ui-1 von rechts mit altem Drehwinkel: Aufbau der H-Matrix in i, (i-1) ter Spalte
   c) alter Drehwinkel = neuer Drehwinkel
   d) fortlaufend Subtraktion von σkund finale Addition für Diagonale

- Schritt N:UN-1 von rechts, Addition von σk zu Diagonalelement N-1,N-1 und N;N

Komplexe Eigenwerte: Der QR-Doppelschritt

Wie wir im vorangegangenen Abschnitt (und im obigen Beispiel) gesehen haben, können bei der „Diagonalisierung" von reellen Matrizen natürlich auch konjugiert komplexe Eigenwerte auftreten, die wir mit unserer bisherigen Vorgehensweise nicht erfassen konnten: Einerseits ergibt die Wahl von σk=hNN(k) ∈ R entsprechend Gl. (1.9) keine vernünftige Näherung für einen komplexen Eigenwert, andererseits ergibt ein komplexer Wert von σk entsprechend Gl. (2.0) eine Matrix mit komplexer Diagonalen, die wir bislang nicht bearbeiten könnten. Der sog. „QR-Doppelschritt" löst diese Problematik. Als Vorbereitung auf diesen betrachten wir noch einmal die quadratische 2×2 Untermatrix in der „linken unteren" Ecke,

$$Ck=hN-1,N-1hN-1,NhN,N-1hNN=abcd$$

jetzt aber nicht nur mit den dazugehörigen reellen, sondern ggf. auch komplexen Eigenwerten und definieren

$$\lambda 1=\alpha+\beta \quad \lambda 1=\alpha+i-\beta \quad \lambda 2=\alpha-\beta \quad \lambda 2=\alpha-i-\beta \qquad (2.4)$$

wobei

$$\alpha = a+d2 \text{ und } \beta = a+d22-ad-bc$$

die Diskriminante des Problems ist.

Falls die Eigenwerte nun komplex sind, definieren wir den QR-Doppelschritt als zweifache UR-Transformation, einmal mit σk=λ1 und danach mit σk+1=λ2, wobei die UR-Transformation das komplexe Analogon der QR-Transformation ist, und die zugehörige UR-Zerlegung durch folgenden Satz spezifiziert ist.

Satz. Für jede komplexe Matrix A ∈ CN×N existiert die unitäre UR-Zerlegung

$$A=U*R \qquad (2.5)$$

wobei U eine unitäre Matrix (U†=U-1) und R eine komplexe Rechtsdreiecksmatrix mit reeller Diagonalen ist (ohne Beweis).

Der QR-Doppelschritt als zweifache UR-Transformation lässt sich nun folgendermaßen darstellen:

a) erste Zerlegung

$$Hk-\sigma k1=UkRk \qquad (2.6)$$

wobei $(Hk-\sigma k1)$ eine komplexe Diagonale hat (s.o.).

b) und zugehörige Transformation

$$Hk+1=RkUk+\sigma k1=Uk^{\dagger}HkUk \qquad (2.7)$$

c) zweite Zerlegung

$$Hk+1-\sigma k+11=Uk+1Rk+1 \qquad (2.8)$$

d) und zugehörige Transformation

$$Hk+2=Rk+1Uk+1+\sigma k+11=Uk+1^{\dagger}Hk+1Uk+1 \qquad (2.9)$$

Mit (2.7) finden wir also zunächst

$$Hk+2=UkUk+1^{\dagger}HkUkUk+1=QTHkQ \qquad (3.4)$$

und $Hk+2$ ist orthogonal ähnlich zu Hk, und wiederum eine Hessenberg-Matrix
(UR-Transformationen erhalten die Hessenberg-Form).
Damit ergibt sich folgender Algorithmus für die Lösung des Eigenwertproblems einer reellen Matrix mit komplexen Eigenwerten

i) Man berechne $X=Hk2-sHk+t1$ mit s und t aus $\sigma k, \sigma k+1=\lambda 1,2$, ,
ii) zerlege dann die reelle Matrix $X=Q\cdot R$ und
iii) führe die Transformation mit diesem Q durch: $Hk+2=QTHkQ$.
iv) Die explizite Subtraktion und (finale) Addition der σ, die im Falle von reellen Eigenwerten notwendig war, entfällt hier, da die Spektralverschiebung schon bei der Berechnung von X berücksichtigt wird und bei der Transformation (d.h. der Rechtmultiplikation mit $Q=UkUk+1$) wieder rückgängig gemacht wird.

Bestimmung der Eigenvektoren nach Durchführung des QR-Algorithmus

Im Folgenden wollen wir kurz diskutieren, wie man nach einer Bestimmung der Eigenwerte mittels QR-Algorithmus die dazugehörigen Eigenvektoren ermittelt.

Direkte Berechnung

Die direkte Berechnung ist sehr aufwendig, da sie eine konsequente Buchhalten über alle orthogonalen Transformationen erfordert, die letztendlich zu der die Eigenwerte darstellenden Quasidreiecksmatrix R (1.4) geführt haben. Dies sind die Transformationen

$$Q=Q1Q2\ldots.QM$$

wenn die Ausgangsmatrix A über Givens in eine Hessenberg-Matrix H und diese auf R QR-transformiert wurde, so dass final
$R=QTAQ$

gilt. Mit der originalen Eigenwertaufgabe

$$A \cdot x = \lambda \cdot x$$

folgt dann

$$QTAQR \cdot QTxy = \lambda\, QTxy$$

d.h. die Eigenwertaufgabe reduziert sich zunächst auf die Lösung von

$$R \cdot y = \lambda\, y \text{ mit } y := QTx$$

was sich aufgrund der Quasidreieckstruktur von R bei bekanntem λi einfach durchführen lässt. Zur Bestimmung der tatsächlich gesuchten Eigenvektoren ist dann allerding die Kenntnis von Q unerlässlich,

$$xi = Q \cdot yi$$

Inverse Iteration

Das zweite Verfahren ist allgemeinerer Natur und setzt nur voraus, dass man die Eigenwerte einer Matrix zuvor schon bestimmt hat (wie auch immer). Aufgrund der iterativen Natur dieses Verfahrens eignet es sich auch zu einer Verbesserung der Qualität von schon zuvor bestimmten Eigenvektoren, z.B. mittels des Jacobi-Verfahrens.
Wir nehmen dazu an, dass für die Originalmatrix A ein numerischer Näherungswert λ , für den entsprechenden „exakten" Eigenwert λi bekannt ist. Definiert man nun die Folge

$$A - \lambda\, 1 zk = zk-1 \qquad (3.5)$$

mit einem geeignet definierten Startwert $z0$, zum Beispiel $1,1,1,\ldots,1t$ oder einem Schätzwert für den gesuchten Eigenvektor, dann konvergiert die Folge zk, $k=1,2,\ldots$ (unter einigen Einschränkungen) gegen den zu λi gehörigen Eigenvektor.

Der Beweis zu dieser Aussage wird im Folgenden skizziert. Unter der (fast immer gerechtfertigten) Annahme, dass A einen vollständigen Satz von Eigenvektoren besitzt (xj, $j=1,\ldots,N$), lassen sich die iterierten Werte z nach diesen Eigenvektoren entwickeln, d.h.

$$zk-1 = \beta\, jxj$$

$$zk = \alpha\, jxj$$

Einsetzen in (3.5) liefert

$$A - \lambda\, 1\, \alpha\, jxj = \beta\, jxj$$

$$\alpha j(\lambda j - \lambda)xj = \beta\, jxj$$

$$\Rightarrow\ \alpha j = \beta\, j(\lambda j - \lambda)$$

Demzufolge lässt sich also zk durch die Koeffizienten von $zk-1$ ausdrücken,

$$zk = \beta\, j(\lambda j - \lambda)xj$$

Sei nun der Startvektor durch die Entwicklung

$$z0 = cjxj$$

gegeben, dann folgt sofort, dass

$$zk=cj(\lambda j-\lambda)kxj \qquad (3.6)$$

ist. (Man beachte die Potenz k im Nenner!). Falls nun der Eigenwert nicht entartet und λ eine guter Schätzwert für λi ist,

$$ci \neq 0 \text{ und } 0<\lambda i-\lambda \ll \min j \neq i \lambda j-\lambda \qquad (3.7)$$

dann gilt mit

$$zk=1 \lambda i-\lambda kcixi+j \neq iNcj \lambda i-\lambda k \lambda j-\lambda kxj \rightarrow 0 \qquad (3.8)$$

dass zk rasch gegen den gesuchten Eigenvektor xi konvergiert, falls nach jeder Iteration zk normiert wird!

Das Problem dieser Vorgehensweise liegt natürlich darin, dass wir de facto das lineare Gleichungssystem

$$A-\lambda \ 1y=b$$

mit $y = zk$ und $b=zk-1$ lösen müssen, zum Beispiel über LU-Zerlegung (pro Eigenwert λ ist dann nur eine einmalige Zerlegung notwendig). Die (zu zerlegende) Matrix $A-\lambda \ 1$ ist aber nahezu singulär, da $\lambda \approx \lambda i$, d.h. das Gleichungssystem ist äußerst schlecht konditioniert, was zu großen Fehlern in der Lösung führt ü, egal wie „gut" der Lösungsalgorithmus ist.
Dass die inverse Iteration überhaupt vernünftige Resultate liefert, liegt nun daran, dass im allgemeinen dieser Fehler, der bei der Lösung von linearen Gleichungssystemen gemacht wird, hier:

$$zk-zk$$

(mit zk der numerische und zk der „exakten" Lösung), eine dominante Komponente in Richtung des Eigenvektors des betragskleinsten Eigenwertes der das Gleichungssystem darstellenden Matrix (hier: $A-\lambda \ 1$) hat.

Auf Grund der Voraussetzung (3.7) ist der betragskleinste Eigenwert aber $\lambda i-\lambda$, d.h. der resultierende numerische Fehler liegt hauptsächlich in Richtung des gesuchten Eigenvektors xi, d.h. wirkt sich kaum aus! *Das ist der eigentliche Grund, warum die inverse Iteration funktioniert.*