Hochschule für Technik, Wirtschaft und Kultur Leipzig
Fakultät Informatik, Mathematik und Naturwissenschaften

Skriptum

zur Vorlesung

Numerik I

von

Prof. Dr. rer. nat. habil. Bernd Engelmann

(Überarbeitung vom September 2014)

Inhaltsverzeichnis

Literatur

Deuflhard,P.; Hohmann,A.: Numerical Analysis in Modern Scientific Computing. Springer-Verlag 2010.

Kielbasinski,A.; Schwetlick, H.: Numerische lineare Algebra. Verlag der Wiss., Berlin 1988.

Knorrenschild, M.: Numerische Mathematik, Reihe Mathematik-Studienhilfen. Fachbuchverlag Leipzig im Carl-Hanser-Verlag München, Wien 2010.

Locher, F.: Numerische Mathematik für Informatiker. Springer-Verlag, 2008.

Preuß, W.; Wenisch, G.: Numerische Mathematik. Fachbuchverlag Leipzig im Carl-Hanser-Verlag München, Wien 2001.

Schwarz, H.R.; Köckler, N.: Numerische Mathematik. Vieweg- und Teubner-Verlag, 2011.

Stoer,J.; Bulirsch,R.: Numerische Mathematik, Bd.1. Springer-Verlag, 10. Auflage 2007.

Stoer,J.; Bulirsch, R.: Numerische Mathematik, Bd.2. Springer-Verlag, 6. Auflage 2012.

1. Grundlagen des numerischen Rechnens und der Fehleranalyse

Kennzeichen der numerisch orientierten Mathematik:

1. Man konstruiert eine Lösung (d.h. man begnügt sich nicht mit Nachweis von Existenz und Eindeutigkeit). Dabei beschränkt man sich i.a. auf die Konstruktion von Näherungen einer exakten Lösung, wobei der Fehler bei Steigerung des numerischen Aufwandes verkleinert werden kann.

2. Hilfsmittel zur Konstruktion von Näherungslösungen sind ausschließlich die arithmetischen Grundoperationen (die auf einem Digitalrechner realisiert werden können: +, -, *, /). Eingangsdaten und interne Größen sind Computerzahlen, die nur eine endliche Genauigkeit erlauben und durch Rundung verfälscht sind.

Ziel des Abschnitts: Grundlegende Fragen der Zahlendarstellung, der Arithmetik mit endlicher Stellenzahl und der Fehlerfortpflanzung kennen lernen.

1.1. Gleitpunktzahlen, Maschinenzahlen

Die Standardform der digitalen Darstellung von Informationen ist die in Gestalt einer geordneten Folge von Ziffern; dieses Prinzip wird z.B. beim System der Dezimalzahlen verwendet. Die gleiche Idee liegt der Darstellung und Speicherung von Zahlen im Computer zugrunde.

Bezeichnungen:

bit (binary digit) : Grundeinheit der Information

byte : kleinste adressierbare Speichereinheit; bestehend aus 8 bit

word : Wort, eine feste Anzahl von byte zur Speicherung einer Zahl

bias, offset : ganzzahlige Verschiebung.

Für die Darstellung einer Zahl x in einem Datenformat steht eine feste Anzahl n von Dualstellen (n=Wortlänge) zur Verfügung. Wenn überhaupt, so kann die Wortlänge nur auf 2n, 3n,… erweitert werden. Ein Wort kann auf zwei prinzipielle Arten zur Darstellung einer Zahl verwendet werden.

(a) Festpunktdarstellung
Neben der Wortlänge n sind außerdem die Anzahl n_1 bzw. n_2 der Stellen vor bzw. nach dem Punkt festgelegt, welcher den ganzzahligen und gebrochenen Anteil der Zahl trennt. Zur Einsparung einer Vorzeichenstelle kann vor der Speicherung eine festgelegte Verschiebung (bias) zu x addiert werden. Bei Festpunktdarstellung ist der Zahlenvorrat sehr eingeschränkt. Eine Darstellungsform für Integer-Zahlen ($n_2 = 0$).

(b) Gleitpunktdarstellung
In Gleitpunktdarstellung wird jede Zahl x ($x \neq 0$) in der Form dargestellt

$$x = (-1)^s \, m \; B^E$$

B … Basis des Zahlensystems (i.a. $B = 2$, 8 oder 16)
m … Mantisse, gebrochener Anteil
E … vorzeichenbehafteter Exponent (ganzzahlig)
s … Vorzeichen (Signum) der Mantisse ($s = 0 \; oder \; 1$)

Es stehen eine feste Anzahl *t* Stellen für die Mantisse und *l* Stellen für den Exponenten zur Verfügung $t + l + 1 = n$.

Um eine Gleitpunktdarstellung eindeutig zu machen, wird sie normalisiert:

Def. 1.1: Eine Gleitpunktdarstellung heißt normalisiert, wenn die erste Ziffer m_1 der Mantisse m verschieden von Null ist, d.h. es gilt

$$B^{-1} \le m < 1$$

$$m = m_1 B^{-1} + m_2 B^{-2} + \ldots + m_t B^{-t} = B^{-t} (m_1 B^{t-1} + m_2 B^{t-2} + \ldots + m_t B^0).$$

Die Menge G der t-stelligen, normalisierten Gleitpunktzahlen zur Basis B ist dann $G = \{ M \cdot B^{E-t} \mid M = 0 \; oder \; B^{t-1} \le | M | \le B^t - 1 \}$.

Bei normalisierter Darstellung liegt der Wert der Mantisse zwischen B^{-1} und $1 - B^{-t}$.

Da der Exponent E eine vorzeichenbehaftete ganze Zahl ist, kann er in geeigneter Festpunktdarstellung gespeichert werden, wobei zur Einsparung des Vorzeichens eine feste Verschiebung addiert wird (biased exponent *e*). Der verschobene Exponent *e* ist eingeschränkt durch den Speicherbereich des Exponenten.

IEEE Standard - single precision:

Ein numerischer Wert (IEEE-Standard real *4) benötigt 4 byte Speicherplatz;

bit 1 -23 : Mantisse m;
bit 24-31 : verschobener Exponent $e = E + bias$ (bias=127)
bit 32 : Vorzeichen s der Mantisse

Bereich der darstellbaren Zahlen:
a) Mantisse: 24 bit (davon ein implizites bit), d.h. 24-stellige Dualzahl entspricht 7-8 Dezimalstellen
b) Exponent: 8 bit, d.h. größter verschobener Exponent e ist $2^8 - 1 = 255$
$E + 127 \; \varepsilon \{ 0, \ldots, 255 \}, d.h. \; E \; \varepsilon \{ -127, \ldots, 128 \}.$

Damit gibt es eine kleinste bzw. größte positive Zahl $x_{\min}$, $x_{\max}$ die im Rechner dargestellt werden können.

Def. 1.2: Die Elemente der Menge $M \subset R$ von reellen Zahlen, die in der Maschine (dem zugrunde liegenden Datenformat) exakt dargestellt werden können, heißen Maschinenzahlen (des Datenformates)

$$M = M(B,t,l).$$

1.2. RUNDUNG

Die Menge M der Maschinenzahlen ist endlich. Damit entsteht die Frage, wie man eine Zahl $x \notin M$ durch eine Maschinenzahl $g \in M$ darstellen kann. Das Verfahren heißt Rundung $x \to rd(x)$, der dabei auftretende Fehler heißt Rundungsfehler oder Darstellungsfehler der Zahl x.

Def. 1.3: $rd : R \to M$ ist eine Abbildung, die jedem $x \varepsilon R$ das nächstgelegene $g \varepsilon M$ zuordnet $\quad |x - rd(x)| \leq |x - g|, \ \forall\, g \varepsilon M$.

Durchführung der Rundung (am Beispiel der Basis $B = 10$)

1. $x \notin M$ wird in normalisierte Form gebracht $x = (-1)^s m \ 10^E, mit \ \ m \geq 10^{-1}$
   mit der Dezimaldarstellung von m
   $m = 0.m_1 m_2 \dots m_t m_{t+1} \dots \quad m_1 \neq 0$
2. Man bildet
   $$m' = \begin{cases} 0.m_1 m_2 \dots m_t & für \ \ 0 \leq m_{t+1} \leq 4 \\ 0.m_1 m_2 \dots m_t + 10^{-t} & für \ \ m_{t+1} \geq 5 \end{cases},$$
   d.h. man erhöht die Stelle m_t um 1, falls $m_{t+1} \geq 5$ ist und schneidet nach der t-ten Ziffer ab. Besonderheiten treten für $m_t = 9$ wegen Stellenüberträgen auf weiter vorn liegende Stellen auf, eventuell ist eine Renormalisierung der Mantisse nötig.
3. $rd(x) = (-1)^s \ m' \cdot 10^E$

Relativer Fehler der Rundung:
Für den relativen Fehler des Wertes rd(x) gilt

$$\left| \frac{rd(x) - x}{x} \right| \leq \frac{0.5 \cdot 10^{-t}}{|m|} \leq 5 \cdot 10^{-t},$$

bzw. mit $eps_M = 5 \cdot 10^{-t}$ (relative Maschinengenauigkeit) gilt

$$rd(x) = x(1 + \varepsilon) \qquad |\varepsilon| \leq eps_M .$$

Bemerkung:
1. Analoges gilt für die Basis $B = 2$, wenn $eps_M = 1 \cdot 2^{-t}$ gesetzt wird.
2. $x \notin M$ muss nicht exakt gegeben sein ,es genügt offenbar, die (t+1)-te Mantissenstelle zu kennen.

Somit erhält man die Aussage:

Satz 1.4 (Rundungsfehlergesetz): Für alle $x \varepsilon [-x_{\max}, -x_{\min}] \cup [x_{\min}, x_{\max}] \subset R$ gilt

$$rd(x) = x(1 + \varepsilon) \ mit \ |\varepsilon| < eps_M \qquad (1.1)$$

wobei $eps_M = \frac{B}{2} B^{-t}$ als relative Maschinengenauigkeit bezeichnet wird.

Bemerkung: Die angegebene Art der Rundung wird auch als symmetrische Rundung bzw. korrekte Rundung bezeichnet. In verschiedenen Systemen wird eine unsymmetrische Rundung (Regel des Abschneidens) verwendet:
Alle Ziffern der Mantisse m nach der t-ten Stelle werden weggelassen.

In Satz 1.4 ist dann eps_M durch

$$eps_{M'} = B \cdot B^{-t} = B^{-t+1}$$

zu ersetzen.

Sonderfälle der Rundung: Einige Zahlen können nicht dargestellt werden, da sie außerhalb des Bereichs der Maschinenzahlen liegen. Sind $E_{\min}$ und $E_{\max}$ der kleinste bzw. größte erlaubte Exponent, so sind

$$x_{\min} = B^{(E_{\min}-1)}, x_{\max} = B^{E_{\max}}(1 - B^{-t})$$

die kleinste bzw. größte (normalisiert) darstellbare positive Zahl. Liegt x außerhalb des Bereiches der darstellbaren Zahlen, so gilt die Fehlerabschätzung (1.1) i.a. nicht mehr:

(a) Exponentenüberlauf (overflow):
Für $|x| > x_{\max}$ wird eine Warnung gesetzt "overflow".

.

(b) Exponentenunterlauf (underflow):
Für $|x| < x_{\min}$ gilt $rd(x) := 0$. Der absolute Fehler ist dann $|rd(x) - x| = |x|$ und der relative Fehler ist folglich stets 1.

Bemerkung.: Um den Bereich der Computernull möglichst klein zu halten, gibt es zwischen den normalisierten Realzahlen und dem Bereich der Computernull im IEEE Standard den Bereich der subnormalen Werte, in diesem gilt die Rundungsfehlerabschätzung (1.1) nicht mehr, da mit verkürzten Mantissen gearbeitet wird.

1.3. GLEITPUNKTARITHMETIK

Sind x, y Maschinenzahlen, so muss das Resultat einer arithmetischen Operation nicht wieder in der Menge M der Maschinenzahlen liegen.

Schema einer Addition:
Sind die beiden Zahlen $x, y \in M(B,t,l)$ zu addieren, so werden sie in den Registern r_1, r_2 gespeichert. Das Register r_2 der betragskleineren Zahl wird soweit nach rechts verschoben bis die Exponenten beider Register übereinstimmen. Das Ergebnis der Addition wird im Resultatregister R gespeichert, welches eine Überlaufstelle vor dem Punkt besitzt. Bei Auftreten einer Überlaufziffer wird R renormalisiert. Da in R i.a. mehr als t Ziffern sind, tritt bei Rundung auf t Ziffern ein Rundungsfehler auf.

Statt der exakten Operation müssen wir somit von Ersatzoperationen bzw. Gleitpunktoperationen sprechen.

> **Def. 1.5**: Sei $x, y \in M$ und $op \in \{+,-,*,\div\}$. Dann heißt $fl(x\ op\ y)$ das Ergebnis der Gleitpunktoperation op (fl … floating point operation). Die Gleitpunktarithmetik rundet korrekt, falls gilt
> $$fl(x\ op\ y) = rd(x\ op\ y) = (x\ op\ y)(1+\alpha) \qquad \forall\, x, y \in M \ \ und \ \ |\alpha| < eps_M .$$

Bemerkungen:

1. Bei der Gleitpunktarithmetik bleiben von den strengen mathematischen Gesetzen nur die Kommutativität der Addition und der Multiplikation erhalten. Assoziativ- und Distributivgesetze gelten i.a. nicht mehr.

2. Der relative Fehler α einer Gleitpunktoperation hängt i.a. von x, y und der konkreten Arithmetik ab, er ist jedoch beschränkt durch die relative Maschinengenauigkeit eps_M.

3. Im Allgemeinen werden in den Registern r_1, r_2 nicht alle t gültigen Stellen der Operanten x und y mitgeführt, so dass neben dem Rundungsfehler zusätzliche Fehler auftreten können. Praktisch ist der relative Fehler α immer durch ein kleines Vielfaches von eps_M beschränkt.

4. Die korrekt rundende Arithmetik kann realisiert werden, wenn das Register R mindestens 2 Schutzstellen mitführt, d.h. über $t+2$ Nachkommastellen verfügt.

Auslöschung: Verschiedene Berechnungen bergen das Risiko, dass der relative Fehler wesentlich größer als die relative Maschinengenauigkeit ist. Dies tritt insbesondere dann auf, wenn nahezu gleichgroße Zahlen x, y subtrahiert werden.
Ist $|x-y|$ klein in bezug auf $|x|$, so ist der Rundungsfehler wesentlich größer als eps_M. Das ist dadurch bedingt, dass die führenden Stellen von x und y übereinstimmen und bei Subtraktion ausgelöscht werden. Die führenden Stellen der Differenz werden dann durch nachgeordnete Stellen von x und y bestimmt. Diese sind i.a. aber bereits durch Rundungsfehler verfälscht und werden durch Auslöschung verstärkt.

Zwei Interpretation der Fehler bei Gleitpunktoperationen

1. Vorwärtsinterpretation: Das Resultat $fl(x\ op\ y)$ hat im Vergleich zum exakten Wert $x\ op\ y$ einen kleinen relativen Fehler.

2. Rückwärtsinterpretation (Wilkinson): Das Resultat $fl(x\ op\ y)$ ist exakt zu relativ wenig geänderten Operanden, d.h.
$fl(x\ op\ y) = x(1+\alpha)\ op\ y(1+\beta)$ mit $|\alpha|,|\beta| < eps_M$

Konsequenz: Zwei bei exakter Rechnung äquivalente Lösungswege müssen bei endlicher Arithmetik nicht mehr äquivalent sein. Aussagen über Rundungsfehler müssen sich also stets auf ein Lösungsverfahren beziehen, bei dem alle Operationen in der Reihenfolge eindeutig festgelegt sind.

> **Def. 1.6**: Ein Algorithmus ist eine in der Reihenfolge eindeutig festgelegte Folge von endlich vielen elementaren Operationen.

1.4. Fehlerfortpflanzung

Fehlermessung
Ist x der exakte Wert einer Größe und $\tilde{x}$ der berechnete oder gemessene Näherungswert, so kann der Fehler in x auf 2 Arten gemessen werden:

(a) absoluter Fehler: $|\delta x| = |x - \tilde{x}|$ abs.Fehler von x,
$\Delta x \geq |x - \tilde{x}|$ Schranke des absoluten Fehlers

(b) relativer Fehler: $\frac{|\delta x|}{|x|} = \left|\frac{x-\tilde{x}}{x}\right|$ $(x \neq 0)$ rel.Fehler von x,

$\varepsilon_x \geq \frac{|\delta x|}{|x|}$ Schranke des relativen Fehlers

Relative Fehler sind den Gleitpunktoperationen inhärent, wie die Abschnitte 1.2., 1.3. zeigen. Für $x = 0$ ist der relative Fehler nicht erklärt, so dass bei der Berechnung von ε_x für x nahe Null Vorsicht geboten ist. Ein Fehlermaß, welches dies vermeidet, ist z. B. durch den Ausdruck

$$\frac{|x - \tilde{x}|}{1 + |x|}$$

gegeben. Dieses Maß besitzt ähnliche Eigenschaften wie der relative Fehler für $|x| >> 1$ bzw. es ist für $|x| < 1$ ähnlich dem absoluten Fehler. Das Maß ist insbesondere für Abbruchtests geeignet.

Fehlerfortpflanzung in Formeln

Berechnungen bestehen oft aus einer Folge von Operationen, die bereits berechnete (und damit fehlerbehaftete) bzw. gemessene Größen einbeziehen. Eine komplexe Berechnung besteht aus Eingangs- und Resultatdaten:

geg.: $x = (x_1, \ldots, x_n)^T \varepsilon D \subset R^n$ Eingangsdaten
ges.: $y = (y_1, \ldots, y_m)^T \varepsilon R^m$ Resultatdaten

Der Resultatvektor y wird mittels eines Algorithmus und der Eingangsdaten bestimmt. Ein Algorithmus definiert somit eine vektorwertige Abbildung der Eingangsdaten auf die Ausgangsdaten

$\varphi : D \to R^m$ bzw. $y = \varphi(x)$

oder koordinatenweise

$$y_i = \varphi_i(x_1, \ldots, x_n) \quad i = 1, \ldots, m .$$

Liegen für die Eingabedaten $x = (x_1, \ldots, x_n)$ nur Näherungen $\tilde{x} = (\tilde{x}_1, \ldots, \tilde{x}_n)$ vor, so werden die Resultate y_i zu $\tilde{y}_i$ verfälscht $\tilde{y}_i = \varphi_i(\tilde{x}_1, \ldots, \tilde{x}_n) \quad i = 1,2,\ldots,m$. Wir untersuchen den Einfluss der Eingangsfehler auf die Resultate.

Unter der Voraussetzung, dass die Abbildung φ stetig differenzierbar ist, erhält man mit Hilfe des Mittelwertsatzes der Differenzialrechnung

$$\tilde{y}_i = \varphi_i(\tilde{x}_1, \ldots, \tilde{x}_n) = \varphi_i(x_1, \ldots, x_n) + \sum_{j=1}^{n} \frac{\partial \varphi_i}{\partial x_j} (\tilde{x}_j - x_j),$$

wobei die Ableitungen an gewissen Zwischenstellen $\psi^i = x + \theta_i * (x - \tilde{x})$ zu berechnen sind. Für $|x_i - \tilde{x}_i| \leq \Delta x_i$ liegen $x, \tilde{x}$ *und* ψ^i in einem n-dimensionalen symmetrischen

Unsicherheitsintervall mit dem Mittelpunkt $\tilde{x}$. Damit gilt die Maximalfehlerabschätzung:

$$\Delta y_i \leq \sum_{j=1}^{n} \left(\max_{x \in [\tilde{x} - \Delta x, \tilde{x} + \Delta x]} \left| \frac{\partial \varphi_i}{\partial x_j} \right| \right) \Delta x_j .$$

Häufig ist man nur an einer Aussage über die zu erwartende Größenordnung des

Fehlers interessiert bzw. man möchte die gültigen Stellen des Resultats bestimmen. Für kleine Unsicherheit Δx_i in allen Variablen ist das Unsicherheitsintervall für x klein und die Ableitungen variieren nur wenig über dem Intervall. In 1.Näherung (bezeichnet durch $\doteq$) können somit die Ableitungen im Mittelpunkt $\tilde{x}$ des Intervalls berechnet werden. Mit so berechneten Größen erhält man eine Schätzung des Fehlers, die man als linearisierte Fehlerschätzung bezeichnet:

$$\Delta y_i = |\tilde{y}_i - y_i| \doteq \sum_{j=1}^{n} \left|\frac{\partial \varphi_i}{\partial x_j}\right|_{x=\tilde{x}} \cdot \Delta x_j$$

$$\varepsilon_{y_i} = \frac{\Delta y_i}{|y_i|} \doteq \sum_{j=1}^{n} \left|\frac{\partial \varphi_i}{\partial x_j} \cdot \frac{x_j}{\varphi_i}\right|_{x=\tilde{x}} \varepsilon_{x_i}$$

Def. 1.7: Die Proportionalitätsfaktoren $\left|\frac{\partial \varphi_i}{\partial x_j}\right|$ und $\left|\frac{\partial \varphi_i}{\partial x_j} \cdot \frac{x_j}{\varphi_i}\right|$, welche die Empfindlichkeit messen, mit der $y_i = \varphi_i(x_1,\ldots,x_n)$ auf absolute bzw. relative Änderungen von x_j reagiert, heißen absolute bzw. relative Konditionszahlen (Faktoren der Fehlerverstärkung).

Bezüglich der Fehlerfortpflanzung bei elementaren arithmetischen Operationen gilt für die relativen Fehler die Aussage:

Satz 1.8:

a) $\varphi(x,y) = x * y \Rightarrow \varepsilon_{xy} \doteq \varepsilon_x + \varepsilon_y$,

b) $\varphi(x,y) = x / y \Rightarrow \varepsilon_{x/y} \doteq \varepsilon_x - \varepsilon_y$,

c) $\varphi(x,y) = x + y \Rightarrow \varepsilon_{x+y} \doteq \frac{x}{x+y}\varepsilon_x + \frac{y}{x+y}\varepsilon_y \quad (falls\ x+y \neq 0)$

d) $\varphi(x) = \sqrt{x} \Rightarrow \varepsilon \doteq \frac{1}{2}\varepsilon_x$

Bemerkung: Die Operationen *, /, $\sqrt{\ }$ sind somit bezüglich der Fehlerfortpflanzung harmlose Operationen. Für "±" gilt dies nur, wenn die Konditionszahlen $|\frac{x}{x \pm y}|, |\frac{y}{x \pm y}|$ nicht zu groß sind (Auslöschung).

1.5. Kondition eines Problems und Stabilität von Algorithmen

Kondition

Unter der Kondition eines Problems versteht man die Auswirkung von Störungen der Problemdaten auf das exakte Ergebnis. Ein Problem nennt man gut konditioniert, wenn kleine Störungen nur kleine Änderungen der exakten Lösung zur Folge haben und schlecht konditioniert (ill-conditioned), wenn kleine Störungen starke Änderungen der Lösung bewirken können.

Beachte: Schlechte Kondition ist eine dem Problem innewohnende Eigenschaft und ist nicht ursächlich durch Gleitpunktarithmetik bei der Problemlösung bedingt. Allerdings wirkt sich Gleitpunktrechnung bei schlechter Kondition stärker auf die erreichbare Genauigkeit aus.

Bsp. 1: Berechnung des Schnittpunktes zweier Geraden
(Lösung eines linearen Gleichungssystems, Störung im Absolutglied von g_2)

(a) g_1, g_2 nahezu senkrecht

(b) g_1, g_2 nahezu parallel

gut konditioniert $\Delta s \approx \varepsilon$

schlecht konditioniert $\Delta s >> \varepsilon$

Bsp. 2: Lösung einer quadratischen Gleichung $y^2 + 2\,p\,y - q = 0$

$y_1 = \varphi(p,q) := -p + \sqrt{p^2+q}$ zu berechnen

$$\frac{\partial \varphi}{\partial p} = -\frac{y_1}{\sqrt{p^2+q}}; \qquad \frac{\partial \varphi}{\partial q} = \frac{1}{2\sqrt{p^2+q}}$$

$$\varepsilon_{y_1} \doteq \left|\frac{-y_1}{\sqrt{p^2+q}} \bullet \frac{p}{y_1}\right| \varepsilon_p + \left|\frac{1}{2\sqrt{p^2+q}} \bullet \frac{q}{y_1}\right| \varepsilon_q =$$

$$= \left|\frac{p}{\sqrt{p^2+q}}\right| \varepsilon_p + \left|\frac{p+\sqrt{p^2+q}}{2\sqrt{p^2+q}}\right| \varepsilon_q = K_p\,\varepsilon_p + K_q\,\varepsilon_q$$

Für $q > 0$ gilt $K_p \le 1, K_q \le 1$; d.h. die Berechnung von y_1 ist gut konditioniert. Falls aber $q \approx -p^2$, und p ist groß, so ist $K_p >> 1$, d.h. es liegt schlechte Kondition vor.

Stabilität eines Algorithmus

Die Resultate bei der numerischen Lösung eines Problems sind Daten, die durch eine Folge von Gleitpunktoperationen (Algorithmus) erzeugt wurden. Man möchte aufgrund der Fehleranalyse entscheiden, ob ein Algorithmus für eine Problemklasse brauchbar ist oder nicht.

Vorwärtsanalyse: Man könnte einen Algorithmus als brauchbar ansehen, wenn eine berechnete Näherungslösung $\tilde{y}$ nahe der exakten Lösung y liegt, d.h. man ist geneigt, eine Abschätzung der Form

$$\|y - \tilde{y}\| \le \delta$$

zu finden, wobei $\|\bullet\|$ ein geeignetes Abstandsmaß ist.
Die Existenz von schlecht konditionierten Problemen zeigt aber, dass die Fehlerschranke δ nicht für jedes Problem klein sein muss. Mit anderen Worten: Hätten wir einen Algorithmus für eine Problemklasse, der nur eine einzige Rundungsoperation enthält, und würden ihn auf ein schlecht konditioniertes Problem anwenden, so könnte eine starke Abweichung von exakter Lösung y und Näherungslösung $\tilde{y}$ auftreten. Man müsste den Algorithmus (und somit jeden Algorithmus) bei der Vorwärtsanalyse als unbefriedigend ansehen.

Rückwärtsanalyse: Im Gegensatz zur Vorwärtsanalyse, betrachtet man hier die berechnete Näherungslösung $\tilde{y}$ als exakte Lösung zu einem gestörten Ausgangsproblem mit einem Datensatz $\tilde{P}$. Die durch die Gleitpunktoperationen erzeugten Fehler werden somit rückwärts verfolgt und als Störungen der Eingangsdaten interpretiert. Bezeichnet P die Daten des Ausgangsproblems, so wird eine Abschätzung der Form

$$\left\| P - \tilde{P} \right\| \le \delta$$

in einer geeigneten Norm angestrebt. Man kann zeigen, dass δ für brauchbare Algorithmen klein ist. Ein Algorithmus A, für den eine solche Beziehung nachgewiesen werden kann, wird als numerisch stabil bezeichnet. Die Fehler bei der Berechnung der Lösung besitzen nur eine geringe Rückwirkung bezüglich der Abweichung vom Originalproblem.

1.6. RUNDUNGSFEHLERANALYSE

Wir wollen an zwei Beispielen das Problem der Rundungsfehleranalyse demonstrieren:

1. Summation: $s_m = \sum_{j=1}^{m} a_j, \; a_j \, \varepsilon \, M$

Man beachte, dass in der Menge M der Maschinenzahlen nicht das Assoziativgesetz gilt, so dass der Wert des Ergebnisses von der Summationsreihenfolge abhängt.

$$\tilde{s}_1 = a_1 \quad (\varepsilon_1 := 0)$$
$$\tilde{s}_2 = \tilde{s}_1 \oplus a_2 = (\tilde{s}_1 + a_2)(1+\varepsilon_2)$$
$$\tilde{s}_m = \tilde{s}_{m-1} \oplus a_m = (\tilde{s}_{m-1} + a_m)(1+\varepsilon_m)$$
$$\Rightarrow \tilde{s} := \tilde{s}_m = \sum_{j=1}^{m} a_j \prod_{i=j}^{m} (1+\varepsilon_i) = \sum_{j=1}^{m} a_j (1+\delta_j)$$

Wegen $|\delta_j| \le (m-j+1)\,eps_M$ nehmen die Fehlerschranken $|\delta_j|$ mit wachsendem Summationsindex j ab. Die Schranke für den Gesamtfehler

$$|\tilde{s} - s| \le \sum_{j=1}^{m} |a_j|\,|\delta_j|$$

wird also am kleinsten, wenn die betragsgrößten Summanden a_j zuletzt in der Summe auftreten, da sie dann mit den kleinsten δ_j multipliziert werden.

Merke: Man summiere von den betragskleinen nach den betragsgroßen Zahlen.

2. Skalarproduktbildung

Das Skalarprodukt von Vektoren ist eine wichtige Konstruktion, es tritt z.B. bei Matrizenmultiplikationen für jedes Element der Produktmatrix auf.

$$s = a^T b = \sum_{i=1}^{n} a_i \, b_i , \quad a_i , b_i \, \varepsilon \, M$$

Algorithmus: $s_0 := 0$

$$\text{für } i := 1,2,\ldots,n: \; s_i := s_{i-1} + a_i \, b_i$$

$$s := s_n$$

Der absolute Fehler ist klein; wenn jedoch die Vektoren *a* und *b* nahezu orthogonal sind, d.h. es gilt $s = \sum_{i=1}^{n} a_i \, b_i \approx 0$, so tritt ein unbeschränkter relativer Fehler $\frac{\Delta s}{s}$ auf.

Da die Skalarprodukt-Berechnung häufig auftritt, wird für sensitive Algorithmen eine Skalarprodukt-Berechnung mit verbesserter Genauigkeit (z.B. doppelter Wortlänge) empfohlen. Die Multiplikationen sind dann exakt, u. auch die Addition wird nahezu exakt.

Bemerkung: In modernen Numerik-Systemen werden wichtige Standardoperationen mit speziellen Algorithmen realisiert, die geringe numerische Fehler garantieren (z.B. die BLAS-Routinen für Vektor- und Matrixoperationen).

2. Normen von Vektoren und Matrizen

Zur quantitativen Beurteilung von numerischen Fehlern bei Vektoren und Matrizen benötigt man entsprechende Maße, die als Normen bezeichnet werden.

2.1. VEKTORNORMEN

Für jeden Vektor $x=(x_1,x_2,\ldots,x_n)^T \varepsilon R^n$ ist

$$|x|=\sqrt{x^T x}=\sqrt{x_1^2+\ldots+x_n^2}$$

als euklidische Länge von x ein natürliches Maß für die Beurteilung der Größe von x. Häufig ist es jedoch günstiger, anders definierte Maße zu verwenden, die ähnliche Eigenschaften wie der Betrag besitzen; derartige Maße werden auch als Normen bezeichnet.

Wesentliche Eigenschaften des Betrages :

(1) $|x|\geq 0 \quad \forall x \varepsilon R^n$ und $|x|=0$ nur für x=0;
(2) $|\lambda x|=|\lambda||x|$;
(3) $|x+y|\leq|x|+|y|$ (Dreiecksungleichung).

Die letzte Beziehung folgt aus

$$|x+y|^2=(x+y)^T(x+y)=x^T x+y^T y+2x^T y=|x|^2+|y|^2+2x^T y \ .$$

sowie

$$x^T y=|x||y|\cos(x,y)\leq|x||y| .$$

Damit gilt

$$|x+y|^2\leq|x|^2+|y|^2+2|x||y|=(|x|+|y|)^2 \ .$$

Def. 2.1: Eine Abbildung $\|\bullet\|:\mathrm{R}^n\to R$ heißt Vektornorm, wenn die folgenden Axiome erfüllt sind:

(N1) $\|x\|\geq 0 \quad \forall x \varepsilon R^n$ und $\|x\|=0$ nur für x=0.
(N2) $\|\lambda x\|=|\lambda|\|x\| \quad \forall \lambda \varepsilon R, \ \forall x \varepsilon R^n$;
(N3) $\|x+y\|\leq\|x\|+\|y\| \quad \forall x,y \varepsilon R^n$

(Bez.: Definitheit, Homogenität und Dreiecksungleichung sind erfüllt).

Beispiele für Normen

a) $\|x\|_2=|x|=\sqrt{x_1^2+\ldots+x_n^2}$: Euklidische Norm, 2-Norm

b) $\|x\|_1=\sum_{i=1}^{n}|x_i|$: Summennorm, 1-Norm

c) $\|x\|_\infty=\max_i|x_i|$: Maximum-Norm, ∞-Norm

d) Die drei Normen unter a)-c) sind enthalten in $\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p \right)^{\frac{1}{p}}$: Hölder-Norm, oder p-Norm $(1 \le p \le \infty)$.

e) $\|x\|_A = (x^T A x)^{\frac{1}{2}}$: Ellipsoid-Norm, (Vor.: A ist eine symmetrische positiv definite (n,n)-Matrix).

Der Unterschied zwischen den Normen wird deutlich, wenn man sich die Einheitskugeln der Normen $S = \{x \varepsilon R^n \mid \|x\| \le 1\}$ veranschaulicht (n=2):

$S_2 = \{ x : \|x\|_2 \le 1 \}$ $\quad x_1^{\,2} + x_2^{\,2} \le 1$

$S_\infty = \{ x : \|x\|_\infty \le 1 \}$ $\quad |x_i| \le 1$

$S_1 = \{ x : \|x\|_1 \le 1 \}$ $\quad |x_1| + |x_2| \le 1$

$S_A = \{ x : \|x\|_A \le 1 \}$

Weitere wichtige Norm-Ungleichung:

$$\|x - y\| \ge | \|x\| - \|y\| | \quad \forall x, y \varepsilon R^n$$

Beweis:

$$\|x\| = \|(x - y) + y\| \underset{(N3)}{\le} \|x - y\| + \|y\| \quad \Rightarrow \|x - y\| \ge \|x\| - \|y\|.$$

Vertausche x und y

$$\|y - x\| = \|x - y\| \ge \|y\| - \|x\|$$

Damit gilt die Behauptung.

Bemerkung: Eine Übertragung auf allgemeinere Vektorräume V, in denen ein Skalarprodukt <x,y> erklärt ist, erfolgt z.B. durch die Normdefinition $\|x\| = \sqrt{< x, x >}$. Ein Vektorraum mit Norm heißt normierter Raum.

Satz 2.2: Eine Norm ist eine gleichmäßig stetige Funktion.

Satz 2.3: Alle Normen im Raum R^n sind äquivalent in folgendem Sinn:
Für jedes Paar $p_1(x), p_2(x)$ von Normen gibt es positive Konstanten m und M mit $m \cdot p_2(x) \le p_1(x) \le M \cdot p_2(x), \quad \forall x \varepsilon R^n$.

Beweis: (Nur für $p_2(x) = \|x\|_\infty$.)

Die Normkugel $S_\infty = \{x \varepsilon R^n : \|x\|_\infty \le 1\}$ ist kompakt (beschränkt und abgeschlossen), d.h. das Maximum und Minimum einer stetigen Funktion auf S existiert.

Da $p_1(x)$ stetig ist, existieren die Werte

$$M = \max_{x \varepsilon S_\infty} p_1(x) \ , \ m = \min_{x \varepsilon S_\infty} p_1(x).$$

Damit gilt für alle $y \neq 0 : \frac{y}{\|y\|_\infty} \varepsilon S_\infty$ und $m \le p_1\left(\frac{y}{\|y\|_\infty}\right) = \underset{(N2)}{\frac{1}{\|y\|_\infty}} p_1(y) \le M$,

d.h. $m \|y\|_\infty \le p_1(y) \le M \|y\|_\infty$.

Bemerkung: Der Beweis für andere Normpaare erfolgt analog. In unendlichdimensionalen Funktionenräumen muss Satz 2.3 nicht gültig sein.

2.2. MATRIXNORMEN

Wir betrachten die durch eine Matrix $A \varepsilon R^{m x n}$ vermittelte lineare Abbildung y=Ax und fragen, um welchen Faktor sich $\|y\|_p = \|Ax\|_p \quad (y \in R^m)$ gegenüber $\|x\|_p$ maximal verändert, wenn $x \varepsilon R^n$ alle Vektoren des R^n durchläuft.

Def. 2.4: Jeder Vektornorm $\|\bullet\|_p$ lässt sich eine Matrixnorm zuordnen (zugehörige oder induzierte Matrixnorm, Operatornorm) durch

$$\|A\|_p := \max \left\{ \frac{\|Ax\|_p}{\|x\|_p} : \ x \neq 0 \right\} = \max \left\{ \|Az\|_p : \|z\|_p \le 1 \right\}.$$

Bemerkungen:

1) Man beachte, dass $\|x\|_p$ im R^n aber $\|Ax\|_p$ im R^m berechnet wird und nur für eine (n,n)-Matrix *A* im gleichen Raum.

2) Die Definition ist sinnvoll, da gilt

$$\frac{\|Ax\|_p}{\|x\|_p} = \left\| A \frac{x}{\|x\|_p} \right\|_p = \|Az\|_p \text{ mit } \|z\|_p = 1$$

und $S_p = \left\{ z \varepsilon R^n \,|\, \|z\|_p \le 1 \right\}$ ist kompakt, d.h. das Maximum existiert.

Allgemeiner definiert man:

Def.2.5: Eine Abbildung $\|\bullet\|: R^{m\times n} \to R$ heißt Matrixnorm, falls folgende Bedingungen erfüllt sind:

(M1) $\|A\| \geq 0 \quad \forall\ A \in R^{m\times n}$ und $\|A\| = 0 \Leftrightarrow A = O^{m\times n}$.

(M2) $\|\lambda A\| = |\lambda|\|A\|, \quad \forall \lambda \in R$;

(M3) $\|A + B\| \leq \|A\| + \|B\|$.

Eine Matrixnorm heißt submultiplikativ, falls

(M4) für m=n gilt $\|AB\| \leq \|A\| \bullet \|B\| \quad \forall\ A,B \in R^{n\times n}$.

Bem.: 1) Die induzierte Matrixnorm aus Def. 2.5 erfüllt die Axiome (M1)-(M4).

$$\|A\,B\| = \max_{x\neq 0} \frac{\|A\,B\,x\|}{\|x\|} = \max_{x\neq 0} \frac{\|A(\,B\,x)\|}{\|B\,x\|} \cdot \frac{\|B\,x\|}{\|x\|} \leq \max_{y\neq 0} \frac{\|A\,y\|}{\|y\|} \bullet \max_{x\neq 0} \frac{\|Bx\|}{\|x\|} = \|A\| \cdot \|B\|$$

2) Aus Def.2.4 folgt unmittelbar $\|A\ x\|_p \leq \|A\|_p \bullet \|x\|_p$.

Geometrische Deutung der Operatornorm

Wegen $\|A\|_p = \max\left\{\|Ax\|_p : \|x\|_p \leq 1\right\}$ stellt $\|A\|_p$ die kleinste Zahl μ dar, für die gilt

$\|Ax\|_p \leq \mu \|x\|_p \quad \forall\ x \in R^n$ bzw. $\|Az\|_p \leq \mu, \forall\ z \in R^n \quad \|z\|_p \leq 1$.

Damit ist μ die größte Abbildungsdehnung, die durch Ax vermittelt wird; d.h. μ ist der Faktor, um den sich $\|x\|$ maximal vergrößert beim Übergang zu $y = Ax$.

Beispiele für Matrixnormen: $A \in R^{m\times n}$

1.Operatornorm zur Vektornorm $\|x\|_1$:

$$\|Ax\|_1 = \sum_{i=1}^{m} |(A\,x\,)_i| = \sum_{i=1}^{m} |\sum_{k=1}^{n} a_{ik}\,x_k| \leq \sum_{i=1}^{m}\sum_{k=1}^{n} |\,a_{ik}\,||\,x_k\,|$$

$$= \sum_{i=1}^{n} (|\,x_k\,| \sum_{i=1}^{m} |\,a_{ik}\,|) \leq \left(\max_k \sum_{i=1}^{m} |\,a_{ik}\,|\right) \bullet \sum_{k=1}^{n} |\,x_k\,| = \left(\max_k \sum_{i=1}^{m} |\,a_{ik}\,|\right) \|x\|_1$$

d.h.

$$\frac{\|Ax\|_1}{\|x\|_1} \leq \max_k \sum_{i=1}^{m} |\,a_{ik}\,|.$$

Es sei nun $k = l$ der Index, für den das Maximum über die Spalten $k \in \{1,\ldots,n\}$ angenommen wird $\sum_{i=1}^{m} |a_{il}| = \max_k \sum_{i=1}^{m} |a_{ik}|$. Für $x = e^l$ gilt dann $\|x\|_1 = 1$ und

$$\|Ax\|_1 = \sum_{i=1}^{m} |\,a_{il} \cdot 1\,| = \max_k \sum_{i=1}^{m} |\,a_{ik}\,|.$$

Damit gilt:

$$\|A\|_1 = \max_{k\varepsilon\{1,\ldots,n\}} \sum_{i=1}^{m} |a_{ik}| \qquad \text{(Spaltensummennorm der Matrix A)}$$

2. Operatornorm zur Vektornorm $\|x\|_\infty$:

$$\|A\|_\infty = \max_{i\varepsilon\{1,\ldots,m\}} \sum_{j=1}^{n} |a_{ij}| \qquad \text{(Zeilensummennorm der Matrix A)}$$

3.Operatornorm zur euklidischen Vektornorm:

Diese Norm ist komplizierter zu bestimmen: Sei $A \varepsilon R^{m x n}$, dann ist $A^T A$ eine symmetrische (n,n)-Matrix, d.h. alle EW sind reell. Sei λ Eigenwert von $A^T A$ und $x \neq 0$ ein Eigenvektor, dann gilt

$$A^T A x = \lambda x, \quad x^T A^T A x = \lambda x^T x \Rightarrow \|Ax\|_2^2 = \lambda \|x\|_2^2, d.h. \lambda \geq 0.$$

Sei $0 \leq \lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_n = \lambda_{\max}$ das Spektrum der Eigenwerte von $A^T A$, so kann man zeigen:

$$\|A\|_2 = \sqrt{\lambda_{\max}(A^T A)},\ \lambda_{\max}\ \textit{größter EW von}\ A^T A \qquad \text{(Spektralnorm der Matrix A)}$$

Bemerkung: Ist A selbst eine symmetrische (n,n)-Matrix, so ist $A = A^T$, d.h. $A^T A = A^2$ und die Eigenwerte von A² sind $\lambda_i = \mu_i^2$,wenn μ_i die Eigenwerte von A sind. Es gilt dann $\|A\|_2 = \max|\mu_i| = \rho(A)$ ist der Spektralradius von *A* (siehe Def. 2.6.).

4. Frobenius-Norm

In Analogie zur Euklidischen Vektornorm kann man die Summe der Quadrate der Matrixelemente bilden und definiert:

$$\|A\|_F = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2} \qquad \text{(Frobenius-Norm der Matrix A)}$$

Die Frobeniusnorm ist keine Operatornorm im Sinne von Def. 2.4., da für die Einheitsmatrix gilt $\|E\|_F = \sqrt{n}$. Nach Def. 2.4. muss für eine Operatornorm wegen Ex=x aber stets $\|E\| = 1$ sein. Die vorwiegende Anwendung der Frobeniusnorm liegt in der Abschätzung der schwierig zu berechnenden Spektralnorm. Es gilt

$$\frac{\|A\|_F}{\sqrt{r}} \leq \|A\|_2 \leq \|A\|_F, \text{ wobei } r = rg(A) \leq \min\{m,n\} \text{ ist.}$$

Der rechte Teil der Ungleichung folgt daraus, dass $\|A\|_F^2 = Spur(A^T A) = \sum_{i=1}^{n} \lambda_i$ ist. Für alle $x \varepsilon R^n$ kann somit $\|Ax\|_2 \leq \|A\|_F \|x\|_2$ abgeschätzt werden.

2.3. Konvergente Matrizen

Konvergente Matrizen spielen eine zentrale Rolle für die Untersuchung von Iterationsverfahren (z. B. zur Lösung linearer Gleichungssysteme). Dieser Frage widmen wir uns im Kapitel 4.7. Zunächst werden einige Eigenschaften konvergenter Matrizen zusammengestellt.

Def. 2.6.: Sei A eine beliebige (n,n)-Matrix und $\mu_1,...,\mu_n \varepsilon C$ ihre (i.a. komplexen) Eigenwerte, dann nennt man

$$\rho(A) = \max_{i\varepsilon\{1,\dots,n\}} |\mu_i|$$

Spektralradius der Matrix A.

Satz 2.7.: Für jede (n,n)-Matrix gilt $\rho(A) = \inf_{\|\bullet\|} \|A\|$, d.h. der Spektralradius ist die größte untere Schranke für jede Matrixnorm von A.

Satz 2.8: Sei A eine (n,n)-Matrix und $\|\cdot\|$ eine beliebige Matrixnorm.
Dann sind die folgenden Aussagen äquivalent:

(a) $\lim_{k->\infty} A^k = O$, (b) $\lim_{k->\infty} \|A^k\| = 0$, (c) $\rho(A) < 1$.

Die Matrix A wird dann als konvergente Matrix bezeichnet.

Die Bedingung $\rho(A) < 1$ ist notwendig und hinreichend dafür, dass A eine konvergente Matrix ist. Da der Spektralradius $\rho(A)$ eine untere Schranke für jede Matrixnorm ist, muss diese Bedingung erfüllt sein, wenn für irgendeine Matrixnorm gilt $\|A\| < 1$. Dies stellt eine hinreichende Bedingung für eine konvergente Matrix dar (die aber nicht notwendig ist).

Bemerkung: Im Allgemeinen findet man keine Norm, für die $\rho(A) = \|A\|$ gilt. Bei symmetrischen Matrizen gilt allerdings $\|A\|_2 = \rho(A)$, d.h. die Spektralnorm stimmt mit dem Spektralradius überein. Die Spektralnorm ist für symmetrische Matrizen somit die kleinste induzierte Norm. Bei unsymmetrischen, diagonalisierbaren Matrizen kann man für jede Matrix eine Norm konstruieren, die dem Spektralradius entspricht.

Für konvergente Matrizen gilt ein wichtiger Zusammenhang zwischen der Reihe der Matrixpotenzen und einer inversen Matrix:

Satz 2.9.: Ist A eine konvergente (n,n)-Matrix, so ist

$$\sum_{k=0}^{\infty} A^k = E + A + A^2 + A^3 + \dots$$

eine konvergente Matrizenreihe und umgekehrt. Für die Reihensumme gilt

$$\sum_{k=0}^{\infty} A^k = (E - A)^{-1}.$$

3. Direkte Methoden zur Lösung linearer Gleichungssysteme

3.1. Elementare Matrixoperationen

In diesem Abschnitt werden einige Grundlagen der Matrizenrechnung zusammengestellt, die für die numerische Behandlung von linearen Gleichungssystemen $Ax = b$ von Bedeutung sind.

A - Skalierung einer Matrix

Anwendung einer Diagonalmatrix

$$D = diag(d_1,\ldots,d_n) = \begin{pmatrix} d_1 & 0 & \ldots & 0 \\ 0 & d_2 & \ldots & 0 \\ .. & .. & .. & .. \\ 0 & 0 & \ldots & d_n \end{pmatrix}, \; d_i \neq 0$$

auf Vektoren und Matrizen (Index bezeichnet das entsprechende Element) liefert:

$$(Dx)_j = d_j x_j; \quad \text{für } x = (x_1,\ldots,x_n)^T$$
$$(DA)_{ij} = d_j a_{ij}; \quad \text{für } A = (a_{ij})$$
$$(AD)_{ij} = a_{ij} d_j$$

Ähnlichkeitstransformation einer Matrix A mit D ergibt mit

$$D^{-1} = diag\left(\frac{1}{d_1},..,\frac{1}{d_n}\right)$$

$$\tilde{A} = D\,A\,D^{-1}, \text{ d.h. } \tilde{a}_{ij} = \frac{d_i}{d_j} a_{ij} \qquad (a_{ii} \text{ bleiben unverändert})$$

B - Vertauschung von Zeilen bzw. Spalten einer Matrix

P_{ij} heißt Permutationsmatrix: $p_{ii} = p_{jj} = 0, \quad p_{ij} = p_{ji} = 1$

$$P_{ij} = \begin{pmatrix} 1 & 0 & \ldots & 0 & 0 \\ 0 & 0 & & 1 & 0 \\ .. & .. & 1 & .. & .. \\ 0 & 1 & & 0 & 0 \\ 0 & 0 & \ldots & 0 & 1 \end{pmatrix} \begin{matrix} \\ i \\ \\ j \\ \\ \end{matrix}$$
$$\begin{matrix} & i & & j & \end{matrix}$$

Anwendung von P_{ij} auf Vektoren bzw. Matrizen ergibt

$P_{ij}\,x$: vertauscht x_i mit x_j

$P_{ij}A$: vertauscht Zeile i mit Zeile j

AP_{ij} : vertauscht Spalte i mit Spalte j

Weitere Eigenschaften: $P_{ij}^{-1} = P_{ij}, denn\ P_{ij} \bullet P_{ij} = E$, $\det(P_{ij}) = -1$

C - Reihenoperationen mit Elementarmatrizen

$$N_{ij}(\alpha) = \begin{pmatrix} 1 & 0 & .. & 0 & 0 \\ 0 & 1 & .. & 0 & 0 \\ .. & .. & 1 & .. & .. \\ 0 & \alpha & .. & 1 & 0 \\ 0 & 0 & .. & 0 & 1 \end{pmatrix} \begin{matrix} \\ \\ \\ i \\ \\ \\ j \end{matrix} \qquad (i \neq j)$$

Anwendung von N_{ij} auf Vektoren bzw. Matrizen ergibt:

$N_{ij}(\alpha)x$: $\alpha x_j + x_i = x_i$' Addiere α* Komponente j zur Komponente i

$N_{ij}(\alpha)A$: Addiere α* Zeile j zur Zeile i

$AN_{ij}(\alpha)$: Addiere α* Spalte j zur Spalte i

Weitere Eigenschaften:

1) $N_{ij}(\alpha)^{-1} = N_{ij}(-\alpha)$

2) $\det N_{ij}(\alpha) = 1$

3) $$N_{1j}(\alpha_1)N_{2j}(\alpha_2)....._{i\neq j}...N_{nj}(\alpha_n) = \prod_{\substack{i=1 \\ i\neq j}}^{n} N_{ij}(\alpha_i) = \begin{pmatrix} 1 & .. & \alpha_1 & .. & 0 \\ 0 & ... & \alpha_{j-1} & & 0 \\ & & 1 & & \\ 0 & & \alpha_{j+1} & & 0 \\ 0 & .. & \alpha_n & .. & 1 \end{pmatrix}$$

4) $$\prod_{i>j} N_{ij}(\alpha_{ij}) = N_{21}(\alpha_{21})N_{31}(\alpha_{31}).....N_{n1}(\alpha_{n1})$$
$$N_{32}(\alpha_{32}).....N_{n2}(\alpha_{n2}) \quad = \begin{pmatrix} 1 & .. & 0 & 0 \\ \alpha_{21} & 1 & .. & 0 \\ ... & & 1 & .. \\ \alpha_{n1} & & \alpha_{n,n-1} & 1 \end{pmatrix}$$
$$....N_{n,n-1}(\alpha_{n,n-1})$$

untere Dreiecksmatrix (normalisiert, Hauptdiagonalelemente sind 1).

3.2. Gauss-scher Eliminationsalgorithmus, LR-Faktorisierung einer Matrix

Wir betrachten den Regelfall eines linearen Gleichungssystems, d.h. n lineare Gleichungen für n Unbekannte

Ax=b

mit einer (n,n)-Matrix A; $x, b \varepsilon R^n$.

Prinzip der Gauß-Elimination:

1) Vorwärtsrechnung: Durch endlich viele Eliminationsschritte (n-1 Hauptschritte) wird das Gleichungssystem auf obere Dreiecksform gebracht.

$$(A:b) := (A^0 : b^0) \to (A^{(1)} : b^{(1)}) \to \ldots \to (A^{(n-1)} : b^{(n-1)}) = (R,c)$$

2) Rückwärtsrechnung: Das gestaffelte System Rx=c wird von unten aufgelöst nach $x_n, x_{n-1}, \ldots, x_1$

Vorwärtsrechnung:

1.Hauptschritt

$$\left(A^{(0)} \middle| b^{(0)}\right) = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \ldots & a_{1n} & b_1 \\ a_{21} & a_{22} & \ldots & a_{23} & b_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ a_{n1} & a_{n2} & \ldots & a_{nn} & b_n \end{array}\right) \begin{array}{c} -l_{21} \\ \downarrow \\ \\ \\ \end{array} \quad \begin{array}{c} -l_{31} \\ \\ \downarrow \\ \\ \end{array} \quad \begin{array}{c} \ldots \\ \\ \ldots \\ \\ \end{array} \quad \begin{array}{c} -l_{n1} \\ \\ \\ \downarrow \end{array}$$

Durch Zeilenumformungen mit Eliminationsfaktoren

$$l_{21} = \frac{a_{21}}{a_{11}}, \quad l_{31} = \frac{a_{31}}{a_{11}}, \ \ldots, \ l_{n1} = \frac{a_{n1}}{a_{11}}, \quad (Vor.: a_{11}^{(0)} \neq 0)$$

sollen in der 1. Spalte der umgerechneten Matrix Nullen entstehen:

$$\left(A^{(1)} \middle| b^{(1)}\right) = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \ldots & a_{1n} & b_1 \\ 0 & a^{(1)}{}_{22} & \ldots & a^{(1)}{}_{23} & b^{(1)}{}_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & a^{(1)}{}_{n2} & \ldots & a^{(1)}{}_{nn} & b^{(1)}{}_n \end{array}\right)$$

2.Hauptschritt

Das gleiche Verfahren auf das Unterschema ohne 1.Spalte und 1.Zeile anwenden $(Vor.: a_{22}^{(1)} \neq 0)$ usw.

Algorithmus:

Vorwärtsrechnung

$j = 1,.2,...,n-1 \quad (Hauptschritte)$

| $r_{jk} = a_{jk}^{(j-1)} \quad für \quad k = j,...,n \ (Pivotzeile)$

| $i = j+1,...,n$

| | $l_{ij} = a_{ij}^{(j-1)} / r_{jj}$

| | $k = j+1,...,n$

| | $a_{ik}^{(j)} = a_{ik}^{(j-1)} - l_{ij} r_{jk}$

Umformen der rechten Seite

$$\underline{j = 1,.2,...,n-1}$$
$$\left| \quad c_j = b_j^{(j-1)} \right.$$
$$\left| \quad \underline{i = j+1,...,n} \right.$$
$$\left| \quad \left| \quad b_i^{(j)} = b_i^{(j-1)} - l_{ij} c_j \right. \right.$$

Rückwärts-Auflösung:

$$\underline{i = n, n-1,...,1}$$
$$\left| \quad x_i = (c_i - \sum_{j=i+1}^{n} r_{ij} x_j) / r_{ii} \right.$$

Gauß-Algorithmus als Dreiecksfaktorisierung (LR-Faktorisierung) der Matrix

Die Hauptschritte können mit Elementarmatrizen beschrieben werden:

$N_{ik}(\lambda) \bullet A$ bedeutet: Addition von $\lambda \bullet$ k-te Zeile von A zur i-ten Zeile.

1.Hauptschritt

$$(A^{(1)} : b^{(1)}) = N_{n1}(-l_{n1}) \bullet \ldots \bullet N_{21}(-l_{21})(A : b) = G_1(A : b)$$

mit der Frobenius-Matrix

$$G_1 = \begin{pmatrix} 1 & 0 & ... & 0 \\ -l_{21} & 1 & ... & 0 \\ ... & ... & ... & ... \\ -l_{n1} & 0 & ... & 1 \end{pmatrix}$$

Wegen $N_{ik}(\lambda)^{-1} = N_{ik}(-\lambda)$ folgt

$$G_1^{-1} = N_{21}(l_{21}) \bullet ... \bullet N_{n1}(l_{n1}) = \begin{pmatrix} 1 & 0 & ... & 0 \\ l_{21} & 1 & ... & 0 \\ ... & ... & ... & ... \\ l_{n1} & 0 & ... & 1 \end{pmatrix}.$$

2.Hauptschritt

$$(A^{(2)} : b^{(2)}) = N_{n2}(-l_{n2}) \bullet \ldots \bullet N_{32}(-l_{32})(A : b) = G_2(A^{(1)} : b^{(1)})$$

mit

$$G_2 = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & -l_{32} & \dots & \dots \\ 0 & -l_{n2} & \dots & 1 \end{pmatrix} \quad G_2^{-1} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ 0 & l_{32} & \dots & \dots \\ 0 & l_{n2} & \dots & 1 \end{pmatrix}$$

usw.

Insgesamt erhält man nach (n-1) Hauptschritten

$$(R,c) = G_{n-1} \bullet \dots \bullet G_2 \bullet G_1 (A \vdots b)$$

Der Gaußalgorithmus liefert somit eine <u>Dreiecksfaktorisierung von A</u>

$$A = L \bullet R$$

mit

$$L = G_1^{-1} G_2^{-1} \dots G_{n-1}^{-1} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ l_{21} & 1 & \dots & 0 \\ l_{31} & l_{32} & 1 \dots & 0 \\ l_{n1} & l_{n2} & \dots l_{n,n-1} & 1 \end{pmatrix}$$

L...normalisierte untere Dreiecksmatrix

R...obere Dreiecksmatrix.

<u>Nebenprodukt:</u> Effektive Berechnung der Determinante von A

$$\det(A) = |L| \bullet |R| = 1 \bullet \prod_{i=1}^{n} r_{ii}$$

<u>Bsp.1:</u>

$$4x_1 - 9x_2 + 2x_3 = 2$$
$$2x_1 - 4x_2 + 4x_3 = 3$$
$$-x_1 + 2x_2 + 2x_3 = 1$$

Vorwärtsrechnung:

$$\begin{matrix} & A^{(0)} & & & b^{(0)} & & & A^{(1)} & & & b^{(1)} \end{matrix}$$

$$\begin{bmatrix} 4 & -9 & 2 & \vdots & 2 \\ 2 & -4 & 4 & \vdots & 3 \\ -1 & 2 & 2 & \vdots & 1 \end{bmatrix} \begin{matrix} \\ -l_{21} \\ \downarrow \end{matrix} \quad \begin{matrix} \\ -l_{31} \\ \downarrow \end{matrix} \quad \Rightarrow \quad \begin{bmatrix} 4 & -9 & 2 & \vdots & 2 \\ 0 & \frac{1}{2} & 3 & \vdots & 2 \\ 0 & -\frac{1}{4} & \frac{5}{2} & \vdots & \frac{3}{2} \end{bmatrix} \begin{matrix} \\ \\ -l_{32} \\ \downarrow \end{matrix}$$

$$l_{21} = \frac{a_{21}}{a_{11}} = \frac{1}{2} \quad l_{31} = \frac{a_{31}}{a_{11}} = -\frac{1}{4} \quad l_{32} = -\frac{a_{32}^{(1)}}{a_{22}^{(1)}} = -\frac{1}{2}$$

$$\Rightarrow \begin{bmatrix} 4 & -9 & 2 & \vdots & 2 \\ 0 & \frac{1}{2} & 3 & \vdots & 2 \\ 0 & 0 & 4 & \vdots & \frac{5}{2} \end{bmatrix} = (R \vdots c)$$

Rückwärtsrechnung: $x_3 = \frac{5}{8} = 0.625 \quad x_2 = 0.25 \quad x_1 = 0.75$

Dreiecksfaktorisierung von A:

$$A = \begin{pmatrix} 4 & -9 & 2 \\ 2 & -4 & 4 \\ -1 & 2 & 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ -\frac{1}{4} & -\frac{1}{2} & 1 \end{pmatrix} \begin{pmatrix} 4 & -9 & 2 \\ 0 & \frac{1}{2} & 3 \\ 0 & 0 & 4 \end{pmatrix} = L \bullet R$$

Effektive Speichertechnik der LR-Zerlegung: $\begin{bmatrix} 4 & -9 & 2 \\ \frac{1}{2} & \frac{1}{2} & 3 \\ -\frac{1}{4} & -\frac{1}{2} & 4 \end{bmatrix}$

d.h. die in L und R steckenden Informationen können "in-situ", d.h. auf den Platz von A abgespeichert werden, wenn A nicht weiter benötigt wird.

<u>Beschreibung des Gauß-Algorithmus mit Matrixfaktorisierung:</u>

Das System Ax=b ist äquivalent dem System
Ax=LRx=b.

Mit Rx=y kann der Algorithmus wie folgt beschrieben werden:

(1). Faktorisiere A in der Form LR.
(2) (a) Löse Ly=b mit unterer Dreiecksmatrix L; Ergebnis ist y.
(b) Löse Rx=y mit oberer Dreiecksmatrix R; Ergebnis ist x.

<u>Bemerkung:</u>
Mit der LR-Zerlegung ist die Lösung mehrerer Systeme mit gleicher Matrix und verschiedenen rechten Seiten besonders vorteilhaft, da die Dreieckszerlegung im Schritt (1) nur einmal ausgeführt wird, und die Schritte (2a) und (2b) wenig Aufwand erfordern.

<u>Beachte:</u> Für Durchführung des Algorithmus war $a_{11} \neq 0, a_{22}^{(1)} \neq 0, \ldots, a_{nn}^{(n-1)} \neq 0$ Voraussetzung. Nicht jede reguläre Matrix A besitzt jedoch eine solche direkte Dreieckszerlegung wie das folgende Beispiel zeigt.

<u>Bsp.:</u> $A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$

Obwohl $|A| = -1 \neq 0$, gilt $a_{11} = 0$, d.h. es existiert keine direkte LR-Faktorisierung. Vertauscht man jedoch die Zeilen (Mult. von A mit der Permutationsmatrix $P = P_{12}$), so gilt

$$PA = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = LR$$ mit L=E, R=E.

Wir betrachten die numerische Stabilität der Gauß-Elimination: Es gilt der Satz:

Satz 3.1: (Fehlerabschätzung, Rückwärtsanalyse der Rundungsfehler)
Der Gauß-Algorithmus berechnet eine Näherungslösung $\tilde{x}$ mit $\tilde{A}\tilde{x} = b$ und $|\tilde{A} - A| \leq (3\varepsilon_0 + \varepsilon_0^{\,2})|\tilde{L}| \bullet |\tilde{R}|$ (elementweise) und

$\varepsilon_0 = \dfrac{n\ eps}{1 - n\ eps}$, $\tilde{L}, \tilde{R}$ berechnete Dreieckszerlegung.

Bem.: Werden die Elemente der Matrizen $\tilde{L}, \tilde{R}$ betragsmäßig sehr groß, so können große numerische Fehler in der Lösung auftreten. Eine Vertauschung von Zeilen in der Ausgangsmatrix kann starken Einfluss auf die Elemente von $\tilde{L}$ *und* $\tilde{R}$ und damit auf die Genauigkeit der Näherungslösung haben.

3.3. Numerische Instabilitäten der Gauß-Elimination, Pivotisierung

Bei der Durchführung des Gauß-Algorithmus können numerische Instabilitäten von katastrophaler Auswirkung auftreten, wenn keine Modifikationen vorgenommen werden. Um dies zu verdeutlichen, analysieren wir ein einfaches Beispiel:

Bsp.1:
$$\begin{aligned} -10^{-5} x_1 + x_2 &= 1 \qquad -l_{21} = -\frac{a_{21}}{a_{11}} \\ 2 x_1 + x_2 &= 0 \qquad\quad \downarrow \end{aligned} \tag{3.1}$$

Exakte Lösung (auf 6 Stellen)
$$x_1 = -0.499\,998 \quad x_2 = 0.999\,995 \tag{3.2}$$

(a) Wir nehmen an, dass das System mit einem hypothetischen Computer mit dezimaler Arithmetik und vier Mantissenstellen gelöst wird. Eine Zahl wird in der Form $0.{*}{*}{*}{*} \bullet 10^p$ dargestellt. Wir wenden den Gauß-Algorithmus an:

$$\begin{aligned} l_{21} &= \frac{a_{21}}{a_{11}} = \frac{0.2 \bullet 10^1}{-0.1 \bullet 10^{-4}} = -0.2 \bullet 10^6 \\ a_{22}^{(1)} &= a_{22} - l_{21} a_{12} = 0.1 \bullet 10^1 - (-0.2 \bullet 10^6)(0.1 \bullet 10^1) = 0.1 \bullet 10^1 + 0.2 \bullet 10^6 \\ a_{22}^{(1)} &\cong 0.2 \bullet 10^6 \quad (exakt: 0.200\,001 \bullet 10^6) \\ b_2^{(1)} &= b_2 - l_{21} b_1 = 0 - (-0.2 \bullet 10^6)(0.1 \bullet 10^1) = 0.2 \bullet 10^6 \end{aligned}$$

Auflösung des gestaffelten Gleichungssystems liefert

$$\begin{aligned} x_2 &= \frac{b_2^{(1)}}{a_{22}^{(1)}} = \frac{0.2 \bullet 10^6}{0.2 \bullet 10^6} = 1.0 \\ x_1 &= \frac{b_1 - a_{12} x_2}{a_{11}} = \frac{0.1 \bullet 10^1 - 0.1 \bullet 10^1}{-0.1 \bullet 10^{-4}} = 0.0 \end{aligned} \tag{3.3}$$

Diese katastrophal schlechte Lösung für x_1 entstand durch einen einzigen Rundungsfehler bei der Berechnung von $a_{22}^{(1)}$.

(b) Werden dagegen vorher die Gleichungen ausgetauscht

$$2\,x_1 + x_2 = 0$$
$$10^{-5}\,x_1 + x_2 = 1$$

so erhält man

$$l_{21} = \frac{a_{21}}{a_{11}} = \frac{-0.1 \bullet 10^{-4}}{0.2 \bullet 10^{1}} = -0.5 \bullet 10^{-5}$$
$$a_{22}^{(1)} = a_{22} - l_{21}a_{12} = 0.1 \bullet 10^{1} - (-0.5 \bullet 10^{-5})(0.1 \bullet 10^{1}) \cong 0.1 \bullet 10^{1}$$
$$(exakt: 0.100\,0005 \bullet 10^{1})$$
$$b_2^{(1)} = b_2 - l_{21}b_1 = 0.1 \bullet 10^{1} - (0.5 \bullet 10^{-5}) \bullet 0 = 0.1 \bullet 10^{1}$$

Auflösung des gestaffelten Systems ergibt

$$x_2 = \frac{b_2^{(1)}}{a_{22}^{(1)}} = \frac{0.1 \bullet 10^{1}}{0.1 \bullet 10^{1}} = 1.0$$
$$x_1 = \frac{b_1 - a_{12}x_2}{a_{11}} = \frac{0 - (0.1 \bullet 10^{1}) \bullet 1}{0.2 \bullet 10^{1}} = -0.5 \qquad (3.4)$$

Diese numerische Lösung stellt im Rahmen der Genauigkeit der Arithmetik die tatsächliche Lösung exakt dar. Wie ist dieses Verhalten zu begründen und wie kann eine Strategie entwickelt werden, um die Lösung mit hoher Genauigkeit zu bestimmen?

Die Strategie zur Bestimmung der Lösung mit hoher Genauigkeit beruht auf der Beschränkung der Multiplikatoren l_{ij} und wird Pivotisierung genannt. Sie ist entscheidend für die numerische Stabilität des Verfahrens. Die Pivotisierung sichert, dass $|l_{ij}| \leq 1$ gilt.

Spaltenpivotisierung:

Die umgerechnete Matrix vor dem j-ten Hauptschritt lautet:

$$\begin{pmatrix}
a_{11}^{(0)} & x & x & x & .. & .. & x \\
0 & a_{22}^{(1)} & x & x & .. & .. & x \\
.. & & ... & & & & \\
0 & 0 & ... & a_{j-1,j-1}^{(j-2)} & x & .. & x \\
 & & & 0 & a_{j,j}^{(j-1)} & x \;\; ... \;\; x & \\
 & & & & a_{j+1,j}^{(j-1)} & x \;\; ... \;\; x & \Leftarrow zu\ bearbeitende\ \mathrm{Re}stmatrix \\
 & & & & ... & & \\
0 & 0 & & 0 & a_{n,j}^{(j-1)} & x & x
\end{pmatrix}$$

Pivotisierungsstrategie:

- Man suche in der ersten Spalte der Restmatrix das betragsgrößte Element (Pivot)

$$|a_{rj}^{(j-1)}| = \max_{i \varepsilon \{j,..n\}} |a_{ij}^{(j-1)}|$$

- Man vertausche die j-te und r-te Zeile und bringe somit a_{rj} in Position (j,j). Dann führe man den Hauptschritt durch.

Folgerungen:

1) Falls A nichtsingulär ist, so ist der Gauß-Algorithmus mit Spaltenpivotsuche bis zum Ende durchführbar. Denn enthielte die Restmatrix in der 1.Spalte nur Nullen, dann wäre $\det(A) = a_{11} \bullet a_{22}^{(1)} \bullet \ldots \bullet a_{j-1,j-1}^{(j-2)} \bullet \det(Restmatrix) = 0$.
2) Es gilt $|l_{ij}| \leq 1$, d.h. die Elemente von L können keine großen numerischen Fehler verursachen.

Algorithmische Durchführung: Es wird zusätzlich ein Integerfeld p der Länge (n-1) benötigt, um die Zeilenvertauschungen rekonstruieren zu können. Im j-ten Hauptschritt wird p(j)=r (Pivotindex) gesetzt $(j = 1,\ldots,n-1)$.

Statt der Dreieckszerlegung von A erhält man die Dreieckszerlegung von PA=LR, wobei die Permutationsmatrix P alle Zeilenvertauschungen repräsentiert und durch das Integerfeld p dargestellt wird.

Es gilt dann PAx=LRx=Pb, d.h. bei der Lösung des Systems auf der Basis der Dreieckszerlegung muss zunächst der Vektor Pb bestimmt werden. Die rechten Seiten b_i sind entsprechend der Zeilenvertauschungen umzuordnen (falls b nicht als (n+1)-te Spalte der Matrix bei der Vorwärtsrechnung mitgeführt wird).

Algorithmus (Berechnung von b:=Pb)

$j = 1,.2,...,n-1$

| *wenn* $p(j) > j$, *dann setze*
| $z = b_j$
| $b_j = b_{p(j)}$
| $b_{p(j)} = z$

Satz 3.2: Ist A nichtsingulär und $P = P_{n-1} P_{n-2} \ldots P_1$ die Matrix aller Zeilenpermutationen, so gilt in exakter Arthmetik PA=LR mit unterer normalisierter Dreiecksmatrix L und oberer Dreiecksmatrix R.

Bsp.2: Man löse das System

$$\begin{pmatrix} 3 & 1 & 6 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 2 \\ 7 \\ 4 \end{pmatrix}$$

mittels Gauß-Elimination und Spaltenpivotisierung und bestimme die Dreiecksfaktorisierung.

Vorwärtsrechnung:

1. Hauptschritt:

$$\begin{bmatrix} 3 & 1 & 6 & \vdots & 2 \\ 2 & 1 & 3 & \vdots & 7 \\ 1 & 1 & 1 & \vdots & 4 \end{bmatrix} \begin{matrix} \left(-\frac{2}{3}\right) \\ \downarrow \\ \end{matrix} \quad \begin{matrix} \\ \left(-\frac{1}{3}\right) \\ \downarrow \end{matrix} \Rightarrow \begin{bmatrix} 3 & 1 & 6 & \vdots & 2 \\ \frac{2}{3} & \frac{1}{3} & -1 & \vdots & \frac{17}{3} \\ \frac{1}{3} & \frac{2}{3} & -1 & \vdots & \frac{10}{3} \end{bmatrix}$$

2. Hauptschritt:

$$\begin{bmatrix} 3 & 1 & 6 & \vdots & 2 \\ \frac{1}{3} & \frac{2}{3} & 1 & \vdots & \frac{10}{3} \\ \frac{2}{3} & \frac{1}{3} & 1 & \vdots & \frac{17}{3} \end{bmatrix} \begin{matrix} \\ \left(-\frac{1}{2}\right) \\ \downarrow \end{matrix} \Rightarrow \begin{bmatrix} 3 & 1 & 6 & \vdots & 2 \\ \frac{1}{3} & \frac{2}{3} & -1 & \vdots & \frac{10}{3} \\ \frac{2}{3} & \frac{1}{2} & -\frac{1}{2} & \vdots & 4 \end{bmatrix}$$

Rückwärtsrechnung:

Lösung des gestaffelten Systems Rx=c:

$$\begin{aligned} 3x_1 + x_2 + 6x_3 &= 2 \\ \tfrac{2}{3}x_2 - x_3 &= \tfrac{10}{3} \\ -\tfrac{1}{2}x_3 &= 4 \end{aligned}$$

$$Lösung: \quad x_3 = -8, x_2 = -7, x_1 = 19$$

Dreieckszerlegung PA=LR

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}\begin{pmatrix} 3 & 1 & 6 \\ 2 & 1 & 3 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 3 & 1 & 6 \\ 1 & 1 & 1 \\ 2 & 1 & 3 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ \frac{1}{3} & 1 & 0 \\ \frac{2}{3} & \frac{1}{2} & 0 \end{pmatrix}\begin{pmatrix} 3 & 1 & 6 \\ 0 & \frac{2}{3} & -1 \\ 0 & 0 & -\frac{1}{2} \end{pmatrix}$$

Pivotindizes: p(1)=1, p(2)=3

Spaltenpivotisierung kann nicht in jedem Fall numerische Schwierigkeiten beheben. Wir betrachten Beispiel 1:

$$\begin{aligned} -10^{-5}x_1 + x_2 &= 1 \\ 2x_1 + x_2 &= 0 \end{aligned}$$

und multiplizieren die erste Gleichung mit dem Faktor 10^6, d.h.

$$\begin{aligned} -10x_1 + 10^6 x_2 &= 10^6 \\ 2x_1 + x_2 &= 0 \end{aligned}$$

Mittels Spaltenpivotisierung erhält man unter Anwendung der hypothetischen vierstelligen dezimalen Arithmetik auf das so erhaltene System

$$l_{21} = \frac{a_{21}}{a_{11}} = -\frac{2}{10} = -0.2 \bullet 10^0$$

$$a_{22}^{(1)} = a_{22} - l_{21}a_{12} = 1 + 0.2 \bullet 10^6 \cong 0.2 \bullet 10^6$$

$$b_2^{(1)} = b_2 - l_{21}b_1 = 0 + 0.2 \bullet 10^6 = 0.2 \bullet 10^6$$

Damit erhalten wir mit Spaltenpivotisierung das gleiche numerisch schlechte Ergebnis $x_2 = 1, x_1 = 0.$

Skalierung:

Das Problem der numerischen Gutartigkeit der Gauß-Elimination ist nur durch Pivotisierung und Skalierung der Matrix zu lösen. Dabei wird A durch die Matrix

$$\bar{A} = D_1 A D_2$$

ersetzt; D_1, D_2 sind Diagonalmatrizen. Die Lösung von Ax=b erhält man durch Lösung des Systems $\bar{A} y = \bar{b} = D_1 b$ und $x = D_2 y$.

D_1, D_2 sind so zu wählen, dass für $\bar{A}$ näherungsweise gilt

$$\sum_{k=1}^{n} |\bar{a}_{ik}| \approx \sum_{j=1}^{n} |\bar{a}_{jl}| \; \forall \, i,l = 1,\ldots,n \quad ,$$

d.h. die Summe der Beträge in allen Zeilen und Spalten soll etwa gleich groß sein. Eine solche Matrix $\bar{A}$ heißt skaliert. Die Bestimmung von D_1, D_2 erfordert einen hohen Aufwand. Man ersetzt dies häufig durch eine einfachere Strategie:

$$D_1 = diag(s_1,\ldots,s_n); D_2 = E$$

$$s_i = \frac{1}{\sum_{k=1}^{n} |a_{ik}|}, (\textit{ Reziproke Zeilensumme})$$

Dann gilt $\sum_{k=1}^{n} |\bar{a}_{ik}| = 1; \; k = 1,2,\ldots,n$ (Matrix ist zeilenskaliert).

Die explizite Berechnung der Matrix $\bar{A}$ und die damit verbundenen Rundungsfehler werden dadurch umgangen, dass bei Anwendung des Algorithmus auf Ax=b mit Spaltenpivotsuche die folgende Regel Anwendung findet:

Modifizierte Pivotregel (j = 1, 2, ..n-1):

$$\textit{Bestimme den Pivotindex} \quad r \geq j \quad \textit{so, dass im } j\textit{-ten Schritt gilt}$$

$$|a_{r,j}^{(j-1)}| s_r = \max_{i \geq j} |a_{i,j}^{(j-1)}| s_i \neq 0.$$

Bei Anwendung auf obiges Beispiel erhält man für j=1:

$$s_1 = \frac{1}{(10 + 10^6)} = 0.1 \bullet 10^{-5}$$

$$s_2 = \frac{1}{(2+1)} = 0.3333$$

Wegen $|a_{11}| s_1 = 0.1 \bullet 10^{-4}$, $|a_{21}| s_2 = 0.6666$ wird a_{21} als Pivot gewählt, und man erhält die im Rahmen der Arithmetik bestmögliche numerische Lösung $x_1 = -0.5, x_2 = 1.0$.

3.4. Iterative Verbesserung einer Näherungslösung

x^* bezeichne die exakte Lösung von Ax=b;
$\tilde{x}$ sei eine Näherungslösung (z.B. durch Gauß-Elimination gewonnen)

Verbesserung von $\tilde{x}$

1. $r := A\tilde{x} - b$, "Residuum" berechnen
2. $\Delta x \text{ aus } A\Delta x = -r$ unter Verwendung der LR-Zerlegung von A bestimmen, d.h. Lösung von
   $$P A \Delta x = L R \Delta x = -P r .$$
3. $x' = \tilde{x} + \Delta x$.

Begründung:

$$x' = \tilde{x} + \Delta x = \tilde{x} + A^{-1}(-r) = \tilde{x} + A^{-1}(b - A\tilde{x}) = \tilde{x} + x^* - \tilde{x} = x^*$$

Praxis: Rundungsfehler bewirken, dass $x' \neq x^*$ ist, aber x' ist i.a. eine bessere Näherung als $\tilde{x}$. Die Verbesserung kann wiederholt werden.

Algorithmische Durchführung:
Wenn $\tilde{x}$ durch den Gauß-Algorithmus gewonnen wurde, dann erfüllt $\tilde{x}$ das Gleichungssystem meist sehr gut, d.h. $b \approx A\tilde{x}$. Dann tritt bei der Berechnung des Residuenvektors $r = A\tilde{x} - b$ aber Auslöschung auf.

Damit: Der erste Schritt sollte mit erhöhter Genauigkeit ausgeführt werden, sonst bleibt die Nachiteration wirkungslos. Rundungsfehler bei der Berechnung der LR-Zerlegung bzw. im 3. Schritt begrenzen die erreichbare Genauigkeit.

3.5. Numerischer Aufwand, spezielle Anwendungen

1. Numerischer Aufwand:
Wir betrachten den Algorithmus ohne Pivotisierung (die Operationen bei der Pivotisierung verändern die Ordnung des Algorithmus nicht wesentlich) in folgender Form:

a) Vorwärtsrechnung:

$$\begin{array}{l} \underline{\text{für } j = 1, 2, \ldots, n-1} \\ \quad \underline{i = j+1, \ldots, n} \\ \quad\quad l_{ij} = a_{ij} / a_{jj} \\ \quad\quad \underline{k = j+1, \ldots, n} \\ \quad\quad\quad a_{ik}^{(j)} = a_{ik}^{(j-1)} - l_{ij} a_{jk} \\ \quad\quad\quad b_i = b_i - l_{ij} b_i \end{array} \qquad (3.5)$$

b) Rückwärtsrechnung: $\underline{für\ j = n, n-1,...,1}$

$$| \quad x_i := \frac{1}{a_{ii}} \left(b_i - \sum_{j=i+1}^{n} a_{ij}\ x_j \right) \tag{3.6}$$

Operationen in der linearen Algebra sind von der Form a+b*c: Wir fassen 1 Addition und 1 Multiplikation zu 1 Operation zusammen und nehmen den Aufwand für eine Division als vergleichbar an.

Anzahl der Operationen

Operationen der der Vorwärtsrechnung:

$$\sum_{j=1}^{n-1} \sum_{i=j+1}^{n} \sum_{k=j+1}^{n} 1 = \sum_{j=1}^{n-1} \sum_{i=j+1}^{n} (n - j) = \sum_{j=1}^{n-1} [(n - j) + \ldots + (n - j)] = \tag{3.7}$$

$$\sum_{j=1}^{n-1} (n - j)^2 = [(n - j)^2 + (n - 2)^2 + \ldots + 2^2 + 1^2] = \frac{n(n-1)(2n-1)}{6} \approx \frac{n^3}{3}$$

Hinzu kommen Operationen zur Umrechnung der b_i, zur Bestimmung der l_{ik} und zur Rückwärtsrechnung, deren Anzahl in der Größenordnung const*n² liegt, wir bezeichnen dies mit O(n²).

Satz 3.3: Für große n ist der Aufwand zur Dreiecksfaktorisierung der Matrix A entscheidend und proportional zu $\frac{n^3}{3}$.

2. Numerische Berechnung von Determinanten:

Wegen PA=LR gilt

$$| P || A | = | L | \bullet | R | = \prod_{i=1}^{n} r_{ii} .$$

Satz 3.4: Ist PA=LR die Dreieckszerlegung der Matrix A, so gilt

$$\det(A) = (-1)^Z \prod_{i=1}^{n} r_{ii} \quad , \tag{3.8}$$

wobei Z die Anzahl der durch P bewirkten Zeilenpermutationen ist.

3. Numerische Berechnung der inversen Matrix:

Die Berechnung der inversen Matrix kann in vielen Fällen vermieden werden. Wenn benötigt, stellt auch hier der Gauß-Algorithmus eine effektive Methode dar. Wegen der Gültigkeit von

$$A^{-1} A = E \tag{3.9}$$

folgt, dass die Spaltenvektoren $x^i (i = 1,\ldots,n)$ der inversen Matrix

$$A^{-1} = (x^1, x^2, \ldots, x^n) \tag{3.10}$$

Lösung von n Gleichungssystemen sind

$$A\,x = e^i \quad i = 1,\ldots,n \tag{3.11}$$

Damit ist zur Lösung der n Systeme der Form (3.11) einmal die Berechnung der LR-Faktorisierung von A erforderlich und n-mal die Durchführung der Rückwärtsrechnung.

> **Satz 3.5**: Der Aufwand für die Berechnung der inversen Matrix liegt in der Größenordnung von n^3 Operationen und ist somit für große Matrizen ca. 3mal höher als die Lösung eines einzigen Systems Ax=b.

Aus diesem Grunde ist die Berechnung von x in der Form $x = A^{-1} b$ uneffektiv, solange nicht mehrere Systeme mit der gleichen Koeffizientenmatrix und verschiedenen rechten Seiten zu lösen sind.

4. Lösung von tridiagonalen Systemen

Systeme spezieller Struktur treten häufig bei bestimmten Anwendungen auf. So entstehen bei der numerischen Lösung von Differenzialgleichungen lineare Gleichungssysteme, welche nur mehrere besetzte Diagonalen in der Koeffizientenmatrix besitzen. Wir betrachten den häufigen Fall eines tridiagonalen Systems:

$$\begin{aligned}
a_{11}x_1 + a_{12}x_2 &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 &= b_2 \\
a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3 \\
&\vdots \\
a_{n,n-1}x_{n-1} + a_{nn}x_n &= b_n
\end{aligned} \qquad (3.12)$$

Wenn wir voraussetzen, dass a_{11} und alle weiteren Divisoren ungleich Null sind, so ergibt die Anwendung des Gauß-Algorithmus, dass in der ersten Spalte nur a_{21} annulliert werden muss. Dabei ändern sich nur die Werte von a_{22} *und* b_2. Das im Ergebnis des 1. Schritts erhaltene System hat wieder Tridiagonalgestalt, somit sind in der Vorwärtsrechnung insgesamt (n-1) Schritte erforderlich.

Aufwand: 2(n-1) Add.; 2(n-1) Multiplikationen; n-1 Divisionen

Die Rückwärtsrechnung erfordert die Lösung von (n-1) Gleichungen der Form

$$a_{ii}\ x_i + a_{i,i+1}\, x_{i+1} = b_i \quad (i = 1, 2, ..., n-1),$$

die letzte Gleichung hat die Form

$$a_{nn}\, x_n = b_n$$

Aufwand: (n-1) Add.; (n-1) Multiplikationen; n Divisionen

Gesamtaufwand: 3(n-1) Add.; 3(n-1) Multiplikationen; (2n-1) Divisionen $\approx$ 5n Op.

> **Satz 3.6**: Der numerische Aufwand bei Systemen mit diagonaler Bandstruktur steigt nur linear mit n.

Es können damit auch Systeme mit sehr vielen Variablen, wie sie bei der Diskretisierung von Differenzialgleichungen auftreten, noch effektiv gelöst werden.

3.6. CHOLESKY-FAKTORISIERUNGEN SYMMETRISCHER MATRIZEN

Während für allgemeine Systeme die Pivotisierung und die damit verbundene Vertauschung von Zeilen der Matrix notwendig für die Sicherung der numerischen Stabilität sind, gibt es Klassen von Problemen, bei denen man ohne Pivotisierung arbeiten kann.

Neben der Klasse von Problemen mit diagonal dominanten Matrizen, d.h. es gilt

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j\neq i}}^{n} |a_{ij}| \quad i = 1,\ldots,n$$

(betragsgrößte Elemente in Hauptdiagonale) betrifft dies Probleme mit symmetrischen, positiv definiten Matrizen. Derartige Probleme treten häufig bei der Modellierung technischer Anwendungen und der numerischen Lösung von Differenzialgleichungen auf.

Def. 3.7: Die symmetrische Matrix $A \varepsilon R^{n\,x\,n}$ heißt positiv definit, wenn gilt $x^T A x > 0 \quad \forall\, x \varepsilon R^n, x \neq 0$.

Satz 3.8: Es sei A symmetrisch, positiv definit. Dann gelten die Aussagen

(1) $a_{ii} > 0 \quad i = 1,2,\ldots,n$

(2) $|a_{ij}| < \frac{1}{2}(a_{ii} + a_{jj})$

(3) $|a_{ij}| < \sqrt{a_{ii}\, a_{jj}}$

(4) Bei der Gauß-Elimination ohne Pivotisierung ist jede Restmatrix wieder symmetrisch, positiv definit.

Folgerung: Die natürlichen Pivots $a_{jj}^{(j-1)}$ bei der Gauß-Elimination sind positiv und genügend groß, so dass der Algorithmus ohne Vertauschungen durchführbar ist. Man kann zeigen, dass auch unter dem Aspekt der Rundungsfehleranalyse Pivot-Suche unnötig ist.

Dreieckszerlegung: In der Dreiecksfaktorisierung der Form A=LR wird die Symmetrie von A nicht genutzt:

$$L = \begin{pmatrix} 1 & 0 & \ldots & 0 \\ * & 1 & & 0 \\ & & \ldots & \\ * & * & \ldots & 1 \end{pmatrix} \qquad R = \begin{pmatrix} r_{11} & * & \ldots & * \\ 0 & r_{22} & & * \\ & & \ldots & \\ 0 & 0 & \ldots & r_{nn} \end{pmatrix}$$

Wegen Satz 3.8 gilt jedoch $r_{jj} = a_{jj}^{(j-1)} > 0$. Es lässt sich die Matrix R in der Form $D\,\hat{R}$ darstellen, d.h. es gilt

$$A = L \bullet D \bullet \hat{R} \;\; mit \;\; D = diag\,(r_{11},\ldots,r_{nn}) \qquad \hat{R} = \begin{pmatrix} 1 & * & \ldots & * \\ 0 & 1 & & * \\ & & \ldots & \\ 0 & 0 & \ldots & 1 \end{pmatrix}, \;\; .$$

Wegen der Symmetrie von A folgt

$$A = L\,R = A^T = \hat{R}^T (D\,L^T),$$

und wegen der Eindeutigkeit der Dreieckszerlegung gilt $L = \hat{R}^T, \hat{R} = L^T$, d.h. $A = L\,D\,L^T$. Eine vollsymmetrische Dreieckszerlegung erhält man, indem

$$\overline{D} = diag\left(\sqrt{r_{11}}, \sqrt{r_{22}}, \ldots, \sqrt{r_{nn}}\right),\ \overline{D}\,\overline{D} = D$$

eingeführt wird. Dann gilt $A = L\,\overline{D} \bullet \overline{D}\,L^T = \overline{L} \bullet \overline{L}^T$ *mit* $\overline{L} = L\,\overline{D}$.

Satz 3.9: Eine symmetrische, positiv definite Matrix A besitzt die Faktorisierungen

$$A = \begin{cases} \overline{L}\,\overline{L}^T & (gewöhnliche\ \ Cholesky\text{-}Zerlegung) \\ LDL^T\ mit\ D = diag(r_{11}, \ldots, r_{nn}) & (rationale\ Cholesky\text{-}Zerlegung) \end{cases}$$

Die Matrix $\overline{L}$ ist nicht normalisiert, zur Berechnung sind n Wurzeloperationen nötig, dagegen ist L normalisiert und es sind keine Wurzeloperationen nötig. Damit ist die rationale Cholesky-Zerlegung auch für allgemeine symmetrische Probleme anwendbar. Man beachte, dass bei indefiniten Matrizen numerische Instabilitäten auftreten können, so dass die Cholesky-Faktorisierungen nur für symmetrische, positiv definite Matrizen geeignet sind.

Berechnung der Elemente l_{ij} von $\overline{L}$:

Beachte $l_{ij} = 0$ für j>i

$$\begin{pmatrix} a_{11} & \ldots & a_{1n} \\ & a_{ii} & \\ a_{n1} & \ldots & a_{nn} \end{pmatrix} = \begin{pmatrix} l_{11} & 0 & \ldots & 0 \\ l_{21} & l_{22} & & 0 \\ & & \ldots & \\ l_{n1} & l_{n2} & \ldots & l_{nn} \end{pmatrix} \bullet \begin{pmatrix} l_{11} & l_{21} & \ldots & l_{n1} \\ 0 & l_{22} & & l_{n2} \\ & & \ldots & \\ 0 & 0 & \ldots & l_{nn} \end{pmatrix}$$

Dann gelten die Beziehungen:

$$a_{11} = l_{11}^{\ 2},\ a_{i1} = l_{i1} \bullet l_{11} \Rightarrow l_{11} = \sqrt{a_{11}},\ \ l_{i1} = a_{i1}/l_{11} \quad i = 2,3,\ldots,n$$
$$a_{22} = l_{21}^{\ 2} + l_{22}^{\ 2} \Rightarrow l_{22} = \sqrt{a_{22} - l_{21}^{\ 2}}$$
$$a_{j2} = l_{21} \bullet l_{j1} + l_{22} \bullet l_{j2} \Rightarrow l_{j2} = (a_{j2} - l_{21} \bullet l_{j1})/l_{22} \quad j = 3,4,\ldots,n \quad \text{usw.}$$

Algorithmus der gewöhnlichen Cholesky-Faktorisierung: $A = \overline{L}\,\overline{L}^T$

$j = 1,.2,...,n$

$$s = a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2, \quad wenn\ s \le eps\ stop$$
$$l_{jj} = \sqrt{s}$$

$i = j+1,...,n$

$$l_{ij} = (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk}) / l_{jj}$$

Der Algorithmus wird abgebrochen, wenn der Radikant s< eps ist, wobei eps eine vorgegebene positive Schranke ist.. (Im Fall s<0 ist A entweder nicht positiv definit oder singulär.) Analog kann der Algorithmus für die rationale Cholesky-Zerlegung $A = L\,D\,L^T$ entwickelt werden.

Algorithmus der rationalen Cholesky-Faktorisierung: $A = L\,D\,L^T$

$j = 1,.2,...,n$

$$d_j = a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 d_k, \quad wenn\ d_j \le eps\ stop$$

$i = j+1,...,n$

$$l_{ij} = (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} d_k) / d_j$$

Speicherung: Effektive Speicherung von A und L im oberen und unteren Dreieck des Arbeitsfeldes unter Verwendung eines zusätzlichen Feldes der Länge n für die Hauptdiagonale von L bzw. D.

Analyse der Operationszahl für $A = \bar{L}\,\bar{L}^T$:
Wurzel: n Berechnungen;
weitere Operationen:

$$\sum_{j=1}^{n}\sum_{i=j}^{n}(j-1) = \sum_{j=1}^{n}(j-1)\bullet(n-j+1) = n\sum_{j=1}^{n}(j-1) - \sum_{j=1}^{n}(j-1)^2 =$$

$$= n\frac{n(n-1)}{2} \approx \frac{n^3}{6} op$$

Divisionen und Lösung der Dreieckssysteme: O(n²). Damit benötigt die Cholesky-Zerlegung nur etwa die Hälfte der Operationen des gewöhnlichen Gaußalgorithmus

Numerische Stabilität: Aus $a_{ii} = l_{i1}^{\ 2} + l_{i2}^{\ 2} + \ldots + l_{ii}^{\ 2} (i = 1,\ldots,n)$
folgt $|l_{ik}| \le \sqrt{a_{ii}}\quad k = 1,\ldots,i.$. Die Elemente von L können somit nicht beliebig groß werden, d.h. die Rundungsfehler bleiben beschränkt.

3.7. Fehlerabschätzung beim Gauss-Algorithmus

3.7.1. Störungen der rechten Seite des Systems

Gegeben ist das System Ax=b mit regulärer Matrix A und der Lösung x^*.
Sei $b + \Delta b$ die rechte Seite eines gestörten Systems und $x^* + \Delta x$ dessen Lösung, d.h.

$$A(x^* + \Delta x) = b + \Delta b\,.$$

Wegen $A\,x^* = b$ folgt $A\,\Delta x = \Delta b$ und

$$\Delta x = A^{-1}\,\Delta b \quad d.h. \|\Delta x\| \le \|A^{-1}\|\,\|\Delta b\| \quad . \tag{3.13}$$

Außerdem gilt

$$\|b\| = \|Ax^*\| \le \|A\|\,\|x^*\| \quad bzw. \quad \|x^*\| \ge \frac{\|b\|}{\|A\|} \; .$$

Damit gilt für die relative Änderung der Lösung x^*

$$\frac{\|\Delta x\|}{\|x^*\|} \le \|A^{-1}\| \bullet \|A\| \frac{\|\Delta b\|}{\|b\|} = cond(A) \frac{\|\Delta b\|}{\|b\|}$$

Def. 3.10: Die Zahl $cond(A) = \|A^{-1}\| \bullet \|A\|$ heißt Konditionszahl der Koeffizientenmatrix A zur verwendeten Matrixnorm $\| \bullet \|$.

Die relative Störung $\frac{\|\Delta b\|}{\|b\|}$ der rechten Seite des Systems kann damit eine um $\|A\| \bullet \|A^{-1}\|$ multiplizierte relative Änderung der Lösung bewirken. Die Zahl *cond(A)* ist von der verwendeten Norm abhängig und gibt den Verstärkungsfaktor der relativen Fehler an. Es gilt

$$cond(A) \ge 1,$$

denn

$$1 = \|E\| = \|A \bullet A^{-1}\| \le \|A\| \bullet \|A^{-1}\| = cond(A).$$

Matrizen mit großer Konditionszahl nennt man schlecht konditioniert. (Praktisch gibt lg cond(A) den zu erwartenden Verlust an gültigen Dezimalziffern an bezüglich des verwendeten Datenformats).

Bemerkung: Gleichung (3.13) kann auch anders interpretiert werden: Ist $\tilde{x}$ eine Näherungslösung von Ax=b und $r(\tilde{x}) = b - A\tilde{x} = Ax^* - A\tilde{x} = A\Delta x$ das Residuum, so ist $\tilde{x}$ exakte Lösung von $Ax = b - r(\tilde{x})$ *und* $\|\Delta x\| \le \|A^{-1}\|\,\|r(\tilde{x})\|$.

3.7.2. Störung der Koeffizientenmatrix des Systems

Satz 3.11: Sei $F \in R^{n \times n}$ eine (n,n)-Matrix und $\|F\| < 1$. Dann gilt

(a) E+F ist regulär, (b) $\|(E+F)^{-1}\| \le \frac{1}{1 - \|F\|}$.

Wir betrachten nun das System Ax=b (A regulär), und es sei A gestört zu $A + \Delta A$, dies bewirkt eine Änderung der Lösung zu $x^* + \Delta x$, d.h.

$$(A + \Delta A)(x^* + \Delta x) = b, \quad Ax^* = b. \tag{3.14}$$

Es sei nun

$$\Delta A = A \bullet F \quad mit \quad \|F\| < 1,$$

d.h. $A + \Delta A = A(E + F)$. und nach Satz 3.11 ist die Matrix $A + \Delta A$ regulär.

Aus (3.14) folgt $\Delta x = (A + \Delta A)^{-1} b - x^* = (A + \Delta A)^{-1} b - A^{-1} b$

$$= (A + \Delta A)^{-1} [\, A - (A + \Delta A)]\, A^{-1} b$$

Mit $A^{-1} b = x^*$ erhält man

$$\frac{\|\Delta x\|}{\|x^*\|} \le \|(A + \Delta A)^{-1}(-\Delta A)\| = \|(A(E + F))^{-1}(-A F)\| =$$

$$= \|-(E + F)^{-1} A^{-1} A F\| \le \|(E + F)^{-1}\| \bullet \|F\| \underset{(b)}{\le} \frac{\|F\|}{1 - \|F\|}.$$

Wegen $F = A^{-1} \Delta A$ gilt

$$\|F\| \le \|A^{-1}\| \bullet \|A\| \frac{\|\Delta A\|}{\|A\|} = cond(A) \frac{\|\Delta A\|}{\|A\|}$$

Insgesamt gilt: Falls $cond(A) \bullet \frac{\|\Delta A\|}{\|A\|} < 1$ gilt (d.h. ΔA ist genügend klein), so gilt

$$\frac{\|\Delta x\|}{\|x^*\|} \le \frac{cond(A)}{1 - cond(A) \bullet \frac{\|\Delta A\|}{\|A\|}} \bullet \frac{\|\Delta A\|}{\|A\|}$$

Damit ist die Verstärkung des relativen Fehlers wieder näherungsweise durch den Faktor cond(A) gegeben, wenn die Größe $cond(A) \frac{\|\Delta A\|}{\|A\|}$ klein ist.

3.7.3. Normeigenschaften orthogonaler Matrizen

Die Matrix $Q \varepsilon R^{nxn}$ heißt orthogonal, wenn $Q^T = Q^{-1}$ gilt.
Für orthogonale Matrizen gilt bezüglich der Spektralnorm

$$\|Q\|_2 = \sqrt{\lambda_{\max}(Q^T Q)} = \sqrt{\lambda_{\max}(E)} = 1, d.h. \quad \|Q\|_2 = \|Q^T\|_2 = \|Q^{-1}\|_2 = 1$$

Satz 3.12: Für alle orthogonalen (unitären) Matrizen gilt $cond_2(Q) = 1$.

Folgerung: Wird bei der Auflösung eines linearen Gleichungssystems Ax=b zur Transformation von A auf obere Dreiecksform eine orthogonale Matrix Q verwendet, d.h. es wird Ax=b überführt in $Q^T A x = Q^T b \; mit \; R = Q^T A$ obere Dreiecksmatrix, so tritt keine Konditionsverschlechterung gemessen in der 2-Norm auf, denn es gilt

$$cond_2(Q^T A) = cond_2(R) \le cond_2(Q^T) \bullet cond_2(A) = cond_2(A).$$

3.8. ORTHOGONALE MATRIZEN ZUR LÖSUNG LIN. GLEICHUNGSSYSTEME, QR-FAKTORISIERUNG

In einem Hauptschritt des Gauß-Algorithmus

$$A^{(j-1)} \rightarrow A^{(j)} = G_j A^{(j-1)}$$

kann sich sich die Konditionszahl stark vergrößern

$$cond(A^{(j)}) >> cond(A^{(j-1)})$$

aufgrund sehr großer Elemente in der Matrix G_j. Werden statt der Matrizen G_j orthogonale Matrizen Q_j gewählt, so kann sich die Kondition in der 2-Norm nicht verschlechtern. Wir untersuchen zunächst, wie geeignete Orthogonalmatrizen Q_j zu konstruieren sind.

3.8.1. Die HOUSEHOLDER-Spiegelungsmatrix

Im ersten Eliminationsschritt wird eine Matrix $Q = Q_1$ gesucht, die den ersten Spaltenvektor a^1 der Koeffizientenmatrix A so verändert, dass der transformierte Vektor nur in der 1.Koordinate ein Nicht-Null-Element besitzt, d.h. $\tilde{a}^1 = \rho e^1$.

Aufgabe: geg.: $x \varepsilon R^n, x \neq 0$ ges.: $Q \varepsilon R^{n x n}$ *mit* $Q x = \rho e^1 = \begin{pmatrix} \rho \\ 0 \\ \vdots \\ 0 \end{pmatrix}$.

Eine Lösung des Problems liefert die Householder-Matrix Q=H

$$H = E - 2 v v^T \quad mit \quad \|v\|_2 = 1 \tag{3.15}$$

(Beachte: $v v^T$ ist als dyadisches Produkt zweier Vektoren eine (n,n)-Matrix.)

Satz 3.13: Die Householder-Matrix H besitzt die Eigenschaften
(a) H ist symmetrisch,
(b) H ist orthogonal,
(c) H ist involutorisch, d.h. $H^2=E$.

Beweis:

(a) $H^T = (E - 2v v^T)^T = E^T - 2(v v^T)^T = E - 2v v^T = H$.

(b) $H \bullet H^T = H \bullet H = (E - 2v v^T)(E - 2v v^T) = E - 2 v v^T - 2 v v^T + 4 v v^T v v^T$

$= E - 4 v v^T + 4 v v^T = E \quad d.h. H^{-1} = H^T$.

(c) folgt aus (b) wegen $H = H^T$.

Geometrische Veranschaulichung
Das Bild Hx des Vektors x ist der an der Ebene E
mit Normalenvektor v gespiegelte Vektor x:

Beweis: Zerlege $x = x_{II} + x_I$ mit $x_{II} = \alpha v$.

Dann gilt $v^T x = \alpha v^T v + v^T x_I = \alpha \bullet 1 + 0 = \alpha$

bzw. $H\,x = (E - 2\,v\,v^T)\,x = x - 2\,v\,v^T\,x =$

$= \alpha\,v + x_I - 2\,\alpha\,v = -\alpha\,v + x_I = -x_{II} + x_I$.

2. Formulierung der Aufgabe

geg.: $x \varepsilon R^n, \quad x \neq 0$

ges.: $u \varepsilon R^n, \quad u \neq 0$ so, dass für $H = E - \frac{1}{\kappa} u\,u^T,\ \kappa = \frac{1}{2} u^T u = \frac{1}{2}\|u\|_2^2$

gilt: $H\,x = \begin{pmatrix} \rho \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \rho\,e^1$.

Bem.: $H = E - 2\,v\,v^T$ mit $v = \frac{u}{\|u\|_2}$ ist dann die Householdermatrix. Der Vektor *u* muß jetzt nicht mehr normiert sein, dies erleichtert seine Berechnung.

Lösung: $u := x - \rho\,e^1, \quad \rho = -sgn(x_1)\|x\|_2$, wobei $sign(x_1) = \begin{cases} 1 & für\ x_1 \geq 0 \\ -1 & für\ x_1 < 0 \end{cases}$.

Dann gilt: $H\,x = \rho\,e^1$.

Beweis: $u^T x = (x^T - \rho\,e^{1T})x = x^T x - \rho\,x_1 = \|x\|_2^2 + |x_1|\,\|x\|_2$,

$$\kappa = \frac{1}{2} u^T u = \frac{1}{2}(x^T - \rho\,e^{1T})(x - \rho\,e^1) = \frac{1}{2}(\|x\|_2^2 - 2\rho|x_1| + \rho^2)$$

$$= \|x\|_2^2 + |x_1|\,\|x\|_2 .$$

Damit gilt

$$H\,x = (E - \frac{u\,u^T}{\kappa})\,x = x - u\,\frac{u^T x}{\kappa} = x - u = x - (x - \rho\,e^1) = \rho\,e^1$$

Bem.: Der Faktor $sgn(x_1)$ verhindert Auslöschung in der 1. Koordinate von u.

3.8.2. Die QR-Faktorisierung

Die Matrix A soll in eine obere Dreiecksmatrix R überführt werden

$$A = A^{(1)} \overset{H_1}{\Longrightarrow} A^{(2)} \overset{H_2}{\Longrightarrow} \ldots \Rightarrow A^{(n)} = R .$$

Situation vor dem j-ten Hauptschritt:

$$A^{(j)} = \begin{pmatrix} * & * & \ldots * & \ldots & * \\ 0 & *\ldots & \ldots * & & \\ 0 & 0\ldots & a_{jj}^{(j)} & \ldots & a_{jn}^{(j)} \\ & & \ldots & & \\ 0 & 0\ldots & a_{nj}^{(j)} & & a_{nn}^{(j)} \end{pmatrix} \begin{matrix} \\ \\ j-1 \\ \\ n-j+1 \end{matrix}$$

Definiere die Transformationsmatrix H_j:

$$H_j = \begin{bmatrix} E_{j-1} & O \\ O & \overline{H}_j \end{bmatrix} \begin{matrix} \} & j-1 \\ \} & n-j+1 \end{matrix}, \qquad \overline{H}_j = E_{n-j+1} - \frac{1}{\kappa_j} \bar{u}^j \bar{u}^{j^T}.$$

Der Vektor $\bar{u}^j \varepsilon R^{n-j+1}$ wird so bestimmt, dass gilt $\overline{H}_j \begin{pmatrix} a_{jj}^{(j)} \\ \vdots \\ \vdots \\ a_{nj}^{(j)} \end{pmatrix} = \begin{pmatrix} \rho_j \\ 0 \\ \vdots \\ 0 \end{pmatrix}$.

Die Matrix H_j lässt in $A^{(j)}$ alle Elemente unverändert außer derer von $\overline{A}^{(j)}$ (letzte *n-j+1* Zeilen und Spalten), H_j ist orthogonale Matrix.

<u>Bestimmung von $\overline{H}_j$</u> :

1) Quadrat des Betrages der ersten Spalte $\bar{x}$ von $\overline{A}^{(j)}$ bestimmen:

$$s_j := \sum_{i=j}^{n} a_{ij}^{(j)2} = \|\bar{x}\|_2^2, \quad \textit{ist A nichtsingulär, so gilt } s_j > 0.$$

Wäre $s_j = 0$, so würde gelten $|A^{(j)}| = a_{11}^{(j)} \dots a_{j-1,j-1}^{(j)} |\overline{A}^{(j)}| = 0$, dann wäre auch $|A| = 0$.

2) $\rho_j := \begin{cases} \sqrt{s_j}, & \textit{falls } a_{jj}^{(j)} < 0 \\ -\sqrt{s_j}, & \textit{sonst} \end{cases}$

3) $\kappa_j := s_j - \rho_j a_{jj}^{(j)}$

4) $\bar{u}^j := \begin{bmatrix} a_{jj}^{(j)} - \rho_j \\ a_{j+1,j}^{(j)} \\ .. \\ a_{n,j}^{(j)} \end{bmatrix}$

5) Bestimmung der transformierten Matrix (ohne $\overline{H}_j$ explizit zu berechnen)

$$\overline{H}_j \overline{A}^{(j)} = \overline{A}^{(j)} - \bar{u}^j \bullet y^{j^T} \quad \textit{und} \quad y^{j^T} = \frac{1}{\kappa_j} \bar{u}^{j^T} \overline{A}^{(j)}$$

(1. Spalte ist Null bis auf 1.Element ρ_j .)

<u>Speichertechnik:</u> Die Rechnung kann in-situ, d.h. auf den Platz der Matrix A ausgeführt werden, zusätzlich ist ein n-dimensionales real-Feld erforderlich, wenn alle Informationen über die Transformationsmatrizen $\overline{H}_j$ aufgehoben werden sollen.

$$\begin{pmatrix} - & * & * & * & * \\ & - & r_{ij}\ (i<j) & * & * \\ & & * & * & * \\ u^1 & u^2 & \dots & u^{n-1} & * \\ & & & & r_{nn} \end{pmatrix} \quad \left(r_{11} = \rho_1 \quad r_{22} = \rho_2 \qquad r_{nn} = a_{nn}^{(n-1)} \right)$$

Algorithmus: tol ist eine vorgegebene Abbruchgenauigkeit

$j = 1,.2,...,n-1 \quad (Hauptschritte)$

$s = \sum_{i=j}^{n} a_{ij}^2, \quad wenn\ s < tol \quad stop$

$\rho_j = \begin{cases} \sqrt{s}, & falls\ a_{jj} < 0 \\ -\sqrt{s}, & sonst \end{cases}$

$\kappa = s - \rho_j a_{jj}$

$a_{jj} = a_{jj} - \rho_j \quad (1.Koordinate\ von\ u^j)$

$k = j+1,...,n$

$y_k = (\sum_{i=j}^{n} a_{ij} a_{ik}) / \kappa$

$i = j,...,n$

$a_{ik} = a_{ik} - a_{ij} y_k$

Lösung eines linearen Gleichungssystems Ax=b

- Transformation der Matrix *A* in $Q^T A = R$
- Transformation der rechten Seite b: $c = H_{n-1} H_{n-2} \dots H_1 b$
- Lösung des Dreieckssystems Rx=c

Man kann zeigen, dass man die Transformation von b erhält, indem der Algorithmus auf die erweiterte Koeffizientenmatrix $(A \vdots b)$ angewendet wird. Definiert man b als (n+1)-te Spalte von A, so wird b mit umgeformt, wenn die 2.Schleife des Algorithmus über k=j+1,…,n+1 läuft.

Aufwand: $\frac{2}{3} n^3$ Operationen, d.h. etwa doppelt so viel wie der gewöhnlicher Gauß-Algorithmus.

3.9. Lineare Ausgleichsrechnung, überbestimmte lin. Gleichungssysteme

Problemstellung:
Zu m Messdatensätzen (t_j, y_j) j=1,…,m konstruiere man eine Modellfunktion y=f(t), welche das Verhalten der Messdaten möglichst gut wiedergibt. Für f(t) wird ein Ansatz mit noch unbekannten Parametern $x_1, \dots, x_n$ gemacht:

$$y = f(t) = x_1 g_1(t) + x_2 g_2(t) + \dots + x_n g_n(t).$$

Die Funktionen $g_i(t) \quad i = 1,2,...,n$ sind ein System geeignet gewählter Modellfunktionen. Die Anzahl der Messdaten m ist dabei im Allgemeinen deutlich größer als die Zahl der noch unbekannten Parameter n.

Die Methode der kleinsten Quadrate

Umformulierung der Aufgabe: Die Modellfunktion y=y(t)

$$y(t) = x_1 g_1(t) + x_2 g_2(t) + \ldots + x_n g_n(t),$$

ausgewertet an den Meßstellen $t_j (j = 1,2,\ldots,m)$, definiert für die unbekannten Koeffizienten der Modellfunktionen $x_i (i = 1,2,\ldots,n)$ (n<m) ein überbestimmtes lineares Gleichungssystem (Fehlergleichungssystem)

$$\begin{aligned} y(t_1) &= x_1 g_1(t_1) + \ldots\ldots + x_n g_n(t_1) = y_1 \\ y(t_2) &= x_1 g_1(t_2) + \ldots\ldots + x_n g_n(t_2) = y_2 \\ &\vdots \\ y(t_m) &= x_1 g_1(t_m) + \ldots\ldots + x_n g_n(t_m) = y_m, \end{aligned}$$

kompakt: Ax=b

mit $A = \begin{pmatrix} g_1(t_1) & \cdots & g_n(t_1) \\ g_1(t_2) & & g_n(t_2) \\ \vdots & & \vdots \\ g_1(t_m) & \cdots & g_n(t_m) \end{pmatrix}$, $b = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$, $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$.

Das System ist wegen n<m i.a. unlösbar. Statt der exakten Lösung fordern wir jetzt nur näherungsweise Übereinstimmung von Modell $y(t_j)$ und Messwert y_j, d.h. dass das Residuum des Systems $\|Ax - b\|_2$ klein sein soll.

Neue Aufgabenstellung:

geg.: $A \varepsilon R^{m \times n}$ $(m \geq n)$, $b \varepsilon R^m$

ges.: $\hat{x} \varepsilon R^n$ so, dass $\|A\hat{x} - b\|_2 \leq \|Ax - b\|_2 \quad \forall x \varepsilon R^n$ (3.16)

sowie Berechnung von $\hat{r} = A\hat{x} - b \ldots$ Residuum.

Lösung: Wir definieren die zu minimierende Zielfunktion

$$F(x) := \|Ax - b\|_2^2 = \sum_{j=1}^{m} [\,(a_{j1} x_1 + \ldots + a_{jn} x_n) - b_j\,]^2 \rightarrow \min_{(x_1,\ldots,x_n)},$$

$$F(x) = (Ax - b)^T (Ax - b) = x^T A^T A x - 2 x^T A^T b + b^T b \rightarrow \min_{x \varepsilon R^n}.$$

Die notwendige Bedingung für ein Extremum von F(x) ist

$$\Delta F = \begin{pmatrix} \frac{\partial F}{\partial x_1} \\ \vdots \\ \frac{\partial F}{\partial x_n} \end{pmatrix} = 0.$$

Diese Bedingung ist auch hinreichend, da F(x) quadratisch in x ist und die Matrix

$\nabla^2 F = A^T A$ der zweiten partiellen Ableitungen positiv definit (bzw. positiv semidefinit) ist. Berechnung der Ableitungen:

$$\frac{\partial F}{\partial x_i} = e_i^T A^T A x + x^T A^T A e_i - 2 e_i^T A^T b$$

$$= 2 e_i^T (A^T A x - A^T b) = 0 \quad i = 1,2,\ldots,n .$$

Damit ist die notwendige Bedingung erfüllt, wenn gilt $A^T A x = A^T b$ bzw. $A^T (A x - b) = A^T r = 0$.

Satz 3.14: Jede Lösung $x = \hat{x}$ der Aufgabe (3.16) ist Lösung des Normalgleichungssystems

$$A^T A x = A^T b$$

und umgekehrt ($\hat{x}$ wird als Quadratmittellösung des überbestimmten Systems Ax=b bezeichnet). Das Residuum $\hat{r} = A\hat{x} - b$ ist orthogonal zu allen Spalten von A. Das Normalgleichungssystem ist stets lösbar, die Lösung $\hat{x}$ ist eindeutig, falls rg(A)=n gilt. Das Residuum $\hat{r}$ ist stets eindeutig.

Beweis: Für jeden Vektor $b \varepsilon R^m$ existiert eine eindeutige Zerlegung der Form $b = u_I + u_{II}$, $u_I \varepsilon U = span\{a^1,\ldots,a^n\}$, $u_{II} \varepsilon U^\perp$ orthogonales Komplement von U. Offenbar löst $\|u_{II}\| = \min_x F(x)$ das Problem. Es ist also $u_I \varepsilon U$ eindeutig bestimmt. Gibt es mehrere Lösungen x^1, x^2 mit $A x^1 = u_I$, $A x^2 = u_I$, so gilt $\hat{r} = A x^1 - b = A x^2 - b$, d.h. $\hat{r}$ ist eindeutig. Im Fall rg(A)=n besitzt $A x = u_I$ eine *n-rg(A)=0* dimensionale Lösungsmenge, d.h. $\hat{x}$ ist eindeutig.

Lösungsmethode mittels Normalgleichungen

Vor.: rg(A)=n

Dann ist die Matrix $A^T A$ regulär, symmetrisch, positiv definit. Die Lösung des Normalgleichungssystems $A^T A x = A^T b$ kann dann mit dem Cholesky-Verfahren bestimmt werden.

Aufwand:

1) $A^T A$ berechnen	$\rightarrow \frac{1}{2} n^2 \bullet m\, Op$	
2) Cholesky-Faktorisierung	$\rightarrow \frac{1}{6} n^3\, Op$	$\approx \frac{1}{2} n^2 (m + \frac{1}{3} n)$
3) Vorwärts-/Rückwärtssubstitution	$\rightarrow n^2\, Op$	

Nachteil: Gefahr hoher Kondition der Matrix $A^T A$, da die Konditionszahl von A quadriert wird. Dies belegt folgendes Beispiel:

$$A=\begin{pmatrix} 1 & 1 & 1 \\ \varepsilon & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon \end{pmatrix} \Rightarrow A^T A=\begin{pmatrix} 1+\varepsilon^2 & 1 & 1 \\ 1 & 1+\varepsilon^2 & 1 \\ 1 & 1 & 1+\varepsilon^2 \end{pmatrix}$$

Theoretisch ist für $\varepsilon \neq 0$ rg(A)=3, d.h. $A^T A$ ist regulär. Praktisch ist aber z.B. für $\varepsilon = 10^{-5}$ und einer Maschinengenauigkeit $eps = 0.5 \bullet 10^{-8} : fl(1+\varepsilon^2)=1$ und die numerisch berechnete Matrix $A^T A$ ist singulär.
Selbst wenn $A^T A$ noch regulär ist kann für die Konditionszahl gelten

$$cond(A^T A) = \left\|(A^T A)\right\| \bullet \left\|(A^T A)^{-1}\right\| \to \infty ,$$

d.h. hohe Fehlerverstärkung.

Bem.: Falls $A^T A$ schlecht konditioniert ist, erhebt sich die Frage, ob die Aufgabe noch sinnvoll ist (z.B. ist eventuell die Modellwahl nicht angemessen). Wenn ja, sollte die Lösung nicht über die Normalgleichungen erfolgen.

Lösung mittels QR-Faktorisierung

Das System Ax=b kann mittels einer orthogonalen Transformation auf obere Dreiecksform gebracht werden:

$$\text{Ax=b} \to Q^T A x = \begin{pmatrix} R \\ \dots \\ 0 \end{pmatrix} x = Q^T b = c .$$

Dabei ist Q eine orthogonale (m,m)-Matrix, R eine obere Dreiecksmatrix vom Typ (n,n). Partitioniert man den m-dimensionalen Vektor c in der Form

$$c=\begin{pmatrix} c^{(1)} \\ c^{(2)} \end{pmatrix}, \quad c^{(1)} \varepsilon R^n , c^{(2)} \varepsilon R^{m-n}$$

so geht das Residuum r(x)=Ax-b über in $Q^T r = s(x) = \begin{pmatrix} R x - c^{(1)} \\ c^{(2)} \end{pmatrix}$.

Damit gilt:

$$\left\|r(x)\right\|_2^2 = \left\|Q\, s(x)\right\|_2^2 = s^T Q^T Q\, s = \left\|s(x)\right\|_2^2 = \left\|R x - c^{(1)}\right\|_2^2 + \left\|c^{(2)}\right\|_2^2 ,$$

und $\left\|r(x)\right\|_2^2$ nimmt ein Minimum an, wenn der erste Summand Null ergibt.
Damit ist $\hat{x}$ Lösung des Dreickssystems

$$R\,\hat{x} = c^{(1)} = \begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{pmatrix}$$

und $\|r(\hat{x})\|_2^2 = \|c^{(2)}\|_2^2 = c_{n+1}{}^2 + \ldots + c_m{}^2 = \sum_{i=n+1}^{m} c_i{}^2$.

Vorteil:
1) Die Kondition von A wird nicht verschlechtert, d.h. wenn rg(A)=n, so gibt es eine eindeutige Lösung $\hat{x}$ von $R\,x = c^{(1)}$.
2) Man hat ein numerisch günstiges Verfahren sowohl im Regelfall m=n (dann fehlt $c^{(2)}$, und $\hat{x}$ ist Lösung von Ax=b) und im Ausgleichsfall m>n.

Aufwand: 1. QR-Zerlegung von A $\rightarrow m \bullet n^2\,Op$
2. Rückwärtsrechnung $R\,x = c^{(1)}$ $\rightarrow \frac{1}{2}\,n^2\,Op$

d.h. für m>>n tritt ein nahezu doppelter Aufwand gegenüber der Normalgleichungsmethode auf.

4. Numerische Lösung nichtlinearer Gleichungen und Systeme

Das grundlegende Prinzip zur numerischen Lösung eines nichtlinearen Problems besteht in der Anwendung der Iteration. Dabei wird eine Lösung des Problems durch eine Folge von iterierten Näherungslösungen erzeugt. In jedem Iterationsschritt ist ein Ersatzproblem zu lösen, welches i.a. durch Linearisierung des nichtlinearen Ausgangsproblems entsteht.

4.1. PROBLEMSTELLUNG, ITERATIONSVERFAHREN

Ziel: Entwicklung von Verfahren zur Bestimmung einer Nullstelle x^* einer gegebenen Funktion f(x):

Gesucht: x^* mit $f(x^*) = 0$ (NST-Problem) (4.1)

Problemtypen:

1) $f : R^1 \to R^1$ Lösung einer reellen Gleichung für eine reelle Unbekannte $x \in R^1$.

2) $f : R^n \to R^n$ Lösung eines Systems von n Gleichungen für n reelle Unbekannte $x = (x_1,...,x_n)^T \in R^n$.

Methode: Im Gegensatz zu linearen Gleichungen bzw. Gleichungssystemen, bei denen die Lösung direkt mit endlich vielen arithmetischen Operationen berechnet werden kann, ist man hier auf iterative Methoden angewiesen:
Ausgehend von einem Startwert x_0 werden Näherungen $x_1, x_2,....$ erzeugt, von denen man hofft, dass sie gegen eine Lösung x^* von (1) konvergieren.

Neben der Frage der Konvergenz ist besonders die Konvergenzgeschwindigkeit von Interesse, wie folgendes Beispiel zeigt.

Bsp. 1: Berechnung der Standardfunktion

$$x = \sqrt{a} = sqrt(a), \quad a > 0$$

in einer höheren Programmiersprache. Diese Funktion wird u.U. sehr häufig aufgerufen und das Programm, welches sich dahinter verbirgt, sollte den Wert in Sekundenbruchteilen zur Verfügung stellen, wobei nur arithmetische Grundoperationen zu verwenden sind. Der Wert x ist auch Lösung der quadratischen Gleichung

$$f(x) = x^2 - a = 0 .$$

Zur Lösung dieser nichtlinearen Gleichung verwenden wir die Iteration

$$x_{k+1} = \Phi(x_k) = \frac{1}{2}\left(x_k + \frac{a}{x_k} \right)$$

mit dem Startwert

$$x_0 = \begin{cases} a & für\ a > 1 \\ 1 & für\ 0 < a \le 1 \end{cases} .$$

Konkret für a=17 erhält man die Iterationsfolge

$x_0 = 17.000\,000\,000\,000$
$x_1 = 9.000\,000\,000\,000$
$x_2 = 5.444\,444\,444\,444$
$x_3 = 4.283\,446\,712\,018$
$x_4 = 4.126\,106\,627\,581$
$x_5 = 4.123\,106\,716\,962$
$x_6 = 4.123\,105\,625\,617$

Die Iteration wird schnell stationär. Praktisch tritt von x_3 an eine Verdopplung der gültigen Dezimalziffern in jeder Iteration ein (Kennzeichen quadratischer Konvergenz).

4.2. FIXPUNKT-ITERATION UND KONTRAHIERENDE ABBILDUNG

Wir betrachten zunächst den Fall n=1, d.h. der Lösung einer Gleichung für eine Unbekannte. Die Mehrzahl der Iterationsverfahren ist vom Typ der Einschrittverfahren

$$x_{k+1} = \Phi(x_k)$$

Bei Konvergenz erhält man einen Punkt

x^* mit $x^* = \Phi(x^*)$. (FP-Problem) (4.2)

Einen solchen Punkt nennen wir **Fixpunkt von Φ.**

Idee der FP-Iteration: Die Nullstellenform f(x)=0 äquivalent umformen in eine Fixpunktform $x = \Phi(x)$.

Für einen Startwert x_0 wird die Iterationsfolge erzeugt, bei Konvergenz liefert das Grenzelement eine Lösung des Ausgangsproblems. Nicht jede naiv erzeugte Fixpunkt-Iteration muss konvergieren wie das folgende Beispiel zeigt:

Bsp. 2: Gleichung: f(x) = x+ln(x) = 0. Aus einer Grafik lesen wir ab: $x^* \approx 0.5\,0$
Die Iteration $x_{k+1} = \Phi(x_k) = -\ln(x_k)$ ergibt die Iterationsfolge

$x_0 = 0.5$
$x_1 = 0.6931$
$x_2 = 0.3665$
$x_3 = 1.0037$
$x_4 = -0.0037$
$x_5 = -\ln(x_4)$ *nicht reell*

Die FP-Iteration $x_{k+1} = \Phi(x_k)$ divergiert also, da die Iterierten das Intervall verlassen, in welchem $\Phi(x)$ definiert ist.

Im Zusammenhang mit der FP-Iteration entstehen 3 Fragen:

A. Konstruktion geeigneter Iterationsfunktionen $\Phi(x)$;
B. Konvergenzbedingungen;
C. Konvergenzgeschwindigkeit.

Bevor wir uns der ersten Frage zuwenden, die auf konkrete Algorithmen führt, wollen wir zunächst die Konvergenzfragen klären.

Dazu führen wir als erstes eine heuristische Betrachtung durch: Ein FP $x^* = \Phi(x^*)$ entspricht offenbar einem Schnittpunkt der beiden Kurven y=x und $y = \Phi(x)$ für x aus einem Intervall I. Wir nehmen an, dass $y = \Phi(x)$ stetig differenzierbar ist und betrachten die folgenden Fälle:

Grafische Interpretation der FP-Iteration $x_{k+1} = \Phi(x_k)$:

Alternierende Konvergenz

Monotone Konvergenz

Divergenz des Verfahrens:

Analytische Untersuchung:
Konvergenz der FP-Iteration liegt offenbar vor, wenn der Abstand

$$|x_{k+1} - x_k| = |\Phi(x_k) - \Phi(x_{k-1})|$$

zweier aufeinanderfolgender Iterierter gegenüber $|x_k - x_{k-1}|$ in der vorhergehenden Iteration abnimmt und für $k \to \infty$ gegen Null strebt.

Def. 4.1: Sei $I = [a,b] \subset R$ ein Intervall und $\Phi : I \to R$ eine Abbildung. $\Phi(x)$ heißt kontrahierend auf I, wenn es eine Konstante L gibt mit 0≤ L<1 und

$$|\Phi(x_1) - \Phi(x_2)| \le L|x_1 - x_2| \quad \forall x_1, x_2 \in I \tag{4.3}$$

Bemerkung.:
1. Anschaulich bedeutet (4.3), dass der Abstand der Bildpunkte $\Phi(x_1), \Phi(x_2)$ von zwei beliebigen Punkten x_1, x_2 stets kleiner als der Abstand $|x_1 - x_2|$ der Urbildpunkte ist.
2. Eine Abbildung $\Phi(x)$, welche (4.3) erfüllt, heißt Lipschitzstetig mit der Lipschitz-Konstanten L. $\Phi(x)$ ist kontrahierend, wenn zusätzlich 0≤L<1 gilt.

Lipschitz-Stetigkeit impliziert die Beschränktheit des Differenzenquotienten von $\Phi(x)$. Ist $\Phi(x)$ stetig differenzierbar auf dem Intervall I, so kann die Lipschitz-Konstante L wie folgt erklärt werden:

Satz 4.2: Ist $\Phi \in C^1(I)$, so gilt $\max\limits_{x_1, x_2 \in I} \dfrac{|\Phi(x_1) - \Phi(x_2)|}{|x_1 - x_2|} = \max\limits_{\xi \in I} |\Phi'(\xi)| = L < \infty$.

Beweis: Einfache Anwendung des Mittelwertsatzes ergibt, dass für jedes Paar $x_1, x_2 \in I$ ein Wert ξ aus dem Intervall zwischen x_1 und x_2 existiert mit der Eigenschaft

$$\Phi(x_1) - \Phi(x_2) = \Phi'(\xi)(x_1 - x_2).$$

Offenbar ist in Beispiel 2 die Abbildung $\Phi(x) = -\ln(x)$ in keinem Intervall I, welches den Punkt $x_0 = 0.5$ enthält, kontrahierend, denn es gilt $\Phi'(x_0) = -2$.

Die Kontraktivität von $\Phi(x)$ erweist sich als entscheidende Konvergenzbedingung. Es gilt der Konvergenzsatz der FP-Iteration

Satz 4.3: Sei $I = [a,b] \subset R$, und es gelten die Bedingungen
(a) $\Phi(x)$ bildet I auf sich ab, d.h. für $x \in I$ gilt $\Phi(x) \in I$;
(b) $\Phi(x)$ ist kontrahierend auf I mit der Konstanten L<1.
Dann gilt:
(i) Es existiert genau ein FP von $\Phi(x)$ im Intervall I: $x^* = \Phi(x^*)$.
(ii) Für jeden Startwert $x_0 \in I$ konvergiert die FP-Iteration gegen x^*, und es gilt

$$|x_{k+1} - x_k| \le L|x_k - x_{k-1}| \tag{4.4}$$

$$|x^* - x_k| \le \frac{L^k}{1-L}|x_1 - x_0| \tag{4.5}$$

Die Bedeutung der Lipschitz-Konstanten L für die Konvergenz wird aus den Formeln (4.4),(4.5) deutlich: Wegen (4.4) wird der Abstand aufeinanderfolgender Iterierter im Vergleich zum Abstand in der vorherigen Iteration um den Faktor L reduziert.
Relation (4.5) wird auch als a-priori Fehlerschätzung des Verfahrens bezeichnet. Bei bekannter Konstanten L gestattet sie nach Berechnung von x_1 eine Vorhersage des Fehlers. Dieser Fehler nimmt wegen des Faktors L^k wie eine geometrische Folge mit q=L ab. Die Fehlerreduktion pro Schritt ist umso größer, je näher L bei 0 liegt, und sie ist klein, wenn L nahe 1 ist.

Bem.: Satz 4.3 ist ein Spezialfall eines allgemeineren Prinzips, des BANACHschen Fixpunkt-Prinzips, welches in metrischen Räumen gilt. Wir werden dieses Prinzip auch bei der Übertragung auf Gleichungssysteme verwenden.

Charakterisierung der Konvergenzgeschwindigkeit

Def. 4.4: Es sei x^* ein Fixpunkt von $\Phi(x)$, und es gelte für die Iterierten $x_{k+1} = \Phi(x_k)$ für alle Startvektoren $x_0 \in U(x^*)$ die Beziehung

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^p} = C = const. \quad (0 < C < \infty) \tag{4.6}$$

dann hat das Iterationsverfahren $x_{k+1} = \Phi(x_k)$ die (lokale) Konvergenzordung p.

Speziell: p=1 Lineare Konvergenzordnung (zusätzlich ist C<1 in (4.6) zu fordern);
p=2 quadratische Konvergenz;
p=3 kubische Konvergenz.

Zwischenstufe: Der Fall p=1 und die Quotienten $C_k = \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|}$ streben gegen

Null wird als superlineare Konvergenz bezeichnet.

Andere Formulierung von Def. 4.4:
Das Iterationsverfahren mit der Verfahrensfunktion $\Phi(x)$ ist mindestens von der Konvergenzordung p(≥1), wenn es ein $\varepsilon > 0$ gibt mit

$$\|x - x^*\| < \varepsilon \Rightarrow \|\Phi(x) - x^*\| \le C\|x - x^*\|^p \qquad \text{(mit C<1, falls p=1).}$$

Ist also $\|x_k - x^*\| = a$, dann wird $\|x_{k+1} - x^*\|$ mindestens auf Ca^p reduziert. Im Fall p=1 ergibt sich somit eine Reduktion um den Faktor C=L. Der Fehler $\|x_0 - x^*\| = a_0$ der Anfangsnäherung wird dann in jeder Iteration um den Faktor C reduziert, wenn 0≤C<1 gilt, d.h. die Folge $\|x_k - x^*\|$ strebt gegen Null wie eine geometrische Folge mit q=C.
Im Fall p=1 heißt q=C die lineare Konvergenzrate.

Bestimmung der Konvergenzordung p

Satz 4.5: Sei $\Phi : I \to I$ kontrahierend auf dem Intervall I=[a,b] und es sei $\Phi(x)$ i (p+1)-mal stetig differenzierbar. Gilt in $x = x^*$

$$\Phi'(x^*) = \Phi''(x^*) = ... = \Phi^{(p-1)}(x^*) = 0 \quad und \quad \Phi^{(p)}(x^*) \neq 0,$$

so konvergiert die FP-Iteration gegen $x = x^*$ mindestens von der Ordnung p.

Beweis: Durch Taylorentwicklung p-ter Ordnung in $x = x^*$ folgt

$$x_{k+1} = \Phi(x_k) = \Phi(x^*) + \frac{\Phi'(x^*)}{1!}(x_k - x^*) + \frac{\Phi''(x^*)}{2!}(x_k - x^*)^2 + ... + \frac{\Phi^{(p-1)}(x^*)}{(p-1)!}(x_k - x^*)^{p-1} +$$
$$+ \frac{\Phi^{(p)}(x^*)}{p!}(x_k - x^*)^p + O(\|x_k - x^*\|^{p+1})$$

Damit gilt

$$|x_{k+1} - \Phi(x^*)| = |x_{k+1} - x^*| \le \left|\frac{\Phi^{(p)}(x^*)}{p!} + \varepsilon\right| |x_k - x^*|^p = C|x_k - x^*|^P .$$

Beachte: Je höher die Konvergenzordung p ist, umso weniger Iterationen werden i.a. benötigt, um eine vorgegebene Genauigkeit zu erreichen.

Abbruchbedingungen für die Iteration:

Eine Nullstelle einer nichtlinearen Funktion f(x) kann i.a. nicht im endlichen Raster der Maschinenzahlen dargestellt werden, so dass $x = x^*$ durch einen gerundeten Wert $\bar{x}$ repräsentiert wird. Dann ist aber f(x)=0 i.a. nicht durch eine Maschinenzahl zu erfüllen. Um einen Abbruch der Iterationen zu erhalten, ist eine Genauigkeit ε vorzugeben und die Abbruchbedingung

$$|f(x_k)| < \varepsilon \qquad (4.7)$$

zu prüfen. Verläuft f(x) in der Umgebung einer Nullstelle sehr flach, so kann (4.7) erfüllt sein, ohne dass die Iterierten stationär geworden sind. Es empfiehlt sich also, neben (4.7) eine zusätzliche Abbruchbedingung

$$\|x_k - x_{k-1}\| < \varepsilon \qquad (4.8)$$

zu testen, welche überprüft, ob die Folge $\{x_k\}$ stationär wird. Für große Werte von x kann dies in (4.7) und (4.8) zu unterschiedlichen Größenordnungen kommen (vor allem, wenn x betragsmäßig groß ist), so dass statt (4.8) der Test

$$\|x_k - x_{k-1}\| < \varepsilon \bullet \max\{1, \|x_k\|\} \qquad (4.9)$$

zu empfehlen ist.

4.3. SPEZIELLE ITERATIONSVERFAHREN

Es sollen nun konkrete Verfahren betrachtet werden. Zu lösen ist die Gleichung f(x)=0 in NST-Form bzw. in FP-Form $x = \Phi(x)$, $f, \Phi : R \to R.$.

A. Klassische Iterationsverfahren

4.3.1. Methode der Parallelsehnen

Man wählt eine Zahl m≠0 und setzt

$$\Phi(x) = x - mf(x). \tag{4.10}$$

Offenbar gilt für $x = x^*$ mit $f(x^*) = 0$ die FP-Gleichung $x^* = \Phi(x^*)$. Die Iteration lautet also

$$x_{k+1} = \Phi(x_k) = x_k - mf(x_k).$$

Konvergenzbedingung: Für die Ableitung der Iterationsfunktion (4.10) erhält man

$$\Phi'(x) = 1 - mf'(x).$$

Gilt in einem Intervall I=[a,b] mit $x^* \in I$ die Ungleichung

$$0 < mf'(x) < 2 \quad \forall x \in I$$

so folgt $|\Phi'(x)| < 1 \quad \forall x \in I$, d.h. die Iteration ist lokal konvergent.

Praktisch: Wählt man $\frac{1}{m} \approx f'(x^*)$, so gilt $|\Phi'(x^*)| \approx 0$, d.h. es würde nahezu lokal quadratische Konvergenz erreicht werden. Tatsächlich ist die Konvergenz linear, da die Bedingung nur näherungsweise erreicht wird. Häufig wird $m = f'(x_0)^{-1}$ als Parameterwert gewählt. Das Verfahren wird dann auch als vereinfachtes Newton-Verfahren bezeichnet.

Geometrische Bedeutung:
Der Iterationspunkt x_{k+1} entsteht als Schnitt der Sehne durch den Punkt $(x_k, f(x_k))$ mit der x-Achse:

Sehne: $m(y - y_k) = x - x_k$ bzw. $y = \frac{1}{m}(x - x_k) + y_k$

x_{k+1} so bestimmen, dass gilt $y_{k+1} = 0$

4.3.2. Das Newtonsche Iterationsverfahren

Auf systematische Weise erhält man eine Iterationsfunktion $\Phi(x)$ durch Taylorentwicklung von f(x) in den jeweiligen Iterationspunkten $x_0, x_1, x_2, ...$

$$f(x) = f(x_k) + (x - x_k) f'(x_k) + \frac{(x - x_k)^2}{2!} f''(x_k) + R_2 \tag{4.11}$$

Vernachlässigt man Terme höherer Ordnung (d.h. man verwendet eine Linearisierung von $f(x)$), so erhält man eine lineare Gleichung, deren Lösung eine verbesserte Näherung für eine Nullstelle $x = x^*$ mit $f(x^*) = 0$ liefert.

Damit erhält man aus (4.11) die lineare Gleichung zur Bestimmung von x_{k+1}

$$0 = f(x_k) + (x - x_k) f'(x_k)$$

bzw.

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \quad k = 0, 1, 2, ... \tag{4.12}$$

Die Iterationsvorschrift wird auch als Newton-Raphson-Verfahren 1.Grades bezeichnet oder kurz als Newton-Verfahren.

Geometrisch ist x_{k+1} der Schnittpunkt der Tangente an f(x) im Punkt $(x_k, f(x_k))$ mit der x-Achse, denn $y = f(x_k) + (x - x_k) f'(x_k)$ ist die Gleichung der Tangente in x_k, und für y=0 erhält man den Schnittpunkt mit der x-Achse.

Die Newton-Iteration kann als Iterationsverfahren mit der Verfahrensfunktion

$$\Phi(x) = x - \frac{f(x)}{f'(x)} \quad (Vor.: f'(x) \neq 0) \tag{4.13}$$

angesehen werden. Wegen

$$\Phi'(x) = 1 - \frac{f'(x) f'(x) - f(x) f''(x)}{[f'(x)]^2} = \frac{f(x) f''(x)}{[f'(x)]^2} \tag{4.14}$$

gilt $\Phi'(x^*) = 0$, wenn x^* eine Nullstelle von f(x) ist, d.h. (4.12) besitzt mindestens quadratische Konvergenzordnung p=2, falls $f'(x^*) \neq 0$ ist (vgl. Satz 4.5). Die Bedingung bedeutet, dass x^* einfache Nullstelle von f(x) ist. Für mehrfache Nullstellen gilt eine andere Konvergenzordnung. Wegen $\Phi'(x^*) = 0$ ist klar, dass aus Stetigkeitsgründen immer eine Umgebung $U(x^*)$ existiert, dass

$$C = \max_{x \in U(x^*)} |\Phi'(x)| < 1 \tag{4.15}$$

gilt. Offensichtlich ist C umso kleiner, je kleiner $U(x^*)$ gewählt ist und je größer $|f'(x^*)|$ ist bzw. je kleiner $|f''(x^*)|$ ist. Die Startnäherung x_0 sollte somit nahe einer Lösung liegen, um eine rasche Konvergenz zu sichern. Genauer gilt der Satz:

Satz 4.6: Es sei I=[a,b] ein Intervall, welches die Startnäherung x_0 und eine Lösung x^* enthalte. Ferner gelte

$$|f'(x^*)| > m \ \ und \ \ |f''(x^*)| < M \quad \forall x \in [a,b]$$

$$L = \frac{M}{2m}(b-a) < 1 \quad \text{(stets erfüllbar, wenn } m > 0 \text{ und b-a hinreichend klein)}$$

Dann konvergiert die Folge $\{x_k\}$ gegen x^* und es gilt

$$|x_k - x^*| \leq e_k = \frac{2m}{M} L^{2^k}. \tag{4.16}$$

Bem.: In Bedingung (4.16) ist die quadratische Konvergenz enthalten:

$$|x_k - x^*| \leq e_k = \frac{2m}{M} L^{2^k} = \frac{2m}{M} (L^{2^{k-1}})^2 = \frac{2m}{M} e_{k-1}^{\ 2} \tag{4.17}$$

wobei e_{k-1} die Fehlerschranke der (k-1)ten Näherung bezeichnet:

$$|x_{k-1} - x^*| \leq e_{k-1} = \frac{2m}{M} L^{2^{k-1}}.$$

Konvergenz des Newton-Verfahrens bei mehrfachen Nullstellen:
Ist x^* eine m-fache Nullstelle, und f(x) ist m-mal stetig differenzierbar in x^*, so folgt aus der Taylorentwicklung

$$f(x) = f(x^*) + (x-x^*) f'(x^*) + \frac{(x-x^*)^2}{2!} f''(x^*) + ... + \\ + \frac{(x-x^*)^{m-1}}{(m-1)!} f^{(m-1)}(x^*) + \frac{(x-x^*)^m}{m!} f^{(m)}(x^* + \vartheta h)$$

wegen $f(x^*) = 0$ die Darstellung

$$f(x) = (x-x^*)[f'(x^*) + \frac{(x-x^*)^1}{2!} f''(x^*) + ... + \frac{(x-x^*)^{m-1}}{m!} f^{(m)}(x^* + \vartheta h)].$$

Für m>1 muss folglich der Klammerausdruck für $x = x^*$ verschwinden, d.h. es muss $f'(x^*) = 0$ sein. Fortführung dieser Überlegung führt auf $f''(x^*) = 0$ bis $f^{(m-1)}(x^*) = 0$.
Damit gilt:

Charakterisierung einer m-fachen Nullstelle: (4.18)
Für eine m-fache Nullstelle $x = x^*$ gilt
$f(x^*) = 0,\ f'(x^*) = 0, ...,\ f^{(m-1)}(x^*) = 0$ und die Funktion f(x) besitzt die Darstellung $f(x) = (x-x^*)^m g(x) \quad mit\ g(x^*) \neq 0$ und g(x) ist mindestens einmal stetig differenzierbar, wenn f(x) m-mal stetig differenzierbar ist.

Aus (4.18) folgt für die Ableitung

$$f'(x) = m(x-x^*)^{m-1} g(x) + (x-x^*)^m g'(x) \qquad (4.19)$$

Das Newton-Verfahren wird damit mit folgender Iterationsfunktion durchgeführt:

$$\Phi(x) = x - \frac{f(x)}{f'(x)} = x - \frac{(x-x^*)^m g(x)}{m(x-x^*)^{m-1} g(x) + (x-x^*)^m g'(x)}$$

$$= x - \frac{(x-x^*) g(x)}{m g(x) + (x-x^*) g'(x)} \quad .$$

Für die Ableitung gilt

$$\Phi'(x) = 1 - \frac{[mg(x) + (x-x^*)g'(x)] \cdot [g(x) + (x-x^*)g'(x)] - (x-x^*)g(x)[mg(x) + (x-x^*)g'(x)]'}{[mg(x) + (x-x^*)g'(x)]^2}$$

und in $x = x^*$

$$\Phi'(x^*) = 1 - \frac{mg(x^*)g(x^*)}{[mg(x^*)]^2} = 1 - \frac{1}{m} \neq 0 \qquad (4.20)$$

Damit ist das Newton-Verfahren für mehrfache Nullstellen nur noch linear konvergent mit einer linearen Konvergenzrate C≥½.

Modifikationen des N-Verfahrens bei mehrfachen Nullstellen

a) Modifikation bei bekannter Vielfachheit m einer Nullstelle:
Besitzt f(x) bei $x = x^*$ eine m-fache Nullstelle, so könnte eine naheliegende Modifikation sein

$$x_{k+1} = \overline{\Phi}(x_k) = x_k - m\frac{f(x_k)}{f'(x_k)} \quad k = 0, 1, 2,... \tag{4.21}$$

Wegen (4.20) gilt nämlich

$$\lim_{x \to x^*} \frac{d}{dx}\frac{f(x)}{f'(x)} = \frac{1}{m}$$

und damit

$$\overline{\Phi}'(x^*) = 1 - m\frac{1}{m} = 0,$$

so dass (4.21) nach (Satz 4.5.) quadratisch konvergent ist. Da die Ordnung einer Nullstelle i.a. selten vorher bestimmbar ist, ist (4.21) nur von eingeschränkter Bedeutung.

b) Modifikation bei unbekannter Vielfachheit m:
Aus (4.18) folgt, dass die Funktion

$$F(x) = \frac{f(x)}{f'(x)} = \frac{(x - x^*)g(x)}{mg(x) + (x - x^*)g'(x)}$$

wegen $g(x^*) \neq 0$ bei $x = x^*$ nur eine einfache Nullstelle hat, d.h. das Verfahren

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)} \quad k = 0, 1, 2,...$$

ist quadratisch konvergent, und m braucht vorher nicht bekannt zu sein. Man beachte aber, dass $F'(x)$ zu berechnen ist, d.h. es sind zweite Ableitungen von f(x) nötig.

Globalisierung des Newton-Verfahrens durch Schrittweitedämpfung

Das Newton-Verfahren verlangt zur Konvergenz relativ gute Startpunkte, wenn x_0 zu weit von einer Lösung $x = x^*$ entfernt liegt, kann es leicht versagen.

Bei der Durchführung ist damit in jedem Schritt zu überprüfen, ob $|f(x)|$ verringert wurde. Andernfalls kann man eine Schrittweite $\lambda = \lambda_k$ so bestimmt, dass im folgenden Iterationspunkt

$$x_{k+1} = x_k - \lambda_k \frac{f(x_k)}{f'(x_k)}$$

gilt $|f(x_{k+1})| < |f(x_k)|$. Für $\lambda = \lambda_k$ kann z.B. das Maximum der Schrittweiten $\left\{1, \frac{1}{2}, \frac{1}{4}, ...\right\}$ verwendet werden, für welche $|f(x_{k+1})| < |f(x_k)|$ erreicht wird.

Bem.: Die so beschriebene Schrittweitestrategie sichert nicht in allen Fällen Konvergenz. In der Fachliteratur werden konvergenzsichernde Strategien beschrieben.

4.3.3 Sekantenverfahren

Statt der Ableitung $f'(x_k)$ im Newton-Verfahren kann man den Anstieg der Sekante durch die letzten zwei Iterationspunkte $(x_{k-1}, f(x_{k-1})), (x_k, f(x_k))$ einsetzen und damit ohne Ableitungsberechnungen arbeiten. Das entsprechende Verfahren

$$x_{k+1} = x_k - \frac{f(x_k)}{s_k} \quad mit \quad s_k = \frac{f(x_k) - f(x_{k-1})}{x_k - x_{k-1}} \tag{4.22}$$

wird als Sekantenverfahren bezeichnet, es benötigt zwei Startwerte x_0, x_1, verlangt jedoch nur eine Funktionswertberechnung pro Iterationsschritt. Für die lokale Konvergenzordnung des Verfahrens kann gezeigt werden, dass gilt $p^2=p+1$. Dies ist für $p=1.618$ erfüllt. Damit sind 2 Schritte des Verfahrens (Aufwand: 2 Funktionswertberechnungen) vergleichbar mit einem Schritt eines Verfahrens der Ordnung $p^2=p+1=2.618$, d.h. die lokale Konvergenz ist besser als beim Newton-Verfahren (Aufwand: Funktionswert und Ableitungsberechnung), wenn der Aufwand für die Ableitungsberechnung im Newton-Verfahren nicht deutlich geringer ist als der für den Funktionswert.

B. Einschließungsverfahren

In modernen Softwaresystemen werden zur Lösung nichtlinearer Gleichungen i. a. Einschließungsverfahren verwendet. Einschließungsverfahren sind sicher konvergierende Iterationsverfahren, bei denen in jedem Iterationsschritt ein Intervall bestimmt wird, welches eine Lösung der Gleichung f(x)=0 enthält. Die Verfahren sind nur anwendbar bei Vorzeichenwechsel des Funktionswertes in Umgebungen der Nullstelle, d.h. in der Regel für einfache Nullstellen. Die Verfahren benötigen ein Startintervall mit Punkten a,b in denen Vorzeichenwechsel des Funktionswertes vorliegt, d.h. $f(a) \bullet f(b) < 0$.

4.3.4. Intervallhalbierung (Bisektionsverfahren)

Voraussetzungen: 1.) $I_0 = [a_0, b_0]$ so bestimmt, dass gilt $f(a_0) \bullet f(b_0) < 0$.
2.) f(x) stetig auf I.

Nach dem Zwischenwertsatz existiert dann ein $x^* \in I_0$ mit $f(x^*) = 0$. Es wird eine Folge von Intervallen $I_k = [a_k, b_k]$ konstruiert, in denen x^* liegen muss.

Algorithmus:

$l_0 = (b-a)$ $(Länge\ des\ Start\,\text{int}\,ervalls)$

$k = 0, 1, .2, ...$ $(Iterationsschritte)$

$$m_{k+1} = \frac{1}{2}(a_k + b_k)$$

für $f(m_{k+1}) f(b_k) < 0$

$$a_{k+1} = m_{k+1};\ b_{k+1} = b_k$$

sonst

$$a_{k+1} = a_k;\ b_{k+1} = m_{k+1}$$

$l_{k+1} = l_k / 2$ *falls* $l_{k+1} < eps$ *stop*

Konvergenzordnung:
Nach n Schritten ist die Nullstelle x^* in ein Intervall eingeschlossen mit der Intervallänge

$$l_n = b_n - a_n = \frac{b_0 - a_0}{2^n}$$

d.h. jede Näherung $x_n \in [a_n, b_n]$ erfüllt die Bedingung

$$|x_n - x^*| \le \frac{|b_0 - a_0|}{2^n} \qquad (4.23)$$

Damit liegt lineare Konvergenz mit der linearen Konvergenzrate C=½ vor.

> **Satz 4.8**: Das Bisektionsverfahren verbessert die Genauigkeit einer Näherung $x_n \in [a_n, b_n]$ pro Iterationsschritt um eine Binärziffer, d.h. man benötigt 3-4 Iterationen für 1 gültige Dezimalziffer.

Vorteil: Robust, einfach, nur Vorzeichen von f müssen exakt bestimmt werden; extrem hohe Genauigkeit möglich.
Nachteil: Nur lineare Konvergenzordnung.

4.3.5. Regula falsi

Die Regula falsi erhält man aus dem Sekantenverfahren. Dabei wird im k-ten Schritt der größte Index $l < k$ bestimmt, für den gilt $f(x_k) \bullet f(x_l) < 0$. (d.h. Vorzeichenwechsel im Intervall zwischen x_k, x_l und es liegt ein Einschließungsintervall vor). Die Nullstelle der Sekante zwischen x_k, x_l liefert den neuen Iterationspunkt

$$x_{k+1} = x_k - \frac{f(x_k)}{s_k} \quad mit \quad s_k = \frac{f(x_k) - f(x_l)}{x_k - x_l} \qquad (4.24)$$

Bewahrt f''(x) in einem Einschließungsintervall das Vorzeichen (f(x) ist in dem Intervall konvex bzw. konkav), so bleibt einer der Intervallendpunkte für alle weiteren Iterationen fixiert, der andere konvergiert gegen die Lösung, wobei die Konvergenz linear ist. Damit ziehen sich die Einschießungsintervalle nicht auf einen Punkt zusammen und die Intervalllängen streben nicht wie gewünscht gegen Null.

4.3.6. Illinois-Verfahren, Pegasus-Verfahren u.a.

Das Illinois-Verfahren und damit verwandten Iterationsverfahren verbessern die Regula falsi dahingehend, dass sich die Einschließungsintervalle auf einen Lösungspunkt zusammenziehen. Die Konvergenz ist überlinear und mit dem der Sekantenmethode vergleichbar. Aufgrund der sicheren Konvergenz und vergleichbar hohen Konvergenzordnung sind diese Verfahren z.B. auch dem Newton-Verfahren vorzuziehen.

Algorithmus:

Die k-te Iteration startet von den Randpunkten x_1, x_2 eines Einschliessungsintervalls. Im Fall $|x_2 - x_1| < eps$ bricht das Verfahren ab, jeder Punkt aus dem Intervall zwischen x_1, x_2 kann als Näherungslösung verwendet werden. Im andern Fall wird ein Sekantenschritt durchgeführt

$$z = x_1 - \frac{f_1}{s} \quad mit \quad s = \frac{f_2 - f_1}{x_2 - x_1}$$

Ist $|f_z| = |f(z)| < eps$, so bricht das Verfahren mit z als Näherungslösung ab, andernfalls wird ein neues Einschließungsintervall berechnet:

(a) Für $f_z \bullet f_2 < 0$ liegt x^* zwischen x_2 und z, man setzt

$x_1 := x_2,\ f_1 := f_2, x_2 := z,\ f_2 := f_z$.

(b) Sonst liegt x^* zwischen x_1 und z, es wird dann der Funktionswert an der Stelle x_1 um einen Faktor m (0<m<1) verkürzt, um ein Stehenbleiben eines Randpunktes zu verhindern, d.h. man setzt $x_2 := z,\ f_2 := f_z, x_1\ bleibt,\ f_1 := mf_1$.

Die einzelnen Verfahrenstypen unterscheiden sich in der Wahl des Verkürzungsfaktors m. Das Illinois-Verfahren verwendet m=0.5, beim Pegasus-Verfahren wird $m = \frac{f_2}{f_2 + f_z}$ gesetzt und beim Verfahren von Anderson/Björk $m = 1 - \frac{f_z}{f_2}$, falls der Wert nichtnegativ ist, sonst wird m = 0.5 gesetzt.

4.3.7. Genauigkeitsfragen bei der Nullstellenbestimmung von Polynomen

a) einfache Nullstellen: Es sei $x = \xi$ eine Nullstelle von p(x), und $\xi(\varepsilon)$ bezeichne die entsprechende Nullstelle eines gestörten Polynoms

$$p_\varepsilon(x) = p(x) + \varepsilon g(x) \tag{4.25}$$

wobei g(x) wieder ein Polynom ist. Man kann dann zeigen, dass $\xi(\varepsilon)$ für kleines ε eine analytische Funktion ist mit $\xi(0) = \xi$. Offenbar gilt

$$p_\varepsilon(\xi(\varepsilon)) = p(\xi(\varepsilon)) + \varepsilon g(\xi(\varepsilon)) \equiv 0.$$

Durch Differentiation nach ε folgt

$$\frac{d}{d\varepsilon} p_\varepsilon(\xi(\varepsilon)) = p'(\xi(\varepsilon))\xi'(\varepsilon) + g(\xi(\varepsilon)) + \varepsilon g'(\xi(\varepsilon))\xi'(\varepsilon) \equiv 0$$

und für $\varepsilon = 0$ erhält man

$$\xi'(0)) = -\frac{g(\xi(0))}{p'(\xi(0))} \quad mit \quad p'(\xi(0)) \neq 0 \text{ für einfache Nullstellen.}$$

Damit gilt in 1.Näherung

$$\xi(\varepsilon) \doteq \xi + \varepsilon\xi'(0) \tag{4.26}$$

d.h. die Störung der Nullstelle ist proportional zu ε, der Störung des Polynoms.

b) mehrfache Nullstellen: Für eine m-fache Nullstelle $x = \xi$ von p(x) kann man zeigen, dass das gestörte Polynom eine Nullstelle

$$\xi(\varepsilon) \doteq \xi + h(\varepsilon^{\frac{1}{m}})$$

besitzt, wobei h(t) für kleines t eine analytische Funktion ist mit h(0)=0.

Durch m-fache Differentiation von $p_\varepsilon(\xi(\varepsilon)) = p(\xi(\varepsilon)) + \varepsilon g(\xi(\varepsilon)) \equiv 0$ erhält man

$$\xi(\varepsilon) \doteq \xi + \varepsilon^{\frac{1}{m}} \left[-\frac{m!\, g(\xi)}{p^{(m)}(\xi)} \right]^{\frac{1}{m}} \tag{4.27}$$

d.h. die Störung der Nullstelle ist proportional zu $\varepsilon^{\frac{1}{m}}$ und damit wesentlich größer als ε .

4.4. FIXPUNKT-ITERATION FÜR NICHTLINEARE GLEICHUNGSSYSTEME

Problemstellung: Gegeben ist ein nichtlineares Gleichungssystem der Form

$$\begin{aligned} f_1(x_1, x_2, ..., x_n) &= 0 \\ f_2(x_1, x_2, ..., x_n) &= 0 \\ \vdots \qquad\qquad &\vdots \\ f_n(x_1, x_2, ..., x_n) &= 0 \end{aligned} \tag{4.28}$$

bzw. kompakt

$$f(x) = 0, \quad f = (f_1, f_2, ..., f_n)^T : R^n \to R^n, \quad x \in R^n.$$

Gesucht ist ein Punkt $x^* \in R^n$ mit $f(x^*) = 0$.

Fixpunkt-Iteration: Das System sei auf Fixpunkt-Form gebracht

$$\begin{aligned} x_1 &= \Phi_1(x_1, x_2, ..., x_n) \\ x_2 &= \Phi_2(x_1, x_2, ..., x_n) \\ &\vdots \\ x_n &= \Phi_n(x_1, x_2, ..., x_n) \end{aligned} \tag{4.29}$$

Genau wie im Fall n=1 können wir die FP-Iteration definieren

$$x^{k+1} = \Phi(x^k) \quad k = 0, 1, 2, ... \tag{4.30}$$

$x^0 \in R^n$ sei ein bekannter Startvektor. Mit $\|x\|$ bezeichnen wir eine Vektornorm des R^n.

Konvergenz:

Satz 4.9: (Banachscher FP-Satz)
Es sei $D \subset R^n$ eine abgeschlossene Menge, und $\Phi : D \to R^n$ erfülle folgende Bedingungen:
(i) $\Phi(x)$ bildet D auf sich ab;
(ii) es existiert eine Konstante C<1 mit

$$\|\Phi(x) - \Phi(y)\| \le C\|x - y\| \quad \forall x, y \in D \quad \text{(Kontraktivität der Abb. } \Phi \text{)}.$$

Dann besitzt (4.28) genau eine Lösung $x^* \in D$, und (4.30) konvergiert für jedes $x^0 \in D$ gegen x^*.

Wenn die Abbildung Φ differenzierbar ist, dann kann Bedingung (ii) wieder über die Bedingung an die Ableitung erfüllt werden $(n = 1 \Rightarrow |\Phi'(x)| < 1.)$ Hier tritt an Stelle der Ableitung die Matrix der partiellen Ableitungen

$$\frac{\partial \Phi}{\partial x} = \left(\frac{\partial \Phi_i}{\partial x_j} \right)_{i,j=1,2,..,n}$$

Jacobi-Matrix vom Typ (n,n)

Satz 4.10: Sei D abgeschlossen, konvex und $\Phi : D \to R^n$ partiell differenzierbar nach allen x_i. Gilt

$$\left\| \frac{\partial \Phi}{\partial x} \right\| \le C < 1 \quad \forall x \in D \tag{4.31}$$

für eine beliebige Matrixnorm, so ist Bedingung (ii) in Satz 4.9 für die entsprechende Vektornorm erfüllt.

Beweis: Mittels Taylorsatz.

Folgerung: Besitzt $\Phi(x)$ einen Fixpunkt $x^* \in D$ und ist $\Phi(x)$ in D stetig partiell differenzierbar mit der Matrix $\frac{\partial \Phi(x^*)}{\partial x}$ und gilt für den Spektralradius dieser Matrix

$$\rho\left(\frac{\partial \Phi(x^*)}{\partial x}\right) < 1 \ , \tag{4.32}$$

so gibt es eine Umgebung (Kugel) von $x^* \in D$ so, dass (4.30) für jedes x^0 aus dieser Umgebung konvergiert. Wichtig ist somit neben der Konstruktion einer geeigneten Verfahrensfunktion $\Phi(x)$ die Kenntnis eines Startpunktes x^0 aus der Umgebung einer Lösung.

Satz 4.9 sichert wieder lineare Konvergenz mit Konvergenzfaktor C, d.h. der Abstand $\|x^k - x^*\|$ wird in jeder Iteration um den Faktor C reduziert. Häufig ist C nahe 1, d.h. sehr langsame Konvergenz. Der folgende Satz gibt Auskunft, wann quadratische Konvergenz zu erwarten ist.

Satz 4.11: Sei D abgeschlossen, konvex; $x^* \in D$ ein FP von $\Phi(x)$ und $\Phi(x)$ sei in einer Umgebung $U(x^*) \subset D$ 2-mal stetig partiell differenzierbar und es sei

$$\frac{\partial \Phi(x^*)}{\partial x} = O \quad \text{die Nullmatrix} \tag{4.33}$$

Dann konvergiert (4.30) mindestens quadratisch gegen $x^* \in D$.

Beweis: Durch Taylorentwicklung folgt wegen (4.33) für jede der Koordinatenfunktionen von $\Phi(x)$

$$\Phi_i(x) - \Phi_i(x^*) = \frac{1}{2}\left(x - x^*\right)^T \nabla^2 \Phi_i(\xi^i)\left(x - x^*\right) \quad \forall x \in U(x^*) \quad \xi^i = x^* + \vartheta_i\left(x - x^*\right)$$

mit $0 < \vartheta_i < 1$ und für die Vektornorm $\|\bullet\|_\infty$ und zugehörige Matrixnorm gilt

$$\left| \ \Phi_i(x) - \Phi_i(x^*) \ \right| \le \frac{1}{2}\left\|x - x^*\right\|^2 \left\|\nabla^2 \Phi_i(\xi^i)\right\|_\infty \tag{4.34}$$

Wegen der 2-maligen stetigen Differenzierbarkeit von $\Phi(x)$ in $U(x^*) \subset D$ gilt

$$\max_{x \in U(x^*)} \left\|\nabla^2 \Phi_i(\xi^i)\right\|_\infty = M_i < \infty \quad i = 1,2,...,n.$$

Aus (4.34) folgt mit $M = \max M_i$ somit

$$\left\|x^{k+1} - x^*\right\|_\infty = \max_{i \in \{1,2,...,n\}} \left| \ \Phi_i(x^k) - \Phi_i(x^*) \ \right| \le \frac{1}{2} M \left\|x^k - x^*\right\|^2_\infty$$

d.h. quadratische Konvergenz.

4.5. Spezielle Verfahren für nichtlineare Gleichungssysteme

4.5.1. Die n-dimensionale Methode der Parallelsehnen

Einen wichtigen Indikator, wie spezielle Iterationsfunktionen $\Phi(x)$ aufgebaut sein sollten, liefert Satz 4.5. Wir betrachten das System

$$f(x) = 0, \quad f = (f_1, f_2, ..., f_n)^T : R^n \to R^n, \quad x \in R^n.$$

Bezeichnet A(x) eine reguläre (n,n)-Matrix, so sind die Nullstellen von f(x) identisch mit den Fixpunkten von

$$\Phi(x) = x - A(x) \cdot f(x),$$

denn aus $x = \Phi(x)$ folgt $-A(x) f(x) = 0$, d.h. $f(x) = 0$.

Fixpunkt-Iteration:

$$x^{k+1} = \Phi(x^k) = x^k - A(x^k) \cdot f(x^k) \tag{4.35}$$

Wählen wir speziell A(x)=A konstant, so erhalten wir die n-dimensionale Methode der Parallelsehnen

$$x^{k+1} = \Phi(x^k) = x^k - A \cdot f(x^k) \tag{4.36}$$

Setzen wir $x^{k+1} = x^k + \Delta x^k$, so ist der Korrekturvektor $s^k = \Delta x^k$ Lösung eines linearen Gleichungssystems

$$A^{-1} \Delta x^k = -f(x^k) \quad k = 0, 1, 2, ... \tag{4.37}$$

Zur Wahl der Matrix: Wir bemerken zunächst, dass in (4.37) statt A die Matrix A^{-1} fest gewählt werden kann. Ist f(x) in einer Umgebung der Lösung x^* differenzierbar und ist die Funktionalmatrix $\frac{\partial \Phi(x^*)}{\partial x}$ invertierbar, so wird (4.36) umso besser konvergieren (Satz 4.9), je kleiner der Spektralradius $\rho\left(\frac{\partial \Phi(x)}{\partial x}\right)$ in der Nähe von x^* ist. Wegen (4.36) gilt

$$\frac{\partial \Phi(x)}{\partial x} = E - A \frac{\partial f}{\partial x} \quad ,$$

d.h. man strebe an, $A^{-1} \approx \frac{\partial f(x^*)}{\partial x}$ zu wählen bzw. eine Näherung für diese Matrix zu bestimmen. Eine naheliegende Wahl ist $B = A^{-1} = \frac{\partial f(x^0)}{\partial x}$ und man löst die linearen Gleichungssysteme

$$B \Delta x^k = -f(x^k) \quad k = 0, 1, 2, ... \tag{4.38}$$

Nach m Schritten kann man die Matrixberechnung erneuern.

Vorteil: Die Gleichungssysteme (4.38) besitzen für k=0, 1, 2, ... die gleiche Matrix, d.h. ist die Dreiecksfaktorisierung von B einmal bestimmt, so brauchen nur die Dreiecks-systeme für verschiedene rechte Seiten gelöst werden.
Nachteil: Nur lineare Konvergenzordnung des Verfahrens.
Das Verfahren kann als vereinfachtes Newton-Verfahren interpretiert werden.

4.5.2. Mehrdimensionales Newton-Verfahren

Analog wie im Fall einer skalaren Gleichung kann man die nichtlineare vektorielle Gleichung $f(x) = 0,\ f: R^n \to R^n$ im Iterationspunkt $x = x^k$ linearisieren

$$f(x) = f(x^k) + J(x^k)(x - x^k) + R_1 .$$

Der neue Iterationspunkt ergibt sich dann aus der Lösung des linearen Ersatzproblems

$$0 = f(x^k) + J(x^k)(x - x^k).$$

Unter der Voraussetzung der Existenz der Inversen der Jacobi-Matrix $J(x^k) = \frac{\partial f(x^k)}{\partial x}$ von f(x) erhält man als Lösung

$$x = x^{k+1} = \Phi(x^k) = x^k - \left(J(x^k)\right)^{-1} \cdot f(x^k) \tag{4.39}$$

Praktische Realisierung: Wir setzen $x^{k+1} = x^k + s^k$ und $s = s^k$ ist Lösung des linearen Gleichungssystems

$$J(x^k)s\Delta x = -f(x^k) \quad k = 0, 1, 2,\ldots. \tag{4.40}$$

Konvergenz:

Wir zeigen die quadratische Konvergenz des Verfahrens. Wegen

$$\Phi(x) = x - \left(J(x)\right)^{-1} \cdot f(x)$$

folgt

$$J(x)\Phi(x) = J(x)x - f(x)$$

und durch Differentiation erhält man wegen $J(x) = \dfrac{\partial f(x)}{\partial x}$

$$\frac{\partial J(x)}{\partial x}\Phi(x) + J(x)\frac{\partial \Phi}{\partial x} = J(x) + \frac{\partial J(x)}{\partial x}x - J(x).$$

Setzt man nun $x = x^*$ ein und beachtet die Fixpunktgleichung $\Phi(x^*) = x^*$, so gilt

$$J(x^*)\frac{\partial \Phi(x^*)}{\partial x} = O_{n,n}.$$

und bei Invertierbarkeit von $J(x^*)$ folgt $\dfrac{\partial \Phi(x^*)}{\partial x} = O_{n,n}$. (Nullmatrix). Nach Satz 4.11 ist damit das Verfahren (4.39) lokal quadratisch konvergent.

Nachteil: In jedem Schritt muss $J(x^k)$ bestimmt werden und ein lineares Gleichungssystem (4.40) für s^k gelöst werden.

Bemerkung: Die Invertierbarkeit von $J(x^*)$, die auch für die eindeutige Lösbarkeit der Gleichungssysteme (4.40) zu fordern ist, bedeutet, dass $x = x^*$ eine einfache Nullstelle des Systems ist.

Modifikation des N-Verfahrens

Die Ausführung des N-Verfahrens erfordert die Programmierung der n Funktionen $f_i(x) \quad i = 1, 2,..., n$ des Gleichungssystems sowie die der $n \times n = n^2$ partiellen Ableitungen

$$\frac{\partial f_i(x)}{\partial x_j} \quad i, j = 1, 2,..., n.$$

Häufig sind die Ableitungen nicht analytisch erhältlich ($f_i(x)$ kompliziert, n zu groß bzw. $f_i(x)$ nur implizit gegeben). Dann kann man eine diskretisierte Variante des Verfahrens verwenden, welches Differenzennäherungen der Ableitungen verwendet. Die j-te Spalte der Jacobi-Matrix $J(x^k)$ wird dann näherungsweise durch den Vektor

$$\frac{1}{h}\left[f(x^k + he^j) - f(x^k)\right] \quad j = 1, 2,..., n$$

dargestellt. Hierbei bezeichnet e^j den j-ten Einheitsvektor des R^n. Die Gleichungen des Systems sind somit zusätzlich n-mal auszuwerten um eine Näherung der Jacobimatrix $J(x^k)$ zu erhalten.

Weitere praktische Probleme:

1.) Je größer n ist, um so schwieriger ist es einen Startpunkt x^0 zu finden, für den man Konvergenz erhält. Die Umgebung $U(x^*)$, aus der für einen beliebigen Startpunkt Konvergenz vorliegt, ist z.T. sehr klein. Die Wahl mehrerer Startpunkte ist oft nötig.
2.) Globalisierung des Verfahrens ist damit wichtig, z. B. Dämpfung des N-Schrittes durch eine Schrittweite $\lambda_k < 1$, wenn der Newton-Schritt zu groß ist.

Algorithmus:

Gegeben: $x^0 \in R^n$ (Startvektor), eps (Genauigkeit)

$k = 0, 1, .2, ...$ $(Iterationsschritte)$

- $f(x^k), J(x^k)$ *berechnen*

  $Test: Wenn \left\| f(x^k) \right\|_2 < eps$ *Abbruch mit* x^k *als Näherungslösung*

  sonst:

- *Gleichungssystem* $J(x^k)s = -f(x^k)$ *lösen; Lösung* $s = s^k$
- $x^{k+1} = x^k + \lambda_k s^k$ *mit* $\lambda_k \in \left\{1, \frac{1}{2}, \frac{1}{4}, ...\right\}$ *größter Wert, sodass* $\left\| f(x^{k+1}) \right\|_2 < \left\| f(x^k) \right\|_2$

Bem.: (1) Der Algorithmus kann in Punkten festlaufen, für die $\left\| f(x) \right\|_2$ ein lokales Minimum besitzt mit $\left\| f(x) \right\|_2 > 0$, d.h. es liegt dann keine Lösung des Gleichungssystems vor. Dies kann nur geschehen, wenn die Jacobimatrix $J(x^*)$ singulär wird.
(2) Die obige Schrittweitestrategie sichert nicht in jedem Fall Konvergenz, in der Literatur werden ausgefeiltere Strategien beschrieben.

4.6. NICHTLINEARE AUSGLEICHSRECHNUNG, GAUSS-NEWTON-ALGORITHMUS

Problemstellung: Betrachtet wird ein überbestimmtes nichtlineares Gleichungssystem vom Typ

$$\begin{aligned} r_1(x_1, x_2, ..., x_n) &= 0 \\ r_2(x_1, x_2, ..., x_n) &= 0 \\ \vdots \quad\quad\quad & \vdots \quad \vdots \quad bzw. \quad r(x) = 0 \quad r: R^n \to R^m \\ r_m(x_1, x_2, ..., x_n) &= 0 \end{aligned} \tag{4.41}$$

Das Problem ist im Fall m>n überbestimmt und wie auch im linearen Fall i.a. unlösbar. Statt einer exakten Lösung des Systems wird eine Ersatzlösung (Quadratmittellösung) gesucht, welche die Residuumsnorm minimiert:

Quadratmittellösung: $x^* \in R^n$ so gesucht, dass gilt

$$F(x) = \left\| r(x) \right\|_2^2 = \sum_{j=1}^{m} r_j(x)^2 \to \min_{x \in R^n}$$

Wichtiger Anwendungsfall: Parameterschätzung in nichtlinearen Modellen.

Gegeben: Modell eines Prozesses $y = M(t; x_1, x_2, ..., x_n)$ mit unbekannten Parametern x_i $(i = 1, 2, ..., n)$ sowie Messdatensätze (t_j, y_j) $j = 1, 2, ..., m$ $(m \geq n)$.

Gesucht: $x^* \in R^n$ so, dass in den Messpunkten t_j eine möglichst gute Übereinstimmung zwischen Messwert y_j und Modellwert $M(t_j; x_1, x_2, ..., x_n)$ besteht.

Man definiert dann Residuumsfunktionen durch die sogenannten Fehlergleichungen:

$$\begin{aligned} r_1(x_1, x_2,..., x_n) &= M(t_1; x_1, x_2,..., x_n) - y_1 \\ r_2(x_1, x_2,..., x_n) &= M(t_2; x_1, x_2,..., x_n) - y_2 \\ \vdots \qquad & \vdots \qquad\qquad \vdots \\ r_m(x_1, x_2,..., x_n) &= M(t_m; x_1, x_2,..., x_n) - y_m \end{aligned} \tag{4.42}$$

Gauss-Newton-Algorithmus:

Der Algorithmus ist eine spezielle Methode zur Minimierung des MkQ-Kriteriums F(x), wobei nur erste Ableitungen benötigt werden. Die Methode ist iterativ und erzeugt ausgehend von einem Startvektor x^0 eine Folge von Vektoren $x^k \quad (k = 0, 1, 2,...)$, die unter moderaten Voraussetzungen gegen eine Lösung x^* konvergiert. Jede Iteration besteht im wesentlichen aus drei Teilschritten:

(1) Durch Linearisierung der Residuumsfunktionen im Punkt $x = x^k$ erhält man für j=1,2,...,m

$r_j(x) = r_j(x^k) + \nabla r_j(x^k)^T (x - x^K) + O(\|x - x^k\|^2)$

bzw. in Vektorschreibweise

$r\ (x) = r(x^k) + J(x^k)\ \ (x - x^K) + O(\|x - x^k\|^2)$.

Dabei bezeichnet J(x) die (m,n)-Matrix der 1.Ableitungen der Residuumsfunktionen (Jacobi-Matrix von r(x))

$$J(x^k) = \frac{\partial r(x^k)}{\partial x} = \left(\frac{\partial r_j(x^k)}{\partial x_i} \right)_{i=1,2,..,n;\ j=1,2,..,m} = \begin{pmatrix} \nabla r_1(x^k)^T \\ \vdots \\ \nabla r_m(x^k)^T \end{pmatrix}_{(m,n)-Matrix} .$$

(2) Die Minimierung von $F(x) = \|r(x)\|_2^2$ wird ersetzt durch die lineare Ausgleichsaufgabe

$$\|r(x^k) + J(x^k)(x - x^k)\|_2^2 = \|r(x^k) + J(x^k)\ s\ \|_2^2 = \|-b + As\|_2^2 \to \min_{s \in R^n} \tag{4.43}$$

wobei $s = x - x^k$ gesetzt wird.

(3) Die Lösung von (4. 43) liefert i.a. eine Abstiegsrichtung $s = s^k$ bezüglich der die Funktion F(x) entlang des Strahls $x = x^k + \lambda s^k \quad (\lambda > 0)$ lokal abnimmt. Es ist die Schrittweite $\lambda > 0$ geeignet zu bestimmen.

Satz 4.12: Ist $s = s^k \neq 0$ Lösung von (4.43) und besitzt die Jacobimatrix $J(x^k)$ den vollen Spaltenrang n, dann gibt es eine Zahl $\bar{\lambda} > 0$ so, dass die Funktion

$$\varphi_k(\lambda) = F(x^k + \lambda s^k) = \|r(x^k + \lambda s^k)\|_2^2$$

streng monoton abnehmend ist für alle $\lambda : 0 < \lambda < \bar{\lambda}$.

Beweis: Wir zeigen, dass $\varphi_k(\lambda)$ für $\lambda > 0$ fallend ist. Es genügt zu zeigen, dass die Ableitung von $\varphi_k(\lambda)$ für $\lambda = 0$ negativ ist. Die Funktion $\varphi_k(\lambda)$ ist stetig differenzierbar nach λ, wenn alle Funktionen $r_j(x)$ stetig differenzierbar bezüglich x sind:

$$\varphi_k'(\lambda) = \frac{d}{d\lambda} r(x^k + \lambda s^k)^T r(x^k + \lambda s^k) = 2\left[\ J(x^k + \lambda s^k)s\ \right]^T r(x^k + \lambda s^k)$$

Damit gilt

$$\varphi_k'(0) = 2\left[\ J(x^k)s\ \right]^T r(x^k).$$

Der Vektor s ist Lösung des linearen Ausgleichsproblems (4.43), d.h. s erfüllt auch das Normalgleichungssystem

$$J(x^k)^T J(x^k)s = -J(x^k)^T r\ (x^k).$$

somit gilt

$$\varphi_k'(0) = 2s^T J(x^k)^T r(x^k) = -2s^T J(x^k)^T J(x^k)s = -2\left\|J(x^k)s\right\|_2^2 < 0, \qquad (4.44)$$

sofern $J(x^k)$ den vollen Rang n besitzt und $s = s^k \neq 0$ ist.

Das Resultat legt den folgenden Algorithmus nahe:

Gauss-Newton-Algorithmus:

Gegeben: $x^0 \in R^n$ (Startvektor), tol (Genauigkeit)

$k = 0, 1, .2, ...$ $(Iterationsschritte)$

- $r(x^k), J(x^k)$ *berechnen*

  $Test: Wenn \left\|J(x^k)^T r(x^k)\right\|_2 < tol$ *Abbruch mit* x^k *als Näherungslösung*

  sonst:

- (*Abstiegsrichtung* s^k *bestimmen*)

  Lineares Ausgleichsproblem $\left\|r(x^k) + J(x^k)s\right\|_2^2 \to \min_s$ *lösen*; *Lösung* $s = s^k$

  z.B. über das Normalgleichungssystem: $J(x^k)^T J(x^k)s = -J(x^k)^T r(x^k)$

  falls $\left\|J(x^k)^T r(x^k)\right\|_2 < tol$ *stop*

- (*Schrittweite* λ *bestimmen*):

  $x^{k+1} = x^k + \lambda_k s^k$ *mit* $\lambda_k \in \left\{1, \frac{1}{2}, \frac{1}{4}, ...\right\}$ *größter Wert, so dass* $\left\|r(x^{k+1})\right\|_2 < \left\|r(x^k\right\|_2$

Konvergenz:
- Bei Nullresiduum $r(x^*) = 0$ lokal quadratische Konvergenz;
- Bei kleinem Residuum lokal lineare Konvergenz;
- Bei großem Residuum kann häufig keine Konvergenz erreicht werden, es ist dann eine Modifikation des Algorithmus erforderlich.

Bem.: (1) Im Fall $m = n$ (Zahl der Modellparameter und der Datensätze sind gleich) geht das Fehlergleichungssystem (4.42) in ein reguläres nichtlineares Gleichungssystem über. Besitzt die Jacobimatrix $J(x)$ wie vorausgesetzt den vollen Rang, so ist sie regulär und dies auch in Umgebungen der Lösung. Das Verfahren entspricht dann dem Newton-Verfahren von Abschnitt 4.5.
(2) Bezüglich der Schrittweite gilt die gleiche Bemerkung wie zum Newton-Verfahren.

4.7. Iterative Lösung linearer Gleichungssysteme

Der numerische Aufwand zur direkten Lösung linearer Gleichungssysteme liegt in der Größenordnung von n^3 Operationen. Bei der Lösung sehr großer Systeme mit tausenden Variablen ist häufig ein iterativer Lösungsprozess auf der Basis geeigneter Fixpunktiterationen vorzuziehen, da bei ihnen pro Iterationsschritt nur eine Multiplikation vom Typ Matrix mal Vektor auftritt. Außerdem können Strukturen der Matrix besser ausgenutzt werden und Rundungsfehler können sich nicht akkumulieren.

Fixpunkt-Form: Gegeben ist das System Ax = b, A eine (n,n)-Matrix, $b \in R^n$. Man verwendet die Zerlegung A = N - P, wobei die Matrix N regulär sein soll und möglichst einfache Struktur besitzen soll. Dann gilt:

$$x = \Phi(x) = N^{-1}Px + N^{-1}b = Bx + c.$$

Fixpunkt-Iteration: $x^0 \in R^n$ gegeben;

$$x^{k+1} = \Phi(x^k) = Bx^k + c \tag{4.45}$$

Damit gilt

$$x^k = \Phi(x^{k-1}) = B^k x^0 + (E + B + B^2 + ... + B^{k-1})c \tag{4.46}$$

Ist $B = N^{-1}P$ eine konvergente Matrix (vgl. Abschnitt 2.3.) so gilt weiterhin

$$\lim_{k\to\infty} B^k = 0, \lim_{k\to\infty} \left(E + B + B^2 + ... + B^{k-1}\right) = (E - B)^{-1}.$$

Die rechte Seite von (4.46) ist für $k \to \infty$ somit unabhängig von k, d.h. sie stellt einen festen Vektor dar, der Grenzelement der Iterationsfolge ist:

$$\lim_{k\to\infty} x^k = (E - B)^{-1}c = x^*,$$

d.h. x^* erfüllt die Fixpunkt-Gleichung und ist wegen der vorausgesetzten Regularität von N auch Lösung von Ax = b.

Satz 4.13: Das Iterationsverfahren (4.45) zur Lösung von Ax = b konvergiert für jeden Startvektor $x^0 \in R^n$ genau dann gegen die Lösung x^*, wenn $B = N^{-1}P$ eine konvergente Matrix ist, d.h. wenn ihr Spektralradius kleiner als 1 ist.

Konvergenzcharakteristik:

Für x^* gilt die Fixpunkt-Gleichung

$$x^* = \Phi(x^*) = Bx^* + c.$$

Subtrahiert man diese Beziehung von der Iterationsgleichung $x^{k+1} = \Phi(x^k) = Bx^k + c$, so folgt für den Fehler $e^k = x^k - x^*$ die Beziehung

$$e^k = Be^{k-1} = B^k e^0.$$

Damit gilt: Die Iteration (4.45) konvergiert linear mit einer linearen Konvergenzrate $C = \rho(B) < 1$.

Spezielle Iterationsverfahren:

(a) Verfahren in Gesamtschritten (Jacobi-Verfahren)

Wenn $A = L + D + R$ die Zerlegung von A in den Diagonalanteil D und den unteren bzw. oberen Dreiecksanteil L bzw. R darstellt, wird

$$N = D = diag(a_{11}, a_{22}, ..., a_{nn}), \quad P = N - A = -(L + R)$$

gesetzt und man erhält die Iteration

$$x_i^{(k+1)} := \frac{1}{a_{ii}}(b_i - \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} x_j^{(k)}) \quad i = 1, 2, ..., n \tag{4.47}$$

Hinreichend für Konvergenz des Verfahrens ist, dass eine Norm der Verfahrensmatrix $B_G = -D^{-1}(L + R)$ kleiner als 1 ist. Dies ist z. B. immer im Fall der diagonalen Dominanz der Ausgangsmatrix A gegeben, d.h. A erfüllt die Bedingung

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}| \quad i = 1, 2, ..., n \ . \tag{4.48}$$

(b) Verfahren in Einzelschritten (Gauss-Seidel-Verfahren)

Wenn $A = L + D + R$ wieder die Zerlegung von A darstellt, wird N = D + L gesetzt und P = -R. Damit folgt die Iteration

$$x_i^{(k+1)} := \frac{1}{a_{ii}}(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}) \quad i = 1, 2, ..., n \tag{4.49}$$

Die Konvergenzmatrix $B_E = -(D + L)^{-1} R$ besitzt ebenfalls bei diagonaler Dominanz von A einen Spektralradius kleiner als Eins. Bei symmetrischen Matrizen liegt Konvergenz vor, wenn A positiv definit ist.

(c) SOR-Verfahren (Sequential Over-Relaxation)

Die Konvergenz des Iterationsprozesses wird durch einen zusätzlichen Parameter ω (Relaxationsparameter) gesteuert. Es wird die Zerlegung von A verwendet:

$$A = L + D + R = N - P \quad mit \quad N = \frac{1}{\omega} D + L, \quad P = \frac{1}{\omega}\big((1 - \omega)D - \omega R\big).$$

Damit erhält man als koordinatenweise Realisierung der Iteration

$$x_i^{(k+1)} := \frac{\omega}{a_{ii}}(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)}) + (1 - \omega) x_i^{(k)} \quad i = 1, 2, ..., n \tag{4.50}$$

Im Fall $\omega = 1$ erhält man das Verfahren in Einzelschritten, $\omega < 1$ wird als Unterrelaxation, $\omega > 1$ als Überrelaxation bezeichnet. Für den Spektralradius der Konvergenzmatrix $B = N^{-1} P$ gilt $\rho(B) \geq |1 - \omega|$. Das SOR-Verfahren kann somit nur für Relaxationsparameter aus dem Bereich $0 < \omega < 2$ konvergieren. Auch wenn die Verfahren aus (a) und (b) nicht konvergieren, kann man in vielen Fällen bei geeigneter Wahl von ω Konvergenz des SOR-Verfahrens erhalten. Für symmetrische, positiv definite Matrizen A konvergiert das SOR-Verfahren für $0 < \omega < 2$ immer. Das Verfahren kann durch Vorkonditionierung der Matrix beschleunigt werden.

5. Eigenwertprobleme symmetrischer Matrizen

5.1. Problemstellung, Ergebnisse aus der linearen Algebra

Eigenwertprobleme symmetrischer Matrizen treten in vielfältiger Weise in Anwendungen auf. Wir betrachten ein Beispiel:

Beispiel 1: geg.: $y''(x)+\mu\, y(x)=0, \quad y(a)=y(b)=0$

Randwertproblem für eine gewöhnliche Differentialgleichung mit einem Parameter μ (Sturm-Liouville-Problem)

ges.: Lösungsfunktion $y(x)$ in [a,b]

Zur numerischen Bestimmung der Lösungsfunktion diskretisieren wir das Intervall [a,b]:

1.) $x_i = a+ih,\ i=0,1,...,n+1$ mit $h=\dfrac{b-a}{n+1}$, d.h. $x_0=a,\ x_1=a+h,...,x_{n+1}=b$

2.) y_i bezeichne Näherungswerte der Lösungsfunktion $y(x)$ an den Stellen x_i, $y_0=y_{n+1}=0$ entsprechend der Randbedingungen.

3.) Die 2.Ableitung in der Differenzialgleichung wird durch einen Differenzenquotienten ersetzt

$$y''(x_i)\approx\frac{y'(x_{i+1})-y'(x_i)}{h}\approx\frac{1}{h}\left(\frac{y(x_{i+1})-y(x_i)}{h}-\frac{y(x_i)-y(x_{i-1})}{h}\right),$$

$$y''(x_i)\approx\frac{1}{h^2}\left(y_{i+1}-2y_i+y_{i-1}\right).$$

Die Diskretisierung der Differenzialgleichung in den Punkten $x_1,x_2,...x_n$ gibt somit die Näherungsgleichungen

$$\frac{1}{h^2}\left(y_{i+1}-2y_i+y_{i-1}\right)+\mu y_i=0 \quad i=1,2,...,n$$

Mit $y_0=y_{n+1}=0$ erhält man das lineare Gleichungssystem

$$Ay=\begin{pmatrix}2&-1&0&...&0\\-1&2&-1&...&0\\0&-1&2&...&0\\&&&...&\\0&0&...&-1&2\end{pmatrix}\begin{pmatrix}y_1\\y_2\\y_3\\...\\y_n\end{pmatrix}=\mu h^2\begin{pmatrix}y_1\\y_2\\y_3\\...\\y_n\end{pmatrix}$$

bzw.

$$\boxed{\left(A-\mu h^2E\right)y=0},$$

d.h. es entsteht ein Eigenwertproblem für die Matrix A mit Eigenwerten $\lambda=\mu h^2$.

Ergebnisse der linearen Algebra:

Vor.: A eine reelle symmetrische (n,n)-Matrix

1. Die Eigenwerte λ_i und Eigenvektoren x^i von A sind Lösungen von $\left(A-\lambda E\right)x=0$.
2. Die Eigenwerte λ_i sind Nullstellen der charakteristischen Gleichung $p_n(\lambda)=\det\left(A-\lambda E\right)=0$.
3. Die Eigenwerte $\lambda_1,\lambda_2,...,\lambda_n$ von A sind reell, und es gibt ein System $u^1,u^2,...,u^n$ von zugehörigen Eigenvektoren, die paarweise orthogonal sind und normiert.
4. Zwei Matrizen A,B heißen ähnlich, wenn es eine reguläre Matrix T gibt mit $T^{-1}AT=B$.

Ähnliche Matrizen haben das gleiche charakteristische Polynom und die gleichen Eigenwerte. Ist x EV von A, so ist $y = T^{-1}x$ EV von B zum gleichen Eigenwert.

5. Ist $u^1, u^2, ..., u^n$ ein System orthonormierter Eigenvektoren entsprechend 3., so gilt für
$T = U = \left(u^1, u^2, ..., u^u\right): T^{-1} = T^T$, d.h. T bzw. U sind orthogonal und
$U^T AU = \Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n)$.

Die Eigenschaft 5. wird als diagonale Ähnlichkeit der Matrizen A und Λ bezeichnet. Matrizen, die n linear unabhängige EV besitzen, sind diagonalähnlich. Außer den symmetrischen Matrizen sind auch viele unsymmetrische Matrizen diagonalähnlich. Matrizen, die keine n linear unabhängigen EV besitzen werden als defektiv bezeichnet. Sie sind ausnahmslos unsymmetrisch und besitzen mehrfache Eigenwerte. Für die numerische Berechnung der EW und EV wird wesentlich die Eigenschaft 4. der Ähnlichkeit ausgenutzt.

5.2. Bestimmung einzelner Eigenwerte und Eigenvektoren

Die Verfahren zur Lösung des Eigenwertproblems sind iterative Verfahren, da sowohl EW als auch EV nicht in endlich vielen Schritten berechnet werden können.
Man unterscheidet im Wesentlichen 2 Gruppen von Verfahren:

(a) Verfahren zur Bestimmung einzelner EW und EV;
(b) Verfahren zur Bestimmung aller EW und EV.

Wir betrachten zunächst 2 einfache Verfahren der Gruppe (a). Sie besitzen Anwendung vor allem bei großen Matrizen, für die i.a. nur wenige EW zu bestimmen sind.

5.2.1. Vektoriteration (Von-Mises-Iteration, Potenzmethode)

Anwendung: Berechnung des betragsgrößten EW und des entsprechenden EV; auch für nichtsymmetrische Matrizen.

Voraussetzung: A ist diagonalähnlich, d.h. $\lambda_1, \lambda_2, ..., \lambda_n$ sind EW und $u^1, u^2, ..., u^n$ ein zugehöriges System normierter EV (linear unabhängig, bei symmetrischen Matrizen paarweise orthogonal). Weiterhin gelte $|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq ... \geq |\lambda_n|$.

Iterationsvorschrift: $x^0 \in R^n$ als Startvektor gewählt

$$\boxed{x^{k+1} := Ax^k \quad k = 0,1,2,...} \tag{5.1}$$

Analyse des Verfahrens: Wir zerlegen $x^0 \in R^n$ als Linearkombination der Eigenvektoren $u^1, u^2, ..., u^n$:

$$x^0 = c_1 u^1 + c_2 u^2 + ... + c_n u^n$$

Wegen $Au^i = \lambda_i u^i$ gilt dann

$$x^1 = Ax^0 = c_1 Au^1 + c_2 Au^2 + ... + c_n Au^n = c_1\lambda_1 u^1 + c_2\lambda_2 u^2 + ... + c_n\lambda_n u^n$$

$$x^2 = Ax^1 = c_1\lambda_1 Au^1 + c_2\lambda_2 Au^2 + ... + c_n\lambda_n Au^n = c_1{\lambda_1}^2 u^1 + c_2{\lambda_2}^2 u^2 + ... + c_n{\lambda_n}^2 u^n$$

$$x^k = Ax^{k-1} = c_1\lambda_1^k u^1 + c_2\lambda_2^k u^2 + ... + c_n\lambda_n^k u^n$$

$$\boxed{x^k = \lambda_1^k \left[c_1 u^1 + c_2 \left(\frac{\lambda_2}{\lambda_1}\right)^k u^2 + ... + c_n \left(\frac{\lambda_n}{\lambda_1}\right)^k u^n \right]} \tag{5.2}$$

Wegen der Voraussetzung der Dominanz von λ_1 streben für $k = 0,1,2,...$ die Potenzen $\left(\frac{\lambda_i}{\lambda_1}\right)^k \to 0$ für $k \to \infty$. Damit gilt $x^k \cdot \lambda_1^{-k} \to c_1 u^1$,d.h. der Iterationsvektor x^k nähert sich der Richtung des EV u^1. Die Beträge von x^k werden für $|\lambda_1| < 1$ gegen Null streben, für $|\lambda_1| > 1$ unbeschränkt wachsen, daraus erwachsen numerische Probleme.

Abhilfe: Normierung von x^k , dies liefert gleichzeitig Näherungen für den Eigenwert λ_1, denn aus $Ax = \lambda_1 x$ folgt für $\|x\|_2 = \sqrt{x^T x} = 1$

$$x^T Ax = \lambda_1 x^T x = \lambda_1$$

Modifizierte Vektoriteration:

$(S0)$ *Wähle* $x^0 \in R^n$, $\|x^0\| = 1$, $k = 0$
$(S1)$ $w^k := Ax^k$
$(S2)$ $\alpha_k := x^{kT} w^k$; $x^{k+1} := \frac{w^k}{\alpha_k}$; $\rho_k := x^{kT} w^k$ $(= x^{kT} Ax^k)$; $k := k+1$ *goto* $(S1)$

Konvergenzuntersuchung:
Mit $\varphi_k = \angle(u^1, x^k)$ bezeichnen wir den Winkel zwischen den EV u^1 und der Näherung x^k. Der "Abstand" von u^1 und x^k wird mit $\tan(\varphi_k)$ gemessen (d.h. sind u^1 und x^k senkrecht, dann ist der Abstand ∞, sind u^1 und x^k parallel, dann ist der Abstand 0).

Satz 5.1: Vor.: λ_1 dominierender EW von A und u^1 EV.
Startvektor $x^0 \in R^n$ erfülle die Bedingung $\sigma = u^{1T} x^0 > 0$.
Für die Folge $\{x^k\}$ gilt mit $q = \left|\frac{\lambda_2}{\lambda_1}\right| < 1$

$$0 \le \tan(\varphi_k) \le q \tan(\varphi_{k-1}) \le q^k \tan(\varphi_0) = q^k \frac{\sqrt{1-\sigma^2}}{\sigma}$$

und für die Folge $\{\rho_k\}$

$$|\rho_k - \lambda_1| \le 2\|A\|_2 \tan^2(\varphi_k) \le 2\|A\|_2 \frac{1-\sigma^2}{\sigma}\left(q^2\right)^k$$

Bew.: Kielbasinski/Schwetlick, S.376 ff.

Bemerkungen:
1.) Der Tangens des Winkels zwischen u^1 und x^k verringert sich in jeder Iteration um den Faktor q, d.h. es liegt bezüglich der Richtung des EV lineare Konvergenz mit der Konvergenzrate q vor. Der Abstand ρ_k von λ_1 verringert sich um den Faktor q², d.h. bezüglich des EW lineare Konvergenz mit der linearen Konvergenzrate q² . Die EW-Näherungen konvergieren somit schneller als die EV-Näherungen.

2.) Satz 5.1 gilt auch für $\sigma = u^{1T} x^0 < 0$, es wird nur $\neq 0$ benötigt, d.h. x^0 nicht orthogonal zu u^1. Dies wird praktisch durch Rundungsfehler immer erreicht.

3.) Wenn mit $\lambda_1 = \lambda_2$ ein doppelter Eigenwert als dominanter EW auftritt und $|\lambda_2| > |\lambda_3|$ gilt, so erzeugt der Algorithmus eine Folge $\{x^k\}$, die mit Konvergenzfaktor $q = \left|\frac{\lambda_3}{\lambda_1}\right| < 1$ gegen irgendeinen normierten EV $u^1 \in L(\lambda_1)$ konvergiert, wobei $L(\lambda_1)$ der zu $\lambda_1 = \lambda_2$ gehörende zweidim. Eigenraum ist. Wird die Iteration mit einem anderen x^0 gestartet, so konvergiert sie i.a. immer gegen einen normierten EV $y \in L(\lambda_1)$, der unabhängig von u^1 ist. Ein Orthogonalisierungsschritt (vgl. Lin. Algebra, Schmidtsches Orthogonalisierungsverfahren)

$$z = y - (u^{1T} y)u^1, \quad u^2 = \frac{z}{|z|}$$

liefert dann zwei orthogonale Vektoren u^1, u^2, die $L(\lambda_1)$ aufspannen.

4.) Sind bereits λ_1, u^1 bestimmt, so kann für $|\lambda_1| > |\lambda_2|$ der nächste EW und EV durch Deflation bestimmt werden. Wegen der Gültigkeit der EW-Zerlegung von A (vgl. Lin. Algebra)

$$A = \lambda_1 u^1 u^{1T} + \lambda_2 u^2 u^{2T} + ... + \lambda_n u^n u^{nT}$$

folgt, dass die reduzierte Matrix

$$\overline{A} = A - \lambda_1 u^1 u^{1T} = \lambda_2 u^2 u^{2T} + ... + \lambda_n u^n u^{nT}$$

den dominierenden EW λ_2 besitzt. Die Prozedur ist also auf $\overline{A}$ anwendbar. Numerisch ist dies nur sinnvoll, wenn λ_1, u^1 mit hoher Genauigkeit berechnet wurden.

Abbruchbedingung: Die Frage, wann die Iteration abgebrochen werden sollte, kann mit Satz 5.6 (Formel (5.16)) entschieden werden. Der relative Fehler $\varepsilon > 0$ sei vorgegeben. Man führe die modifizierte Iteration so lange aus, bis gilt

$$\left\| Ax^k - \rho_k x^k \right\|_2 \le \varepsilon \left\| Ax^k \right\|_2 \qquad (5.3)$$

Dann gilt wegen $cond_2(T) = cond_2(u^1, u^2, ..., u^n) = 1$ für den relativen Fehler von λ_1

$$\left| 1 - \frac{\rho_k}{\lambda_1} \right| = \left| \frac{\lambda_1 - \rho_k}{\lambda_1} \right| \le \varepsilon .$$

Hier müssen wir voraussetzen, dass $\varepsilon > 0$ so klein gewählt ist, dass ρ_k näher an λ_1 liegt als an λ_2.

5.2.2. Inverse Iteration nach Wieland

Der Nachteil der Vektoriteration besteht vor allem darin, dass nur der betragsgrößte EW und der zugehörige EV bestimmt werden. Ihr Anwendungsbereich ist damit stark eingeschränkt. Oft will man einen anderen EW (z.B. den betragskleinsten bestimmen).

Anwendung der inversen Iteration: Verbesserung einer bekannten Näherung μ eines beliebigen EW λ von A

Methode: Verschiebung des Spektrums der Matrix A. Es seien $\lambda_1, \lambda_2, ..., \lambda_n$ die EW von A und $u^1, u^2, ..., u^n$ ein System linear unabhängiger normierter Eigenvektoren. Dann besitzt die Matrix

$$\bar{A} = A - \mu E \tag{5.4}$$

die EW $\bar{\lambda}_1 = \lambda_1 - \mu, \bar{\lambda}_2 = \lambda_2 - \mu, ..., \bar{\lambda}_n = \lambda_n - \mu$.

Damit kann jeder EW λ_j von A zum betragskleinsten von $\bar{A} = A - \mu E$ gemacht werden, wenn $\mu \approx \lambda_j$ gewählt wird. Dann gilt aber: $p_j = \bar{\lambda}_j^{-1} = \dfrac{1}{\lambda_j - \mu}$ ist der betragsgrößte Eigenwert von $\bar{A}^{-1} = (A - \mu E)^{-1}$ und der zu λ_j gehörende normierte Eigenvektor u^j ist der zu p_j gehörende Eigenvektor dieser Matrix. Wir behandeln nun das Problem der Bestimmung von p_j mit Hilfe der Vektoriteration. Die Folge $\{x^k\}$ sei erzeugt gemäß

$$\boxed{x^{k+1} := \bar{A}^{-1} x^k = (A - \mu E)^{-1} x^k \quad k = 0,1,2,...}$$

bzw. aus

$$(A - \mu E) x = x^k \tag{5.5}$$

d.h. in jedem Schritt ist zur Bestimmung von x^{k+1} ein lineares Gleichungssystem zu lösen.

<u>Konvergenzanalyse:</u> Wegen (5.5) ist in (5.2) stets λ_1 durch $p_j = \bar{\lambda}_j^{-1} = \dfrac{1}{\lambda_j - \mu}$ zu ersetzen. Wenn μ eine Näherung für λ_j ist, so gilt

$$x^k = \frac{1}{(\lambda_j - \mu)^k} \left[c_j u^j + \sum_{i \neq j} c_i \left(\frac{\lambda_j - \mu}{\lambda_i - \mu} \right)^k u^i \right]$$

und die Konvergenz wird linear sein mit der linearen Konvergenzrate

$$q = \max_{i \neq j} \left| \frac{\lambda_j - \mu}{\lambda_i - \mu} \right| < 1, \tag{5.6}$$

Wobei q umso kleiner ist, je näher μ an λ_j liegt.

<u>Bestimmung einer Näherung μ:</u> Wegen $Au = \lambda u$ muss für EW λ und EV u gelten:

$$u^T A u = \lambda u^T u = \lambda |u|^2$$

bzw.

$$\lambda = \frac{u^T A u}{u^T u}$$

Damit gilt: Ist x eine Näherung für den EV u, so wählt man als Eigenwertnäherung μ

$$\mu = \frac{x^T A x}{x^T x} \quad \text{(Rayleigh-Quotient)} \tag{5.7}$$

Inverse Iteration mit Rayleigh-Quotient:

$(S0)$ *Wähle* $x^0 \in R^n$, $\|x^0\| = 1$, $k = 0$

$(S1)$ $\rho_k := x^{kT} A x^k \ (= \mu)$

$(S2)$ *Löse das Gleichungssystem* $(A - \rho_k E) w := x^k$; $\alpha_k := \|w\|_2$; $x^{k+1} := \frac{w}{\alpha_k}$;

$k := k + 1$ *goto* $(S1)$

Konvergenz ist gesichert, wenn die Matrix $A - \rho_k E$ nicht für ein ρ_k singulär wird (dann ist ρ_k exakter EW). Bei Konvergenz strebt ρ_k gegen einen EW λ_j von A, die Konvergenzordnung ist kubisch und x^k strebt gegen den entsprechenden normierten EV. Rundungsfehler verhindern, dass die Matrix $A - \rho_k E$ singulär wird und verbessern eher die Konvergenz.

Abbruch: Es sei $\varepsilon > 0$ als Genauigkeit vorgegeben in der Größenordnung von $\varepsilon \geq eps \cdot \sqrt{n} \|A\|_\infty$ ($\|A\|_\infty$ ...Zeilensummennorm). Wenn gilt $\varepsilon * \alpha_k > 1$, dann Abbruch, das Paar (ρ_k, x^k) ist akzeptables Eigenpaar in dem Sinn, dass es exaktes Eigenpaar einer benachbarten Matrix $A + \delta A$ ist mit $\|\delta A\|_\infty \leq \varepsilon \|A\|_\infty$.

Bem.:
1. Schritt (S2) mittels Gauß-Algorithmus und Pivotisierung durchführen, d.h. LR-Zerlegung der Matrix bestimmen $P(A - \mu E) = LR$ und damit das Gleichungssystem lösen. Für x^0 wird i.a. einer der Einheitsvektoren gewählt.
2. Zur Berechnung des betragskleinsten EW von A (falls ≠0) ist $\mu = 0$ zu setzen und die Vektorfolge $\{x^k\}$ zu bilden mit $A x^{k+1} = x^k$, die Folge $\{\rho_k\}$ erzeugt dann eine Näherung für den EW.

5.3. Das Jacobi-Verfahren

Anwendung: Berechnung aller EW und EV einer symmetrischen Matrix.

Methode: Die Matrix wird durch Ähnlichkeitstransformationen iterativ auf Diagonalform transformiert, indem die Summe der Quadrate der Nichtdiagonalelemente fortlaufend verkleinert wird.

Def. 5.2: Die Matrix

$$G_{pq} = \begin{pmatrix} 1 & 0 & \dots & \dots & 0 \\ 0 & 1 & & & \\ & & c & s & \\ & & -s & c & \\ 0 & 0 & \dots & \dots & 1 \end{pmatrix} \begin{matrix} \\ \\ p-te\ Zeile \\ q-te\ Zeile \\ \\ \\ p \quad q \end{matrix}, \quad c^2 + s^2 = 1$$

die sich von der Einheitsmatrix E nur in 4 Elementen unterscheidet, heißt Rotationsmatrix oder Givens-Drehungsmatrix. Im Fall p = q gilt $G_{pq} = E$.

Transformationsschritt: $A_1 = A$ setzen. Die Matrix A_k wird in $A_{k+1} = \overline{A}$ überführt durch $A_{k+1} = \overline{A} = G_k{}^T A_k G_k$. Die Matrix $G_k = G_{pq}$ ist eine Orthogonalmatrix vom Typ der Givens-Drehung und die Transformation ändert nur die Elemente in den Zeilen p,q bzw. den Spalten p,q. Wegen der Symmetrie von A_k ist A_{k+1} wieder symmetrisch, so dass nur z.B. die Zeilenelemente der Zeilen p,q umzurechnen sind:

Es ergeben sich folgende Umrechnungsformeln:

$$\overline{a}_{pp} = c^2 a_{pp} - 2cs\, a_{pq} + s^2 a_{qq},$$
$$\overline{a}_{qq} = s^2 a_{pp} + 2cs\, a_{pq} + c^2 a_{qq},$$
$$\overline{a}_{pq} = cs(a_{pp} - a_{qq}) + (c^2 - s^2) a_{pq},$$
$$\text{für } i = 1,2,...,n \quad (i \neq p,q): \quad \overline{a}_{ip} = ca_{ip} - sa_{iq}, \quad \overline{a}_{iq} = sa_{ip} + ca_{iq}$$

Bewertung: Als Maß für die Abweichung der Matrix A bzw. der transformierten Matrizen von der Diagonalform wird die Summe der Quadrate der Nichtdiagonalelemente verwendet:

$$w(A) = \sum_{\substack{i,j=1 \\ i<j}}^{n} a_{ij}^2$$

Zielstellung: In der Transformationsmatrix $G_k = G_{pq}$ die Werte c,s so wählen, dass eine Reduktion der Bewertung der Matrix erfolgt, d.h. es gilt

$$w_{k+1} = w(A_{k+1}) = w(\overline{A}) < w(A_k) = w_k .$$

Wegen $w(\overline{A}) = w(A\ ) + (\overline{a}_{pq}{}^2 - a_{pq}^2)$ wird die größte Reduktion der Bewertung erreicht, wenn $\overline{a}_{pq}{}^2 = 0$ ist und a_{pq}^2 der maximale Summand in $w(A)$ ist.

Varianten des Verfahrens, Bestimmung der Pivotindizes p,q:

(a) Klassisches Jacobi-Verfahren: Es wird p,q so gewählt, dass gilt

$$a_{pq}^2 = \max_{i,j \in \{1,2,...,n\}\ i<j} a_{ij}^2 \quad \text{(Maximalpivot)}.$$

Das Verfahren erfordert in jedem Schritt $N = \frac{n}{2}(n-1)$ Vergleichsoperationen.

(b) Schwellenwert-Jacobi-Verfahren: p,q durchlaufen jeweils einen vollen Zyklus der Indizes i =1,..,n; j = i+1,...,n.

Sobald gilt $a_{ij}^2 > \frac{w_k}{N}$ (Schwellenwert ist Mittelwert der a_{ij}^2) wird eine Rotation zum Annullieren von a_{ij} ausgeführt, andernfalls ergibt das Annullieren von a_{ij} zu geringen Gewinn.

Konvergenz: Bei beiden Varianten gilt $w_{k+1} = w_k - a_{pq}^2 \leq (1 - \frac{1}{N}) w_k .,$

d.h. es liegt mindestens lineare Konvergenz der Folge $\{w_k\}$ der Bewertungen gegen Null vor mit der linearen Konvergenzrate C=1-1/N. Praktisch kann sogar quadratische Konvergenz nachgewiesen werden.

Rotationsschritt: Ein Hauptschritt des Verfahrens besteht aus einer Jacobi-Rotation, welche das Annullieren des Pivotelementes $a_{pq} \neq 0$ und die Umrechnung der Matrix zum Ziel hat.

1. Annullieren von $a_{pq} \neq 0$: Wir setzen t = c/s, dann folgt aus

$$\bar{a}_{pq} = cs(a_{pp} - a_{qq}) + (c^2 - s^2)a_{pq} = 0 \text{ die quadratische Gleichung}$$

$$t^2 + 2\delta t - 1 = 0 \quad mit \quad \delta = \frac{a_{qq} - a_{pp}}{2a_{pq}}$$

Die Größen s,c können dann stabil berechnet werden mit Hilfe der Beziehungen

$$\rho = |\delta| + \sqrt{1 + \delta^2}$$

$$t = \begin{cases} \rho^{-1} & für \quad \delta \geq 0 \\ -\rho^{-1} & für \quad \delta < 0 \end{cases}$$

$$c = \frac{1}{\sqrt{1+t^2}}, \quad s = ct, \quad \tau = \frac{s}{1+c}$$

2. Transformation der Matrix entsprechend der Beziehung $A_{k+1} = \bar{A} = G_k{}^T A_k G_k$. Für die Elemente der geänderten Zeilen und Spalten mit Index p,q gilt:

$$\begin{aligned} &\bar{a}_{pp} = a_{pp} - t\,a_{pq}, \\ &\bar{a}_{qq} = a_{qq} + t\,a_{pq}, \\ &\bar{a}_{pq} = 0, \\ &für\ i = 1,2,...,n \quad (i \neq p,q): \quad \bar{a}_{ip} = a_{ip} - s(a_{iq} + \tau a_{ip}), \quad \bar{a}_{iq} = a_{iq} + \tau(a_{ip} + \bar{a}_{ip}) \end{aligned} \tag{5.8}$$

Die Bewertung wird in der Form $w_{k+1} = w_k - a_{pq}^2$ umgerechnet.

Berechnung der Eigenvektoren: Zur Berechnung der normierten Eigenvektoren kann das Verfahren erweitert werden. Wegen

$$A_k = V_k{}^T A V_k \to \Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n) \text{ für } k \to \infty$$

gilt bei Konvergenz die Beziehung $V_k \to V$ bzw. $V^T A V = \Lambda$, und da V eine Orthogonalmatrix ist gilt $AV = \Lambda V$. Die Spalten von V sind somit normierte Eigenvektoren zu den Eigenwerten λ_i als Hauptdiagonalelemente der Matrix Λ. Setzt man $V_1 = E$, so gilt $V_{k+1} = V_k G_k$. Bei dieser Umrechnung von V_k ändern sich nur die p-te und q-te Spalte, so dass die Elemente von V_k nach den Formeln (14) berechnet werden können.

$$für\ i = 1,2,...,n: \quad \bar{v}_{ip} = v_{ip} - s(v_{iq} + \tau v_{ip}), \quad \bar{v}_{iq} = v_{iq} + \tau(v_{ip} + \bar{v}_{ip})$$

5.4. REDUKTION AUF TRIDIAGONALFORM

Anwendung: Berechnung aller Eigenwerte einer symmetrischen Matrix. Die Matrix wird zunächst durch eine Ähnlichkeitstransformation $\bar{A} = T^{-1} A T$ auf Tridiagonalgestalt transformiert, in dieser Form sind die nachfolgenden Faktorisierungsalgorithmen effektiv.

Das HOUSEHOLDER-Verfahren auf Tridiagonalform (1958)

Die symmetrische Matrix A kann durch n-2 Householder-Transformationen

$$H_k: \ A^{(k+1)} = H_k{}^{-1} A^{(k)} H_k \quad k = 1,2,...,n-2; \qquad A^{(1)} = A$$

auf Tridiagonalform gebracht werden. Da die Matrizen $A^{(k)}$ *und* $A^{(k+1)}$ ähnlich sind, besitzen sie die gleichen EW. Die Transformationsmatrizen H_k sind orthogonal, damit tritt keine Konditionsverschlechterung ein und alle Matrizen $A^{(k)}$ bleiben symmetrisch.

HOUSEHOLDER-Matrix (vgl. Abschnitt 3.8.1):

$$\boxed{H = E - 2uu^T, u \in R^n, \|u\|_2 = 1.}$$

Eigenschaften: a) H ist symmetrisch: $H^T = H$
b) H ist orthogonal $H^T = H^{-1}$
c) H ist involutorisch $HH = H^2 = E$ (folgt aus a) und b)).

Transformation:

Start: $A^{(1)} = A$

k-ter Schritt: Es werden in der k-ten Spalte und Zeile von $A^{(k)}$ Nullen erzeugt. Die Matrix $A^{(k)}$ sei also von der Form

$$A^{(k)} = \begin{pmatrix} * & * & 0 & 0\ldots & 0 \\ * & * & * & 0\ldots & 0 \\ 0 & * & (T^{(k)}) & * & 0 \\ 0 & 0 & * & * & a^{kT} \\ & & & a^k & B^{(k)} \\ & & & & \end{pmatrix} \quad mit\ a^k \in R^{n-k} \tag{5.9}$$

(a) Ist der Vektor $a^k = 0$, so sind die geforderten Nullen vorhanden, wir setzen $H_k = E$, sonst

(b) $H_k = \begin{pmatrix} E_k & 0 \\ 0 & \overline{H}_k \end{pmatrix}$

Die (n-k, n-k)-Matrix $\overline{H}_k$ wird so bestimmt, dass gilt

$$\overline{H}_k a^k = \rho_k e^1 = \begin{pmatrix} \rho_k \\ 0 \\ \ldots \\ 0 \end{pmatrix} \in R^{n-k}, \quad a^k = \begin{pmatrix} a_{k+1,k} \\ a_{k+2,k} \\ \ldots \\ a_{n,k} \end{pmatrix}.$$

Wir setzen wie in 3.8.1

$$\overline{H}_k = E - \frac{1}{\kappa_k} u^k u^{kT}, \quad wobei\ s_k = \left\|a^k\right\|_2^2\ ist\ und\ \kappa_k = \frac{1}{2} u^{kT} u^k = s_k - \rho_k a_{k+1,k}$$

$$u^k = a^k - \rho_k e^1,\ \rho_k = \begin{cases} \sqrt{s_k} & für\ a_{k+1,k} < 0 \\ -\sqrt{s_k} & für\ a_{k+1,k} \geq 0 \end{cases}$$

ρ_k wird in dieser Form gewählt, um Auslöschung zu vermeiden. (Vergleiche auch Kapitel 3.8.1)

Praktische Durchführung: Die Matrix H_k wird nicht explizit berechnet.

$$A^{(k+1)} = H_k A^{(k)} H_k = \begin{pmatrix} * & * & 0 & 0... & 0 \\ * & * & * & 0... & 0 \\ 0 & * & (T^{(k)}) & * & 0 \\ 0 & 0 & * & * & \rho_k e^{1T} \\ & & & \rho_k e^1 & \overline{B}^{(k)} \\ & & & & \end{pmatrix} \tag{5.10}$$

mit

$$\overline{B}^{(k)} = \overline{H}_k B^{(k)} \overline{H}_k \tag{5.11}$$

Man kann nun zeigen, dass die Matrix (5.11) wie folgt effektiv berechnet werden kann

$$\overline{B}^{(k)} = B^{(k)} - (p^k u^{kT} + u^k p^{kT}) \tag{5.12}$$

mit dem Vektor u^k wie oben festgelegt und

$$w^k = \frac{1}{\kappa_k} B^{(k)} u^k, \quad p^k = w^k - \frac{u^{kT} w^k}{2\kappa_k} u^k .$$

Es gilt dann:

Satz 5.3: Nach (n-2) Schritten ist

$\overline{A} = A^{(n-1)} = H_{n-2}...H_2 H_1 A H_1 H_2 ... H_{n-2}$

eine Tridiagonalmatrix und $Q = H_1 H_2 ... H_{n-2}$ eine Orthogonalmatrix.

Aufwand: $\sim \frac{2}{3} n^3$ ops (Add.+Mult.), wenn (5.12) verwendet wird)

Bemerkungen:

1.) Der Algorithmus kann wegen der Symmetrie von A und der transformierten Matrizen $A^{(k)}$ auf den Platz des oberen Dreiecks durchgeführt werden. Unter der Hauptdiagonale werden die Vektoren u^k abgespeichert, die alle Informationen über die H_k enthalten. Diese Informationen sind nötig für die Berechnung der Eigenvektoren:
Ist y ein EV von $\overline{A}$, so ist $x = Qy$ EV von A.

2.) Die Anwendung der Householder-Transformation auf unsymmetrische Matrizen transformiert diese auf obere HESSENBERG-Form.

$$\overline{A} = H_{n-2}...H_2 H_1 A H_1 H_2 ... H_{n-2} = \begin{pmatrix} * & * & * & *... & * & * \\ * & * & * & *... & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ ... & & & * & * & * \\ 0 & 0 & ... & 0 & * & * \end{pmatrix}$$

Im k-ten Schritt werden in der k-ten Spalte die entsprechenden Nullen erzeugt, die aber aufgrund der Nichtsymmetrie von A nicht auch in der Zeile auftreten müssen. Die obigen Formeln sind dann zu modifizieren.

5.5. FAKTORISIERUNGSMETHODEN

Moderne Methoden zur Berechnung aller EW (und EV) einer mittleren Matrix beruhen auf Faktorisierungen der Matrix. Bei voll besetzten Matrizen ist der Aufwand, z.B. in jeder Iteration die Dreiecksfaktorisierung zu bestimmen, außerordentlich hoch. Darum wird die Matrix zuerst durch Ähnlichkeitstransformationen in Tridiagonalform (symm. Matrizen) bzw. obere Hessenberg-Form (unsymm. Matrizen) überführt. Das Faktorisierungsverfahren wird dann auf die reduzierte Matrix angewendet.

5.5.1. Das LR-Verfahren (Rutishauser 1958)

Beschreibung des Verfahrens:

$(S0)$ $A_1 := A; \quad k = 1$

$(S1)$ *Zerlege* A_k *in ein Produkt von Dreiecksmatrizen* $(LR-Zerlegung)$

$$A_k = L_k R_k, \quad L_k = \begin{pmatrix} 1 & 0 & ... & 0 \\ * & 1 & ... & 0 \\ * & * & ... & 0 \\ * & * & ...* & 1 \end{pmatrix}, \quad R_k = \begin{pmatrix} * & * & ... & * \\ 0 & * & ... & * \\ 0 & 0 & ... & * \\ 0 & 0 & ...0 & * \end{pmatrix}$$

$(S2)$ *Bilde das Produkt* $A_{k+1} = R_k L_k;$

$k := k+1 \quad goto\ (S1)$

Der Schritt (S1) kann mit Hilfe des Gauß-Algorithmus ohne Pivotisierung durchgeführt werden. Damit ist klar, dass das Verfahren versagt, sobald eines der ersten n-1 Hauptdiagonalelemente von A_k gleich 0 ist. Dieser entscheidende Nachteil des LR-Verfahrens wird mit dem QR-Verfahren überwunden.

Aufwand: Ist $A = A_1$ eine Tridiagonalmatrix, so ist in A_1 in jedem Hauptschritt der Gauß-Elimination nur 1 Subdiagonalelement zu annullieren, d.h. L_1, R_1 besitzen in jeder Spalte max. zwei Elemente ungleich Null. Das Produkt $R_1 * L_1$ ist dann wieder eine Tridiagonalmatrix . Aufwand in jedem Schritt: O(n).

Analyse des Verfahrens: Wegen $A_k = L_k R_k$ gilt $R_k = L_k^{-1} A_k$ und damit

$$A_{k+1} = R_k L_k = L_k^{-1} A_k L_k$$

d.h. die Matrizen A_k, A_{k+1} in jedem Schritt sind ähnlich und besitzen somit die gleichen EW. Unter zusätzlichen Voraussetzungen konvergiert die Folge $\{A_k\}$ gegen eine obere Dreiecksmatrix mit den Eigenwerten als Diagonalelemente.

Satz 5.4: Die Matrix A erfülle folgende drei Voraussetzungen:
(a) Das LR-Verfahren ist durchführbar.
(b) Für die Eigenwerte λ_i von A gilt $|\lambda_1| > |\lambda_2| > ... > |\lambda_n|$
(c) Wenn A in der Form zerlegt wird $A = U^T \Lambda U$ mit der Diagonalmatrix $\Lambda = diag(\lambda_1,...,\lambda_n)$ und der Matrix U, so besitzen U und U^T jeweils eine LR-Zerlegung $U = L_x R_x, U^T = L_y R_y$.
Dann konvergieren die durch das Verfahren erzeugten Folgen von Matrizen A_k, R_k, L_k und es gilt

$$\lim_{k\to\infty} A_k = \lim_{k\to\infty} R_k = \begin{pmatrix} \lambda_1 & * & *\ldots & * \\ 0 & \lambda_2 & *\ldots & * \\ 0 & 0 & \ldots & * \\ 0 & 0 & 0\ldots & \lambda_n \end{pmatrix}, \quad \lim_{k\to\infty} L_k = E.$$

Beweis: Wilkinson (1965); vgl. Stoer/Bulirsch II, S.60; Finckenstein I, S.194

Praktische Durchführung des Verfahrens:

1. Die Konvergenzuntersuchung zeigt, dass die letzte Zeile von A_k gegen $(0,0,...,0,\lambda_n)$ konvergiert, wobei die Konvergenzordnung linear ist mit der linearen Konvergenzrate

$$C = \left|\frac{\lambda_n}{\lambda_{n-1}}\right|.$$

Wenn das Nichtdiagonalelement $a_{n-1,n}^{(k)}$ von A_k genügend klein geworden ist, kann somit $a_{n,n}^{(k)}$ als Näherung für λ_n abgespalten werden, und die Iteration wird mit einer (n-1,n-1)-Matrix fortgesetzt, die aus A_k entsteht, indem die letzte Zeile und Spalte gestrichen werden.

Abbruchtest (in S2): Wenn gilt

$$\left|a_{n,n-1}^{(k)}\right| < eps(\left|a_{n,n}^{(k)}\right| + \left|a_{n-1,n-1}^{(k)}\right| + \left|a_{1,1}^{(k)}\right|) \qquad (5.13)$$

dann $\lambda_n := a_{n,n}^{(k)}$ und streiche n-te Zeile und Spalte (eps ist die vorgegebene relative Genauigkeit).

2. Liegen die betragskleinsten EW λ_n, λ_{n-1} eng zusammen, so kann die Konvergenz sehr langsam sein. Eine Konvergenzbeschleunigung kann dann über eine Spektrumsverschiebung der Matrix erfolgen (Shift-Technik), die i.a. die Konvergenz enorm beschleunigen kann. Wir wenden die LR-Zerlegung nicht auf A_k selbst an, sondern auf die Matrix

$$\overline{A}_k = A_k - \mu E$$

wobei μ eine Näherung für einen der Eigenwerte von A_k ist. Für den Konvergenzfaktor C gilt dann

$$\overline{C} = \left|\frac{\lambda_n - \mu}{\lambda_{n-1} - \mu}\right|.$$

Der Konvergenzfaktor $\overline{C}$ liegt umso näher an Null, je genauer μ den Eigenwert λ_n annähert.

Wahl von μ : Wegen $a_{n,n}^{(k)} \to \lambda_n$ wird die Wahl von $\mu = a_{n,n}^{(k)}$ günstig sein. Shift-Technik sollte bei fortgeschrittener Konvergenz einsetzen, etwa wenn gilt

$$\left|1 - \frac{a_{n,n}^{(k-1)}}{a_{n,n}^{(k)}}\right| \le \eta < 1 \qquad (5.14)$$

($\eta = \frac{1}{3}$ führt zu praktisch guten Ergebnissen).

Der Schritt (S1) im Algorithmus ist dann entsprechend zu modifizieren. Ist Test (5.14) erfüllt, so wird $\overline{A}_k = A_k - \mu E$ in der Form zerlegt $\overline{A}_k = \overline{L}_k \overline{R}_k$. Im Schritt (S2) gilt dann

$$\overline{A}_{k+1} = \overline{R}_k \overline{L}_k, \quad A_{k+1} = \overline{A}_{k+1} + \mu E$$

d.h. in der Hauptdiagonale von A_k ist jeweils μ zu subtrahieren, und bei der Berechnung von A_{k+1} ist μ jeweils zu addieren.

5.5.2. QR-Verfahren (Francis 1961)

Das QR-Verfahren ist eine Weiterentwicklung des LR-Verfahrens, es beruht auf der QR-Faktorisierung (vgl. Abschnitt 3.8.) und ist somit immer durchführbar. Die Ausgangsmatrix A sei wieder in Tridiagonalform überführt.

Beschreibung des Verfahrens:

$(S0)$ $A_1 := A; \quad k = 1$

$(S1)$ *Zerlege* A_k *in ein* Produkt $(QR - Zerlegung)$

$$A_k = Q_k R_k, \quad Q_k \text{ Orthogonalmatrix}, \quad R_k = \begin{pmatrix} * & * & ... & * \\ 0 & * & ... & * \\ 0 & 0 & ... & * \\ 0 & 0 & ...0 & * \end{pmatrix}$$

$(S2)$ *Bilde das* Produkt $A_{k+1} = R_k Q_k;$

$k := k+1 \quad goto\ (S1)$

Der Schritt (S1) kann mit Hilfe von Householder-Spiegelungen $H_1^{(k)}, H_2^{(k)}, ..., H_{n-1}^{(k)}$ (Abschnitt 3.8.) durchgeführt werden:

$$H_{n-1}^{(k)} * ... * H_2^{(k)} * H_1^{(k)} A_k = R_k$$

Wegen der Symmetrie und Orthogonalität der Householder-Matrizen folgt

$$Q_k = H_1^{(k)} * H_2^{(k)} * ... * H_{n-1}^{(k)}.$$

Analyse des Verfahrens: Wegen $A_k = Q_k R_k$ gilt $R_k = Q_k^{-1} A_k$ und damit

$$A_{k+1} = R_k Q_k = Q_k^{-1} A_k Q_k \ ,$$

d.h. die Matrizen A_k, A_{k+1} sind wieder ähnlich und besitzen die gleichen EW. Es kann ein Konvergenzsatz analog zu Satz 5.3. bewiesen werden, wobei die Voraussetzung (a) überflüssig ist. (Stoer/Bulirsch II, S.60; Kielbasinski/Schwetlick S.424 ff.)

Praktische Durchführung des Verfahrens

Bezüglich des Abbruchtest und der Shift-Strategie gelten die gleichen Aussagen wie für das LR-Verfahren. Besonderheiten ergeben sich für die Durchführung der Faktorisierung. Die Verwendung von Householder-Spiegelungen erweist sich i.a. als zu aufwändig. Bei symmetrischer Ausgangsmatrix hat A_k stets Tridiagonalform

$$A_k = \begin{pmatrix} \alpha_1 & \gamma_1 & 0 & 0... & 0 & 0 \\ \beta_1 & \alpha_2 & \gamma_2 & 0... & 0 & 0 \\ 0 & \beta_2 & \alpha_3 & \gamma_3 & 0 & 0 \\ 0 & 0 & ... & ... & ... & 0 \\ & & & \beta_{n-2} & \alpha_{n-1} & \gamma_{n-1} \\ 0 & 0 & 0 & 0... & \beta_{n-1} & \alpha_n \end{pmatrix}$$

bzw. Hessenberg-Form bei unsymmetrischer Ausgangsmatrix. Es sind somit nur die Elemente $\beta_1,\beta_2,...,\beta_{n-1}$ zu annullieren, um A_k in eine obere Dreiecksform R_k zu bringen. Statt mit Householder-Spiegelungen kann dies effektiver durch n-1 Givens-Rotationen (vgl. Definition 5.2) $G_{12},G_{23},...,G_{n-1n}$ geschehen, die nacheinander $\beta_1,\beta_2,...,\beta_{n-1}$ annullieren:

β_1 annullieren: $\quad \rho^{(1)} := \sqrt{\alpha_1^2+\beta_1^2}, \quad c := \frac{\alpha_1}{\rho}, \quad s := \frac{\beta_1}{\rho}$

$$G_{12}A = \begin{pmatrix} c & s & & & 0 \\ -s & c & & & \\ & & 1 & & \\ & & & \ddots & \\ 0 & & & & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & \gamma_1 & 0 & & 0 \\ \beta_1 & \alpha_2 & \gamma_2 & & \\ 0 & \beta_2 & \alpha_3 & & \\ & & & \ddots & \\ 0 & & & & \alpha_n \end{pmatrix} = \begin{pmatrix} \rho & \gamma_1^{(1)} & \delta_1^{(1)} & & \\ 0 & \alpha_2^{(1)} & \gamma_2^{(1)} & & \\ 0 & \beta_2 & \alpha_3 & & \\ & & & \ddots & \\ & & & & \alpha_n \end{pmatrix}$$

β_2 annullieren: $\quad \rho^{(2)} := \sqrt{\alpha_2^{(1)^2}+\beta_2^2}, \quad c := \frac{\alpha_2^{(1)}}{\rho}, \quad s := \frac{\beta_2}{\rho}$

$$G_{23}G_{12}A = \begin{pmatrix} 1 & 0 & 0 & & 0 \\ 0 & c & s & & \\ & -s & c & & \\ & & & \ddots & \\ 0 & & & & 1 \end{pmatrix} \begin{pmatrix} \rho^{(1)} & \gamma_1^{(1)} & \delta_1^{(1)} & & 0 \\ 0 & \alpha_2^{(1)} & \gamma_2^{(1)} & 0 & \\ 0 & \beta_2 & \alpha_3 & \gamma_3 & \\ & & & \ddots & \\ 0 & & & & \alpha_n \end{pmatrix} = \begin{pmatrix} \rho^{(1)} & \gamma_1^{(1)} & \delta_1^{(1)} & & \\ 0 & \rho^{(2)} & \gamma_2^{(2)} & \delta_2^{(2)} & \\ 0 & 0 & \alpha_3^{(2)} & \gamma_3^{(2)} & \\ & & & \ddots & \\ & & & & \alpha_n \end{pmatrix}$$

Dabei gibt es sechs geänderte Elemente in jedem Schritt (im (n-1)-ten Schritt nur 4 geänderte Elemente); zusätzliche fill-in Elemente $\delta_i^{(i)}$ entstehen.

Es gilt:

$$G_{n-1,n} * ... * G_{23} * G_{12} A_k = R_k = Q_k^T A_k$$

bzw.

$$A_k = Q_k R_k \ mit \ \ Q_k = G_{12}^T * G_{23}^T * ... * G_{n-1,n}^T.$$

Bemerkungen:

1. Die Matrizen $G_{12},G_{23},...,G_{n-1n}$ usw. sind durch die 3 Größen ρ,c,s jeweils eindeutig bestimmt, brauchen somit nicht explizit gespeichert zu werden.
2. Es muss die Matrix Q_k nicht explizit bestimmt zu werden, sondern Schritte (S1) und (S2) können verbunden werden (gestaffelte Ausführung von Vorwärts- und Rückwärtsrechnung). Da jeweils nur 6 Elemente in der Matrix geändert werden, gilt

$$A_{k+1} = R_k Q_k = ...G_{34}[(G_{23}G_{12}A_k)G_{12}{}^T]G_{23}^T....$$

bzw. eine analoge Formel, wenn mit Shift-Technik gearbeitet wird. Wenn $R_k Q_k$ bestimmt wurde, müssen also die Hauptdiagonalelemente noch um μ vergrößert werden, um A_{k+1} zu erhalten.
3. Die Konvergenz mit Shift-Technik ist kubisch. Praktisch kann nach etwa 1.7*n QR-Schritten abgebrochen werden, wobei ca. 4-6 Schritte für die Abspaltung von λ_n,λ_{n-1} benötigt werden und dann 1-2 Schritte für die weiteren EW.

Bemerkungen zum unsymmetrischen EW-Problem

1. Der QR-Algorithmus ist auch auf unsymmetrische EW-Probleme effektiv anwendbar. Die Ausgangsmatrix wird dabei mittels Householder-Transformation zunächst auf obere Hessenberg-Form gebracht.

$$\overline{A} = \begin{pmatrix} * & * & * & *\ldots & * & * \\ * & * & * & *\ldots & * & * \\ 0 & * & * & * & * & * \\ 0 & 0 & * & * & * & * \\ \ldots & & & * & * & * \\ 0 & 0 & \ldots & 0 & * & * \end{pmatrix}$$

Alle Matrizen A_k besitzen dann wieder diese Form, sind also ähnlich.

2. Die Matrizen können im unsymmetrischen Fall auch komplexe EW haben. Bei Konvergenz verbirgt sich ein Paar konjugiert komplexer EW in einer reellen (2x2)-Teilmatrix $\begin{pmatrix} a & b \\ c & d \end{pmatrix}$ der Hauptdiagonale:

$$\lim_{k\to\infty} A_k = \lim_{k\to\infty} R_k = \begin{pmatrix} \lambda_1 & * & *\ldots & * \\ 0 & a & b\ldots & * \\ 0 & c & d\ldots & * \\ 0 & 0 & 0\ldots & \lambda_n \end{pmatrix}$$

Die Shift-Strategie ist in diesem Fall zu erweitern, es werden QR-Doppelschritte durchgeführt, um eine komplexe Rechnung zu vermeiden.

3. Die Verfahren können im symm. und auch unsymm. Fall erweitert werden, um alle EV zu bestimmen. Der Aufwand steigt allerdings stark an. Werden nur einige EV benötigt, ist nach Berechnung der EW die inverse Iteration günstiger zur Berechnung der EV, da sie jeweils nur wenige Schritte benötigt.

5.6. Kondition des Eigenwert-Problems und Fehlerabschätzungen

Von entscheidender Bedeutung für die Stabilität des numerischen Prozesses ist die Frage: Wie stark ändern sich die EW bei (kleinen) Änderungen der Matrix?

Gestörtes EW-Problem:
Statt der Matrix A habe man die gestörte Matrix $A+\varepsilon B$. Da A symmetrisch ist (es wird i.a. nur das obere Dreieck verwendet), nimmt man die Störmatrix B sinnvoller Weise auch als symmetrisch an. Es sei λ ein einfacher Eigenwert von A: $Ax = \lambda x$ mit $\|x\|_2 = 1$. Stattdessen liege nun das gestörte Problem vor:

$$(A+\varepsilon B)x(\varepsilon) = \lambda(\varepsilon)x(\varepsilon) \text{ mit } \|x(\varepsilon)\|_2 = 1$$

Es gilt: Da $A(\varepsilon) = A + \varepsilon B$ bezüglich ε differenzierbar ist, sind $\lambda(\varepsilon), x(\varepsilon)$ bezüglich ε differenzierbar (Satz über implizite Funktionen). Wir machen daher den Ansatz:

$$\lambda(\varepsilon) = \lambda + a_1\varepsilon + a_2\varepsilon^2 + \ldots \quad mit \quad a_1 = \left(\frac{d\lambda(\varepsilon)}{d\varepsilon}\right)_{\varepsilon=0}$$

$$x(\varepsilon) = x + b^1\varepsilon + b^2\varepsilon^2 + \ldots \quad mit \;\; b^1 = \left(\frac{dx(\varepsilon)}{d\varepsilon}\right)_{\varepsilon=0}.$$

Wir setzen ein und führen einen Koeffizientenvergleich durch, um den Entwicklungskoeffizienten a_1 zu bestimmen:

$$(A+\varepsilon B)(x+\varepsilon b^1+\ldots) = (\lambda + a_1\varepsilon + a_2\varepsilon^2 + \ldots)(x+\varepsilon b^1+\ldots)$$

$$\varepsilon^0: \quad Ax = \lambda x$$

$$\varepsilon^1: \quad Bx + Ab^1 = a_1 x + \lambda b^1$$

Multiplikation mit x^T liefert: $a_1 = \left(\frac{d\lambda(\varepsilon)}{d\varepsilon}\right)_{\varepsilon=0} = \frac{x^T Bx}{x^t x}$

Ergebnis: $\lambda_i(A+\varepsilon B) - \lambda_i(A) = \varepsilon \frac{x^T Bx}{x^T x} + O(\varepsilon^2).$

Da man die Störungsmatrix B i.a. nicht kennt, nimmt man eine pauschale Schranke an: $\|B\|_2 \le \|A\|_2$.

Wegen $\left| \frac{x^T Bx}{x^t x} \right| \le \|B\|_2 \le \|A\|_2$ folgt dann:

$$\boxed{\left| \lambda_i(A+\varepsilon B) - \lambda_i(A) \right| \le |\varepsilon| \, \|A\|_2 + O(\varepsilon^2).}$$

Damit gilt:

Satz 5.5: Die einfachen Eigenwerte einer symmetrischen Matrix sind gut konditioniert. Kleine (symmetrische) Störungen in A bewirken kleine Störungen in den Eigenwerten.

Bem.: Die Aussage gilt auch für mehrfache Eigenwerte symmetrischer Matrizen, aber i.a. nicht für die EV. Dass die Aussage für unsymmetrische Matrizen nicht mehr gilt, zeigt das folgende Beispiel:

Beispiel 1:

$$A = \begin{pmatrix} 1 & 1 \\ 0 & 1+10^{-10} \end{pmatrix}, \; B = \begin{pmatrix} 1 & 1 \\ 10^{-10} & 1+10^{-10} \end{pmatrix}$$

A besitzt die Eigenwerte $\lambda_1 = 1, \lambda_2 = 1+10^{-10}$. Die Matrix B besitzt die Eigenwerte $\mu_1 = 1-10^{-5}, \mu_2 = 1+10^{-5}$. Die Änderung eines Elementes in A um 10^{-10} führt damit zu einer Änderung in den Eigenwerten, die 10^5 -mal größer ist.

Die ungünstigen numerischen Eigenschaften machen das unsymmetrische Eigenwertproblem häufig kompliziert.

Der obige Satz 5.5 sichert für symmetrische Matrizen, dass bei kleinen Eingangs- und Rundungsfehlern das Ergebnis als exakter Eigenwert einer benachbarten Matrix interpretiert werden kann. In der Regel möchte man nachträglich überprüfen, mit welcher Genauigkeit das Ergebnis zu bewerten ist (a-posteriori Fehlerschätzung):

Satz 5.6: A sei eine diagonalähnliche Matrix (nicht notwendig symmetrisch) mit den EW $\lambda_1, \lambda_2, \ldots, \lambda_n$; d.h. es gibt dann gibt eine reguläre Matrix T mit

$$T^{-1}AT = \Lambda = diag(\lambda_1, \lambda_2, \ldots, \lambda_n). \qquad (5.15)$$

Es sei $\tilde{\lambda}$ die Näherung eines Eigenwertes und $\tilde{x} \neq 0$ die des zugehörigen Eigen-

vektors, und für ein $\varepsilon > 0$ gilt

$$\left\| A\tilde{x} - \tilde{\lambda}\tilde{x} \right\| \le \varepsilon \left\| A\tilde{x} \right\| . \tag{5.16}$$

Dann gilt

$$\min_{\lambda_j \neq 0} \left| \frac{\tilde{\lambda} - \lambda_j}{\lambda_j} \right| \le \varepsilon \left\| T \right\|_2 \left\| T^{-1} \right\|_2 = \varepsilon \, cond_2(T) \tag{5.17}$$

Bemerkungen:

1. Für symmetrische Matrizen kann T=U sogar als orthogonale Matrix gewählt werden. Für diese gilt $cond_2(U) = 1$ und (5.17) erhält die Form

$$\min_{\lambda_j \neq 0} \left| \frac{\tilde{\lambda} - \lambda_j}{\lambda_j} \right| \le \varepsilon$$

d.h. ein Eigenwert λ_j von A wird durch $\tilde{\lambda}$ mit relativem Fehler $\le \varepsilon$ angenähert.

2. Aus (5.16) folgt als Abbruchbedingung für die Vektoriteration:
Ist $\left\| Ax^k - \rho_k x^k \right\| \le \varepsilon \left\| Ax^k \right\|$,so werden $\lambda = \rho_k$ und $x = x^k$ als Näherungen eines EW bzw. EV akzeptiert.

Für Abschätzungen der Lage der Eigenwerte ist die folgende Aussage von Bedeutung:

Satz 5.7: (Gerschgorinscher Kreissatz)
Sei A eine beliebige (n,n)-Matrix (nicht notwendig symmetrisch) und gilt

$$s_i := \sum_{\substack{j=1 \\ j \neq i}}^{n} \left| a_{ij} \right| \quad i = 1, 2, ..., n \tag{5.18}$$

Dann gelten die Aussagen:

(a) Jeder Eigenwert λ_i von A liegt in der Vereinigung der Kreise

$$K_i = \left\{ z \in C \mid \left| z - a_{ii} \right| \le s_i \right\} \quad i = 1, 2, ..., n \tag{5.19}$$

(b) Jede Zusammenhangskomponente der Menge $\bigcup_{i=1}^{n} K_i$, die aus m Kreisen besteht, enthält genau m Eigenwerte.

Bem.: Da die Eigenwerte von A mit denen von A^T übereinstimmen, gilt auch:
(a') Jeder Eigenwert liegt in der Vereinigung der Kreise

$$K_j' = \left\{ z \in C \mid \left| z - a_{jj} \right| \le r_j \right\} \quad j = 1, 2, ..., n \qquad r_j = \sum_{\substack{i=1 \\ i \neq j}}^{n} \left| a_{ij} \right|$$

(b') Jede Zusammenhangskomponente von $\bigcup_{j=1}^{n} K_j'$, die aus q Kreisen besteht, enthält genau q Eigenwerte.

Beispiel:

$$A = \tfrac{1}{8} \begin{pmatrix} -8 & -2 & 4 \\ -1 & -4 & 2 \\ 2 & 2 & -10 \end{pmatrix} \quad \begin{matrix} s_1 = \tfrac{6}{8} & a_{11} = -1 \\ s_2 = \tfrac{3}{8} & a_{22} = -\tfrac{1}{2} \\ s_3 = \tfrac{4}{8} & a_{33} = -\tfrac{5}{4} \end{matrix}$$

$$r_1 = \tfrac{3}{8} \quad r_2 = \tfrac{1}{2} \quad r_3 = \tfrac{3}{4}$$

Bemerkungen zur naiven Berechnung der EW

Die Bestimmung der Eigenwerte einer Matrix durch Berechnung des charakteristischen Polynoms $P_n(\lambda)$ und Berechnung der Nullstellen des Polynoms ist nur bei kleiner Dimension n praktisch durchführbar. Für größere n werden die Polynomkoeffizienten durch Rundungsfehler verfälscht, der Berechnungsweg

Elemente von A $\rightarrow$ Koeffizienten des Polynoms $\rightarrow$ Eigenwerte

kann hochgradig instabil werden. Wir betrachten dazu das Polynom

$$P_n(\lambda) = \lambda^n + p_{n-1}\lambda^{n-1} + ... + p_1\lambda + p_0$$

Verfälschung von Koeffizienten durch Rundungsfehler:

$$p_j \rightarrow \tilde{p}_j = p_j + \varepsilon q_j \quad j = 0,1,...,n-1$$

$$\tilde{P}_n(\lambda) = P_n(\lambda) + \varepsilon Q(\lambda) \quad \text{(gestörtes Polynom)}$$

$$Q(\lambda) = q_{n-1}\lambda^{n-1} + ... + q_1\lambda + q_0 \quad \text{(Störungspolynom)}$$

Die Nullstellen $\tilde{\lambda}_k$ von $\tilde{P}_n(\lambda)$ hängen zwar stetig von ε ab, sind i.a. aber nicht stetig differenzierbar abhängig von ε, d.h. kleine ε können große Störungen in λ bewirken.

Voraussetzung.: $\tilde{\lambda}_k$ einfache Nullstelle von $\tilde{P}_n(\lambda)$: $P_n(\lambda) = 0,\ P_n'(\lambda) \neq 0.$.

Dann gilt für kleines ε

$$\tilde{\lambda}_k = \lambda_k + \Delta\lambda \qquad \text{mit } \Delta\lambda = -\frac{\tilde{P}(\lambda_k)}{\tilde{P}'(\lambda_k)} = -\frac{\varepsilon Q(\lambda_k)}{P'(\lambda_k) + \varepsilon Q'(\lambda_k)} \quad \text{(Newtonkorrektur).}$$

Falls gilt $|\varepsilon Q'(\lambda_k)| << |P'(\lambda_k)|$, so erhält man

$$\boxed{\tilde{\lambda}_k = \lambda_k - \varepsilon\frac{Q(\lambda_k)}{P'(\lambda_k)}}$$

und $\dfrac{Q(\lambda_k)}{P'(\lambda_k)}$ ist die absolute Konditionszahl (Faktor der Fehlerverstärkung) bezüglich der Nullstelle λ_k.

Beispiel: n=12

Eigenwerte: $\lambda_1 = 1, \lambda_2 = 2, ..., \lambda_{12} = 12 \Rightarrow$

$$P_{12}(\lambda) = \prod_{k=1}^{12}(\lambda - k) = \lambda^{12} - 78\lambda^{11} \pm .... - 6926634\lambda^7 + ... + 479001600$$

Wir stören einen Koeffizienten in $P_{12}(\lambda)$: $Q(\lambda) = q_j\lambda^j \quad j \in \{0,...,11\}$

Es ergibt sich dann die Konditionszahl

$$K_{k,j} = \left| \frac{Q(\lambda_k)}{P'(\lambda_k)} \right| \qquad (5.20)$$

Es sei der Koeffizient p_7 gestört: $p_7 \rightarrow \tilde{p}_7 = -6926634.001$, d.h. $Q(\lambda) = p_7\lambda^7$ wird mit ε multipliziert $\varepsilon = 1.444 \cdot 10^{-10}$. Wir wollen die Fehlerverstärkung $K_{9,7}$ berechnen, d.h. die Auswirkung der Störung von p_7 auf den Eigenwert $\lambda_9 = 9$

$$K_{9,7} = \left| \frac{Q(\lambda_9)}{P'(\lambda_9)} \right| = \frac{6926634 \cdot 9^7}{8!\,3!} = 1.37 \cdot 10^8$$

Der Fehler in p_7 wird auf mit dem Verstärkungsfaktor 10^8 übertragen; wegen $\varepsilon \approx 10^{-10}$ treten bei λ_9 Fehler schon in der 2.Dezimalstelle auf.