Verifizierende Numerik mit PASCAL-XSC BCD-Version

INVERSE MATRIZEN

Frithjof Blomquist

3. September 2000

Verifizierende Numerik mit PASCAL-XSC BCD-Version

INVERSE MATRIZEN

Inversions-Algorithmen der Numerik-Bibliothek I
für reelle Punkt- und Intervall-Matrizen

E-Mail Adressen: Blomquist@math.uni-wuppertal.de
ae29@rz.uni-karlsruhe.de

Ein Postsript- und dvi-File mit dem Inhalt dieses Artikels kann über die Web-Seite

`www.math.uni-wuppertal.de/org/WRST/blom.html`

bezogen werden.

Inhaltsverzeichnis

1 Einleitung

PASCAL-XSC (PASCAL eXtension for Scientific Computation) ist eine leistungsfähige, durch Erweiterung von Standard PASCAL geschaffene Programmiersprache für numerische Anwendungen. Sie liefert alle notwendigen Werkzeuge zur Realisierung von Algorithmen, die zu einem vorgegebenen Problem numerische Ergebnisse liefern, aus denen mathematische Aussagen gewonnen werden können, d.h dem Anwender wird ermöglicht, mit Gleitkomma-Ergebnissen eines Computers mathematische Beweise zu führen.

Wertet man mit **PASCAL-XSC** einen selbstverifizierenden Algorithmus beispielsweise zur Berechnung der Inversen $\boldsymbol{A}^{-1}$ einer gegebenen, regulären Matrix $\boldsymbol{A}$ aus, so erhält man hochgenaue Intervalleinschließungen, in denen die jeweils gesuchten, inversen Elemente **bewiesenermaßen** enthalten sind. Bekanntlich liefern herkömmliche Inversionsalgorithmen wegen unvermeidbarer Rundungsfehler in den Gleitkommarechnungen um so stärkere Abweichungen vom exakten Ergebnis $\boldsymbol{A}^{-1}$, je schlechter die zu invertierende Matrix $\boldsymbol{A}$ konditioniert ist. In solchen Fällen ist es dann sehr hilfreich, diese fehlerbehafteten Ergebnisse mit den Einschließungen der exakten Ergebnisse eines selbstverifizierenden Algorithmus von **PASCAL-XSC** vergleichen zu können.

An dieser Stelle soll noch ausdrücklich darauf hingewiesen werden, daß neben den oben beschriebenen Rundungsfehlern bei Gleitkommarechnungen auch noch die sogenannten Konversionsfehler zu beachten sind, wenn dezimale Eingabedaten (z.B. Matrixelemente) in das oft benutzte binäre **IEEE**-Datenformat umgewandelt werden müssen. Bekanntlich lassen sich schon so einfache Zahlen wie 1.1 oder 0.1 nicht exakt in diesem Datenformat darstellen. Die Erfahrung zeigt nun, daß sich Konversionsfehler auf die Berechnung von $\boldsymbol{A}^{-1}$ merkbar negativ auswirken, wenn $\boldsymbol{A}$ schlecht konditioniert ist. Lassen sich diese Konversionsfehler nicht dadurch vermeiden, daß man die Elemente von $\boldsymbol{A}$ durch Multiplikation mit einer geeigneten Zehnerpotenz ganzzahlig macht, was beim nachfolgenden Inversionsalgorithmus oft zum Overflow führt, so müssen die nicht exakt konvertierbaren Matrixelemente durch **IEEE**-Intervalle mit von Null verschiedenen Durchmessern eingeschlossen werden. Bei schlecht konditionierten Matrizen lassen sich dann die inversen Elemente oft nur noch sehr grob oder auch gar nicht mehr einschließen. Diese nur durch Konversionsfehler verursachte Fehlerquelle läßt sich ganz einfach ausschalten, wenn man die BCD-(**B**inary **C**oded **D**ecimal)-Version von **PASCAL-XSC** benutzt, bei der alle dezimalen Eingabedaten des benutzten Datenformats (real oder Longreal) **rundungsfehlerfrei** gespeichert werden.

In den nachfolgenden Abschnitten werden selbstverifizierende Algorithmen zur Inversion dichtbesetzter, quadratischer Matrizen behandelt, wobei sich die garantierte Einschließung von $\boldsymbol{A}^{-1}$ durch Anwendung des Fixpunktsatzes von Brouwer ergibt [6, 3]. Bei den Inversionsalgorithmen werden zunächst nur Matrizen mit reellen Punkt- und Intervall–Elementen betrachtet; die Ergebnisse lassen sich jedoch direkt auch auf komplexe Matrizen übertragen. In der Numerik-Bibliothek I der BCD–Version, die über die auf Seite 4 angegebene Internet–Adresse frei verfügbar ist, findet man alle Routinen zur Matrixinversion. Diese Routinen werden in [2] ausführlich beschrieben.

Zur Einschließung von $\boldsymbol{A}^{-1}$ und zur Einschließung des Lösungsvektors $\boldsymbol{x}$ eines linearen Gleichungssystems $\boldsymbol{Ax} = \boldsymbol{b}$ benötigt man gute **Näherungen** der Inversen $\boldsymbol{A}^{-1}$, so daß auch Algorithmen vorgestellt werden, mit denen man nur punktförmige Approximationen von $\boldsymbol{A}^{-1}$ berechnen kann.

Viele numerische Beispiele zeigen den korrekten Einsatz der Routinen zur Matrix–Inversion und demonstrieren die mathematische Aussagekraft der Einschließungsmethoden.

Im Anhang findet man eine Zusammenstellung aller Routinen zur Matrix–Inversion reeller Matrizen zum Basistyp `Longreal` mit einer Kurzbeschreibung ihrer Eingangs- und Ausgangsgrößen, wodurch eine sofortige Anwendung der Algorithmen auch ohne Studium der vorderen Abschnitte möglich ist. Vorausgesetzt werden dabei lediglich Grundkenntnisse in Standard-Pascal und **PASCAL-XSC**. In der Sprachbeschreibung [7], im Toolbox-Band 1 [3] und in der Beschreibung der Numerik–Bibliothek I [2] findet der ungeübte Anwender viele einfache Programm-Beispiele zu **PASCAL-XSC**.
Dokumentationen zur Sprache **PASCAL-XSC** stehen im Netz unter

`www.uni-karlsruhe.de/~iam/html/literatur/download.html`

zur Verfügung.

2 Zur Notation

$I\!R$	Menge der reellen Zahlen
$II\!R$	Menge der reellen Intervalle
R	Menge der reellen Gleitkommazahlen
IR	Menge der reellen Gleitkomma–Intervalle
$\mathbb{C}$	Menge der komplexen Zahlen
$I\mathbb{C}$	Menge der komplexen Intervalle
C	Menge der komplexen Gleitkommazahlen
IC	Menge der komplexen Gleitkomma–Intervalle
$\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}, \boldsymbol{D}, \boldsymbol{I}, \boldsymbol{L}$	Punktmatrizen
$\boldsymbol{M}, \boldsymbol{U}, \boldsymbol{V}, \boldsymbol{R}, \boldsymbol{S}, \boldsymbol{Y}$	Punktmatrizen
$\boldsymbol{I}$	Einheitsmatrix
$[\boldsymbol{Y}]$	Intervallmatrix $[\boldsymbol{Y}] \in II\!R^{n\times n}$ oder $[\boldsymbol{Y}] \in IR^{n\times n}$
$[\boldsymbol{Y}]^{(k)}$	Bei einer Iteration die k-te Intervallmatrix
$[\boldsymbol{Y}]_\diamond$	mit Intervallarithmetik im Gleitkommaformat berechnete Intervallmatrix $[\boldsymbol{Y}]_\diamond \in IR^{n\times n}$
$\boldsymbol{A}^T, [\boldsymbol{Y}]^T$	Transponierte Matrizen
$\boldsymbol{A}^{-1}$	Inverse Matrix: $\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I}$
$[\boldsymbol{Y}]^{-1}$	enthält zu jedem $\boldsymbol{A} \in [\boldsymbol{Y}]$ auch die Inverse $\boldsymbol{A}^{-1}$
$\boldsymbol{P}, \boldsymbol{X}$	Punkt– oder Intervallmatrizen
$\boldsymbol{X} = (x_{ij})$	$x_{i,j}:$ Elemente der Matrix $\boldsymbol{X}$
$\boldsymbol{x}_j$	j-te Spalte der Matrix $\boldsymbol{X}$
$\boldsymbol{M}$	Mittelpunktsmatrix
$\boldsymbol{D}$	Boothroyd/Dekker–Matrix
$\boldsymbol{R}$	Näherungsinverse: $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$
$\boldsymbol{R}_1, \boldsymbol{R}_2$	doppeltlange Näherung: $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$
$\boldsymbol{R}^{(\nu)}$	ν-te Näherung: $\boldsymbol{R}^{(\nu)} \approx \boldsymbol{A}^{-1}$
$\boldsymbol{R}_1^{(\nu)}, \boldsymbol{R}_2^{(\nu)}$	doppeltlange ν-te Näherung: $\boldsymbol{R}_1^{(\nu)} + \boldsymbol{R}_2^{(\nu)} \approx \boldsymbol{A}^{-1}$
${}^l\boldsymbol{D}$	Defekt, Version links: ${}^l\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}$
${}^r\boldsymbol{D}$	Defekt, Version rechts: ${}^r\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}$

$\Vert \boldsymbol{A} \Vert$	Matrix–Gesamtnorm: $\Vert \boldsymbol{A} \Vert := n \cdot \max \vert a_{ij} \vert$
$\boldsymbol{b}, \boldsymbol{p}, \boldsymbol{x}$	Vektoren bzw. Spaltenvektoren
$\boldsymbol{x}^T$	Transponierter Vektor bzw. Zeilenvektor
$\boldsymbol{e}_j$	j-ter Einheitsvektor
$[\boldsymbol{x}]$	Intervallvektor $[\boldsymbol{x}] \in I\!I\!R^n$ oder $[\boldsymbol{x}] \in IR^n$
$[\boldsymbol{x}]^{(k)}$	Bei einer Iteration der k-te Intervallvektor
$[\boldsymbol{x}]_\diamond$	mit Intervallarithmetik im Gleitkommaformat berechneter Intervallvektor $[\boldsymbol{x}]_\diamond \in IR^n$
$\circ \in \{+, -, \bullet, /\}$	exakte Punkt- oder Intervall–Operation
$\circ \in \{\oplus, \ominus, \odot, \oslash\}$	Maschinenoperation mit Rundung zur nächsten Gleitkommazahl
$\circ \in \{\diamond\!\!\!\!\!+\,, \diamond\!\!\!\!\!-\,, \diamond\!\!\!\!\!\cdot\,, \diamond\!\!\!\!\!/\,\}$	Intervalloperator
$\square(\ldots)$	der exakte Ausdruck $(\ldots)$ wird (komponenten–weise) zur nächsten Gleitpunktzahl gerundet
$\diamond(\ldots)$	der exakte Ausdruck $(\ldots)$ wird (komponenten–weise) maximalgenau eingeschlossen
$[S_1, S_2]$	reelles Intervall: $[S_1, S_2] = \{x \in I\!R \mid S_1 \leq x \leq S_2\}$
$\varrho(\boldsymbol{A})$	Spektralradius von $\boldsymbol{A}$: Maximum der Beträge der Eigenwerte von $\boldsymbol{A}$

Anmerkungen:

1. Punkt– und Intervallmatrizen lassen sich im ganzen Text an Hand ihrer Bezeichner nicht immer unterscheiden; so bedeutet z.B. $\boldsymbol{X}$ auf Seite 19 eine Punktmatrix, während $\boldsymbol{X}$ auf Seite 90 eine Intervallmatrix ist.

2. Auch die Bedeutung des Bezeichners einer Punktmatrix ist nicht immer einheitlich. Auf Seite 19 ist $\boldsymbol{U}$ eine obere Dreiecksmatrix, und auf Seite 40 wird mit $\boldsymbol{U}$ eine beliebige quadratische Matrix angesprochen.

3 Definitionen und Sätze

Wir betrachten zunächst nur quadratische, reelle Matrizen $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. Mit der Einheitsmatrix $\boldsymbol{I} \in \mathbb{R}^{n \times n}$ ist zur Matrix $\boldsymbol{A}$ die Inverse $\boldsymbol{A}^{-1} \in \mathbb{R}^{n \times n}$ definiert durch die Gleichung:

$$\boxed{\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I}} \tag{1}$$

Ist $\mathrm{Det}(\boldsymbol{A})$ die Determinante von $\boldsymbol{A}$, so gelten folgende Sätze, wobei eine normierte obere bzw. untere Dreiecksmatrix in der Hauptdiagonalen nur Einsen und unterhalb bzw. oberhalb der Hauptdiagonalen nur Nullen besitzt:

1. $\boldsymbol{A}^{-1}$ existiert $\quad\Longleftrightarrow\quad$ $\boldsymbol{A}$ ist regulär $\quad\Longleftrightarrow\quad$ $\mathrm{Det}(\boldsymbol{A}) \neq 0$
2. $\boldsymbol{A}^{-1}$ ist eindeutig
3. $\boldsymbol{A}^{-1}$ ist regulär, d.h. $\mathrm{Det}(\boldsymbol{A}^{-1}) \neq 0$
4. $\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I}$
5. $(\boldsymbol{A}^{-1})^{-1} = \boldsymbol{A}$
6. $(\boldsymbol{A}\boldsymbol{B})^{-1} = \boldsymbol{B}^{-1}\boldsymbol{A}^{-1}$
7. $\boldsymbol{A}$ ist reguläre, normierte untere Dreiecksmatrix
   $\Longleftrightarrow$ $\boldsymbol{A}^{-1}$ ist reguläre, normierte untere Dreiecksmatrix.
8. $\boldsymbol{A}$ ist reguläre, normierte obere Dreiecksmatrix
   $\Longleftrightarrow$ $\boldsymbol{A}^{-1}$ ist reguläre, normierte obere Dreiecksmatrix.
9. $\boldsymbol{A}$ ist regulär u. symmetrisch $\quad\Longleftrightarrow\quad$ $\boldsymbol{A}^{-1}$ ist regulär u. symmetrisch.

Die obigen Sätze 7 und 8 ermöglichen bei der Inversion von regulären Dreiecksmatrizen deutliche Laufzeitverkürzungen im Vergleich zu den Inversions–Algorithmen beliebiger regulärer Matrizen.

Bei der iterativen Berechnung der eindeutig bestimmten Inversen $\boldsymbol{A}^{-1}$ einer regulären Matrix $\boldsymbol{A}$ spielt die Defektmatrix eine zentrale Rolle. Zunächst gilt nach Satz 4. die Beziehung

$$\boldsymbol{I} - \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{I} - \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{0}$$

Setzt man hier für $\boldsymbol{A}^{-1}$ eine Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ ein, so erhält man an Stelle der Nullmatrix $\mathbf{0}$ die Defektmatrix, welche jedoch wegen $\boldsymbol{AR} \neq \boldsymbol{RA}$ auf zwei Arten zu definieren ist, je nachdem ob $\boldsymbol{A}$ von rechts oder links mit $\boldsymbol{R}$ multipliziert wird:

$$\boxed{{}^{r}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{AR}; \qquad {}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{RA}} \tag{2}$$

Wegen der Eindeutigkeit von $\boldsymbol{A}^{-1}$ gilt der

Satz 10. $\quad {}^{r}\boldsymbol{D} = {}^{l}\boldsymbol{D} = \mathbf{0} \iff \boldsymbol{R} = \boldsymbol{A}^{-1}$

Bedeutet n die Dimension einer beliebigen Matrix $\boldsymbol{D} = (d_{ij})$, und ist

$$\|\boldsymbol{D}\| := n \cdot \max_{i,j} |\, d_{ij} \,| \tag{3}$$

die mit den drei Vektornormen [13, S. 36] verträgliche Gesamtnorm[1] von $\boldsymbol{D}$, so gelten für eine gegen $\boldsymbol{A}^{-1}$ konvergierende Folge $\boldsymbol{R}^{(\nu)} \to \boldsymbol{A}^{-1}$ die Aussagen:

$$\lim_{\nu\to\infty} \boldsymbol{R}^{(\nu)} = \boldsymbol{A}^{-1} \quad \Leftrightarrow \quad \lim_{\nu\to\infty} {}^{r}\boldsymbol{D}^{(\nu)} = \mathbf{0} \quad \Leftrightarrow \quad \lim_{\nu\to\infty} \|\,{}^{r}\boldsymbol{D}^{(\nu)}\,\| = 0 \tag{4}$$

$$\lim_{\nu\to\infty} \boldsymbol{R}^{(\nu)} = \boldsymbol{A}^{-1} \quad \Leftrightarrow \quad \lim_{\nu\to\infty} {}^{l}\boldsymbol{D}^{(\nu)} = \mathbf{0} \quad \Leftrightarrow \quad \lim_{\nu\to\infty} \|\,{}^{l}\boldsymbol{D}^{(\nu)}\,\| = 0 \tag{5}$$

In der Praxis wird man daher iterative Algorithmen zur Approximation der Inversen so konstruieren, daß die Gesamtnormen $\|\,{}^{r}\boldsymbol{D}^{(\nu)}\,\|$ bzw. $\|\,{}^{l}\boldsymbol{D}^{(\nu)}\,\|$ möglichst schnell klein werden. Dabei zeigt sich insbesondere bei schlechtkonditionierten Matrizen $\boldsymbol{A}$, daß es keineswegs gleichgültig ist, welche der beiden möglichen Defektdefinitionen in (2) benutzt wird. Wir betrachten dazu das folgende Beispiel:

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix}$$

besitzt die maximalgenaue Näherungsinverse

$$\boldsymbol{R} := \begin{pmatrix} 1-10^{20} & 0 & 10^{20} & -1+10^{-20} \\ 1-10^{20} & 10^{20} & 0 & -1+10^{-20} \\ 1-10^{-20} & 0 & 0 & -1+10^{-20} \\ 2\cdot 10^{20}-2 & -10^{20} & -10^{20} & 3-2\cdot 10^{-20} \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

[1] Die Gesamtnorm ist von allen Matrixnormen am schnellsten zu berechnen.

Dabei heißt $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ *maximalgenau*, wenn $\boldsymbol{R}$ aus $\boldsymbol{A}^{-1}$ durch Rundung aller Komponenten $\boldsymbol{A}^{-1}[i,j]$ zur jeweils nächsten Rasterzahl entsteht. Berechnet man jetzt für die maximalgenaue Näherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ nach (2) die Normen der beiden möglichen Defektmatrizen ${}^{r}\boldsymbol{D}, {}^{l}\boldsymbol{D}$, so erhält man:

$$\|{}^{r}\boldsymbol{D}\| := \|\boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}\| = 4 \cdot 10^{-40}, \qquad \|{}^{l}\boldsymbol{D}\| := \|\boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}\| = 8 \cdot 10^{-20}$$

Je nach Wahl der Defektdefinition unterscheiden sich bei diesem Beispiel die entsprechenden Normen sogar um 20 Größenordnungen. Dabei wird die große Differenz begründet durch die bekannte Nichtkommutativität der Matrix–Multiplikation: $\boldsymbol{AR} \neq \boldsymbol{RA}$. Dies führt zu unserem ersten Merksatz:

> *Je nach Wahl der in (2) gegebenen Defektdefinitionen können sich die entsprechenden Normen ganz wesentlich unterscheiden.*

Bei den später zu behandelnden Defektiterationen werden wir sehen, daß eine erfolgreiche Konvergenz bei schlechtkonditionierten Matrizen ganz wesentlich von der richtigen Wahl der beiden möglichen Defektdefinitionen (2) abhängt. Als Faustregel kann man sagen, daß diejenige Defektdefinition die *richtige* ist, mit der man für eine nicht zu schlechte Ausgangsnäherung $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ die kleinere Norm erhält. Beachten Sie bitte in diesem Zusammenhang, daß die Defektiteration für die beiden auf Seite 49 angesprochenen Matrizen $\boldsymbol{A}$, $\boldsymbol{B}$ nur mit jeweils einer der beiden möglichen Defektdefinitionen Konvergenz liefert!

Da die in (2) definierten Defekte bei der Berechnung einer Näherungsinversen eine zentrale Rolle spielen, sollen die folgenden Beispiele auf mögliche Mißverständnisse hinweisen. Ausgehend von den Gleichungen (4),(5) könnte man zunächst der Auffassung sein, daß eine Iteration erfolgreich abgebrochen werden kann, wenn eine Defektnorm klein genug geworden ist und z.B. den Wert 10^{-50} unterschreitet. Die folgende erste Beispielmatrix $\boldsymbol{B}$ zeigt jedoch, daß auch bei beliebig kleinen Defektnormen die Näherung $\boldsymbol{R} \approx \boldsymbol{B}^{-1}$ völlig unbrauchbar ist, wenn die Matrix $\boldsymbol{B}$ nur hinreichend schlecht konditioniert ist. Die Matrix $\boldsymbol{C}$ demonstriert im zweiten Beispiel den praktisch gegenteiligen aber genauso überraschenden Effekt, daß nämlich auch bei beliebig großen Defektnormen die Näherung $\boldsymbol{R} \approx \boldsymbol{C}^{-1}$ beliebig gut sein kann, wenn die Matrix $\boldsymbol{C}$ nur hinreichend schlecht konditioniert ist. Die beiden folgenden Beispiele zeigen damit, daß die Größe einer Defektnorm als Abbruchkriterium für eine Defektiteration nicht benutzt werden darf!

1. Beispiel:

$$\boldsymbol{B} := \begin{pmatrix} 1 & 1 & 0 \\ 0 & \varepsilon & 0 \\ 0 & 0 & \varepsilon^2 \end{pmatrix}; \qquad \boldsymbol{B}^{-1} = \begin{pmatrix} 1 & -\varepsilon^{-1} & 0 \\ 0 & +\varepsilon^{-1} & 0 \\ 0 & 0 & \varepsilon^{-2} \end{pmatrix}; \qquad 0 < \varepsilon < 1$$

Die Näherungsinverse $\boldsymbol{R} := \begin{pmatrix} 1 & -\varepsilon^{-1} & \varepsilon^{-1} \\ 0 & +\varepsilon^{-1} & 0 \\ 0 & 0 & \varepsilon^{-2} \end{pmatrix}$ liefert nach (2) die Defekte:

$${}^{l}\boldsymbol{D} := \begin{pmatrix} 0 & 0 & -\varepsilon \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \|{}^{l}\boldsymbol{D}\| = 3\varepsilon; \qquad {}^{r}\boldsymbol{D} := \begin{pmatrix} 0 & 0 & -\varepsilon^{-1} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \|{}^{r}\boldsymbol{D}\| = \frac{3}{\varepsilon};$$

Für $0 < \varepsilon < 1$ erhält man zusammen mit der durch $\varkappa(\boldsymbol{B}) := \|\boldsymbol{B}\| \cdot \|\boldsymbol{B}^{-1}\| = 9/\varepsilon^2$ definierten Konditionszahl $\varkappa(\boldsymbol{B})$ die folgenden Ergebnisse:

Defektnorm und Konditionszahl $\varkappa(\boldsymbol{B})$				
ε	$\boldsymbol{R}[1,3] \neq 0$	$\|{}^{l}\boldsymbol{D}\|$	$\|{}^{r}\boldsymbol{D}\|$	$\varkappa(\boldsymbol{B})$
10^{-50}	10^{+50}	$3 \cdot 10^{-50}$	$3 \cdot 10^{+50}$	$9 \cdot 10^{+100}$
10^{-100}	10^{+100}	$3 \cdot 10^{-100}$	$3 \cdot 10^{+100}$	$9 \cdot 10^{+200}$

Man erkennt, daß sich die Normen $\|{}^{l}\boldsymbol{D}\|$, $\|{}^{r}\boldsymbol{D}\|$ mit wachsender Konditionszahl immer stärker unterscheiden und daß dabei $\|{}^{l}\boldsymbol{D}\|$ beliebig klein wird, obwohl sich die $\boldsymbol{R}[1,3] = 1/\varepsilon$ von $\boldsymbol{B}^{-1}[1,3] = 0$ immer stärker unterscheiden. Wir formulieren daher den nächsten Merksatz:

> *Auch bei beliebig kleiner Defektnorm kann die Näherungsinverse unbrauchbar sein, wenn die zu invertierende Matrix nur hinreichend schlecht konditioniert ist.*

2. Beispiel:

$$\boldsymbol{C} := \begin{pmatrix} 1 & 1 & 0 \\ 0 & \varepsilon & 1 \\ 0 & 0 & \varepsilon \end{pmatrix}; \qquad \boldsymbol{C}^{-1} = \begin{pmatrix} 1 & -\varepsilon^{-1} & +\varepsilon^{-2} \\ 0 & +\varepsilon^{-1} & -\varepsilon^{-2} \\ 0 & 0 & \varepsilon^{-1} \end{pmatrix}; \qquad 0 < \varepsilon < 1$$

Die Näherungsinverse $\boldsymbol{R} := \begin{pmatrix} 1 & -\varepsilon^{-1} & +\varepsilon^{-2} + \varepsilon^{-1} \\ 0 & +\varepsilon^{-1} & -\varepsilon^{-2} \\ 0 & 0 & \varepsilon^{-1} \end{pmatrix}$ liefert nach (2) die Defekte:

$${}^{l}\boldsymbol{D} := \begin{pmatrix} 0 & 0 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \|{}^{l}\boldsymbol{D}\| = 3; \qquad {}^{r}\boldsymbol{D} := \begin{pmatrix} 0 & 0 & -\varepsilon^{-1} \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \|{}^{r}\boldsymbol{D}\| = \frac{3}{\varepsilon};$$

Für $0 < \varepsilon < 1$ erhält man zusammen mit der durch $\varkappa(\boldsymbol{C}) := \|\boldsymbol{C}\| \cdot \|\boldsymbol{C}^{-1}\| = 9\varepsilon^2$ definierten Konditionszahl $\varkappa(\boldsymbol{C})$ die folgenden Ergebnisse:

Defektnorm und Konditionszahl $\varkappa(\boldsymbol{C})$				
ε	$\boldsymbol{R}[1,3] = \varepsilon^{-2} + \varepsilon^{-1}$	$\\|{}^l\boldsymbol{D}\\|$	$\\|{}^r\boldsymbol{D}\\|$	$\varkappa(\boldsymbol{C})$
10^{-50}	$10^{+100} + 10^{+50}$	3	$3 \cdot 10^{+50}$	$9 \cdot 10^{+100}$
10^{-100}	$10^{+200} + 10^{+100}$	3	$3 \cdot 10^{+100}$	$9 \cdot 10^{+200}$

Man erkennt wieder, daß sich die Normen $\|{}^l\boldsymbol{D}\|, \|{}^r\boldsymbol{D}\|$ mit wachsender Konditionszahl immer stärker unterscheiden und daß dabei $\|{}^r\boldsymbol{D}\|$ beliebig groß wird, obwohl der relative Fehler $(\boldsymbol{R}[1,3] - \boldsymbol{B}^{-1}[1,3])/\boldsymbol{B}^{-1}[1,3] = \varepsilon$ beliebig klein wird. Wir formulieren daher den nächsten Merksatz:

> *Auch bei beliebig großer Defektnorm kann die Näherungsinverse beliebig gut sein, wenn die zu invertierende Matrix nur hinreichend schlecht konditioniert ist.*

Abschließend noch zwei Anmerkungen zur Defektberechnung:

1. Es ist hinreichend bekannt, daß z.B. die Korrektur der Lösung eines linearen Gleichungssystems mit Hilfe des Defekts nur dann erfolgreich ist, wenn dieser möglichst genau berechnet werden kann. Daher hat man früher die Defekte teils unter großem Rechenaufwand in doppelter Länge bestimmt, was nicht in allen Fällen zum Ziel führte. In der Sprachumgebung von **PASCAL-XSC** stehen mit den Lattenkreuzoperationen sehr effektive Hilfsmittel zur Verfügung, mit denen man jetzt die Defekte stets *maximalgenau* berechnen kann.

2. Sind `I,A,R,Dl` **PASCAL-XSC** Variable vom Typ rmatrix, so liefert die Anweisung

```
Dl := #*( I - R*A );   { I: Einheitsmatrix }
```

   mit `Dl` die maximalgenaue Näherung des exakten Defekts ${}^l\boldsymbol{D} \approx$ `Dl`. Ist nun `Dl=0`, so darf man daraus nicht schließen, daß der exakte Defekt ${}^l\boldsymbol{D} = I - \boldsymbol{RA}$ verschwindet, da z.B. positive Akkuinhalte zu 0 gerundet werden, wenn sie kleiner sind als die kleinste positive Rasterzahl des benutzten real–Formats.

4 Einschließung der Inversen

4.1 Überblick

Wählt man in (1) von Seite 11 für die Inverse $\boldsymbol{A}^{-1}$ den Namen $\boldsymbol{X}$, so lautet die Definitionsgleichung der inversen Matrix:

$$\boxed{\boldsymbol{A}\,\boldsymbol{X} = \boldsymbol{I}} \tag{6}$$

Bedeutet dann $\boldsymbol{x}_j$ den j-ten Spaltenvektor von $\boldsymbol{X}$ und $\boldsymbol{e}_j$ den j-ten Einheitsvektor, so läßt sich (6) schreiben als:

$$\boldsymbol{A}\,\boldsymbol{x}_j = \boldsymbol{e}_j, \qquad j = 1, 2, \ldots, n \tag{7}$$

Damit hat man die Berechnung der Inversen $\boldsymbol{A}^{-1} \in I\!R^{n \times n}$ zurückgeführt auf die Lösung von n linearen Gleichungssystemen mit jeweils gleicher Koeffizientenmatrix $\boldsymbol{A}$. Nach der Philosphie der verifizierenden Numerik werden nur solche Algorithmen betrachtet, die **Einschließungen** der Lösungsvektoren $\boldsymbol{x}_j$ berechnen und damit auch **Einschließungen** der gesuchten Inversen $\boldsymbol{A}^{-1}$ liefern. Weitere Einzelheiten dazu findet man in den Toolbox–Bänden I,II [3, 4]. Auf den ersten Blick scheint damit das Problem der Matrixinversion gelöst zu sein. Dabei übersieht man jedoch, daß insbesondere bei schlecht konditionierten Matrizen die Einschließungsalgorithmen nur dann erfolgreich arbeiten, wenn eine punktförmige Näherungsinverse in ausreichender Genauigkeit zur Verfügung steht. Außerdem existieren Algorithmen zur Einschließung von $\boldsymbol{A}^{-1}$, welche die Lösung der n linearen Gleichungssysteme (7) vermeiden und bei nicht zu schlecht konditionierten Matrizen laufzeitmäßig sogar günstiger sind. Aber auch diese Einschließungs–Algorithmen benötigen wieder eine möglichst gute Punktnäherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$, wobei jedoch eine sonst übliche Fehlerabschätzung hier nicht erforderlich ist.

Zur Berechnung einer ersten Näherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ existieren Varianten des Algorithmus von Gauß, von denen wir die LU-Zerlegung nach Crout [3] vorziehen. Bei schlechtkonditionierten Matrizen $\boldsymbol{A}$ kann der Fall eintreten, daß der Einschließungsalgorithmus versagt, weil die erste Approximation $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ zu grob berechnet wurde. Für eine nachträgliche Korrektur von $\boldsymbol{R}$ existieren mehrer Algorithmen, die an Hand vieler Beispiele ausführlich diskutiert werden. Zu nennen sind dabei insbesondere die Verfahren von G. Schulz [10], S. Rump [4] mit einer nachfolgenden Defektiteration. Falls die

Berechnung einer Näherungsinversen in einfacher Länge (LU–Zerlegung mit nachfolgender Defektiteration) nicht erfolgreich ist, kann die Approximation der Inversen in doppelter Länge erfolgen. Dabei wird zunächst nach der Methode von Rump eine erste Näherung berechnet, die dann mit Hilfe einer Defektiteration verbessert wird. Insbesondere bei der doppeltlangen Rechnung ist darauf zu achten, daß sowohl nach (2) die günstigste Defektdefinition als auch die dazu passende Version des Rump'schen Verfahrens gewählt wird, vergl. dazu die Seiten 33, ...

Erhält man mit den beschriebenen Verfahren Konvergenz auf der Maschine, so hat man die gesuchte Inverse i.a. maximalgenau approximiert. Daß dies jedoch nicht in allen Fällen garantiert werden kann, zeigen die Beispiele auf den Seiten 26,66. Um mathematisch gesicherte Aussagen zu erhalten, muß daher eine **garantierte Einschließung** der Inversen $\boldsymbol{A}^{-1}$ berechnet werden. Dazu gibt es neben der Einschließung der n Lösungsvektoren der entsprechenden n Gleichungssysteme (7) noch drei weitere Verfahren:

Im ersten Verfahren wird nach den beschriebenen Methoden eine Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ berechnet. Für den dann noch bestehenden Fehler $\boldsymbol{Y}$ aus dem Ansatz $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ ergibt sich mit dem Defekt ${}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{RA}$ die Iterationsgleichung:

$$(8) \qquad \boldsymbol{Y}^{(k+1)} = {}^{l}\boldsymbol{D}(\boldsymbol{R} + \boldsymbol{Y}^{(k)}), \quad k = 0, 1, ...; \qquad \boldsymbol{Y}^{(0)} := {}^{l}\boldsymbol{D}\boldsymbol{R};$$

Berechnet man für ${}^{l}\boldsymbol{D}$ mit dem Akkumulator eine maximalgenaue Einschließung $[{}^{l}\boldsymbol{D}] \ni {}^{l}\boldsymbol{D}$, so erhält man mit $[\boldsymbol{Y}]_{\diamond}^{(0)} := [{}^{l}\boldsymbol{D}] \diamond \boldsymbol{R}$ die intervallmäßige Iterationsgleichung:

$$(9) \qquad [\boldsymbol{Y}]_{\diamond}^{(\nu+1)} = [{}^{l}\boldsymbol{D}] \diamond (\boldsymbol{R} \oplus [\boldsymbol{Y}]_{\diamond}^{(\nu)}), \quad \nu = 0, 1, \dots ,$$

wobei $\oplus$, $\diamond$ als Intervalloperatoren zu verstehen sind. Gilt für ein spezielles k die Beziehung $[\boldsymbol{Y}]_{\diamond}^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]_{\diamond}^{(k)}$, wobei $[\boldsymbol{Y}]_{\diamond}^{(k+1)}$ ganz im Innern von $[\boldsymbol{Y}]_{\diamond}^{(k)}$ liegen muß, so folgt nach dem Fixpunktsatz von Brouwer [4, Seite 9] und mit den Ergebnissen von Rump [9], daß die zur Gleichung (8) zugehörige Fixpunktgleichung

$$(10) \qquad \boldsymbol{Y} = f(\boldsymbol{Y}) \quad \text{mit} \quad f(\boldsymbol{Y}) = {}^{l}\boldsymbol{D}(\boldsymbol{R} + \boldsymbol{Y})$$

genau einen Fixpunkt $\boldsymbol{Y}^{*}$ besitzt, der durch $[\boldsymbol{Y}]_{\diamond}^{(k+1)}$ eingeschlossen wird. Für die eindeutig bestimmte Inverse $\boldsymbol{A}^{-1}$ erhält man dann die gesuchte Einschließung:

$$(11) \qquad \boldsymbol{A}^{-1} = (\boldsymbol{R} + \boldsymbol{Y}^{*}) \in \boldsymbol{R} \oplus [\boldsymbol{Y}]_{\diamond}^{(k+1)}$$

Das zweite Verfahren von G. Alefeld und J. Herzberger wird in [1, Abschnitt 18] angegeben. Zu einer bereits existierenden Einschließung $[\boldsymbol{X}]^{(0)} \ni \boldsymbol{A}^{-1}$ der Inversen $\boldsymbol{A}^{-1}$ wird mit Hilfe einer Mittelpunktsmatrix $\boldsymbol{M}^{(k)} := \boldsymbol{M}([\boldsymbol{X}]^{(k)})$ eine Iterationsgleichung

$$[\boldsymbol{X}]^{(k+1)} = \boldsymbol{M}^{(k)} + [\boldsymbol{X}]^{(k)}(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(k)}), \quad k \geq 0 \tag{12}$$

angegeben, mit der man die Einschließung $[\boldsymbol{X}]^{(0)} \ni \boldsymbol{A}^{-1}$ weiter verbessern kann. Zur Berechnung einer Einschließung muß die Iterationsgleichung (12) intervallmäßig ausgewertet werden:

$$[\boldsymbol{X}]_\diamond^{(k+1)} := \boldsymbol{M}^{(k)} \diamond\!\!\!\!+ [\boldsymbol{X}]_\diamond^{(k)} \diamond\!\!\!\!\cdot \diamond(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(k)}) \tag{13}$$

Dabei sind $\diamond\!\!\!\!+$ $\diamond\!\!\!\!\cdot$ wieder Intervalloperatoren, und die Intervallmatrix $\diamond$ (...) ist dadurch definiert, daß der Ausdruck (...) komponentenweise zunächst im Akkumulator exakt zu berechnen und dann zur jeweils nächsten Rasterzahl auf- bzw. abzurunden ist. Das Verfahren von Alefeld/Herzberger kommt dann zur Anwendung, wenn die mit der Fixpunktiteration nach Brouwer berechnete Einschließung

$$[\boldsymbol{X}]_\diamond^{(0)} := \boldsymbol{R} \diamond\!\!\!\!+ [\boldsymbol{Y}]_\diamond^{(k+1)}$$

aus (11) zu grob berechnet wurde. In einem solchen Fall ist dann nämlich die Voraussetzung $\boldsymbol{A}^{-1} \in [\boldsymbol{X}]_\diamond^{(0)}$ automatisch erfüllt, so daß die Alefeld–Iteration unmittelbar nach Brouwer's Fixpunktiteration erfolgen kann. Weitere Einzelheiten und Anwendungsbeispiele findet man ab Seite 87.

Im dritten Verfahren zur Einschließung einer Inversen $\boldsymbol{A}^{-1}$ wertet man den Crout'schen Algorithmus [3, Seite 176] zur LU–Zerlegung mit Hilfe einer *staggered arithmetic* intervallmäßig aus und erhält damit automatisch die gesuchte, i.a. maximalgenaue Einschließung von $\boldsymbol{A}^{-1}$. Wegen der langen Laufzeiten sollte man dieses Verfahren bei sehr schlecht konditionierten Matrizen jedoch erst dann anwenden, wenn alle anderen Einschließungsmethoden versagt haben. Hinweise zur *staggered arithmetic* findet man ab Seite 69 und weitere Anwendungsbeispiele ab Seite 92.

5 Berechnung einer ersten Näherung

Die näherungsweise Berechnung der Inversen $\boldsymbol{A}^{-1} \in I\!R^{n\times n}$ kann nach der direkten Methode mit Hilfe der LU-Zerlegung von $\boldsymbol{A}$ erfolgen oder mit dem Austauschalgorithmus [11, Seite 51..]. Da dieser im Vergleich zur ersten Methode merklich langsamer ist, werden wir nur die LU-Zerlegung betrachten, die sich durch Anwendung des Gauß-Algorithmus und gleichzeitiger Pivot-Suche nach der Kolonnenmaximumstratigie ergibt [11, Seite 21]. Es gilt dazu nach [11, Seite 18] der folgende

Satz: Zu jeder regulären Matrix $\boldsymbol{A}$ existiert eine durch geeignete Zeilenvertauschungen in $\boldsymbol{I}$ definierte Permutationsmatrix $\boldsymbol{P}$, so daß das Produkt $\boldsymbol{PA}$ eindeutig als Produkt zweier Dreiecksmatrizen geschrieben werden kann:

$$(14) \qquad \boldsymbol{PA} = \boldsymbol{LU}, \quad \text{mit:} \quad \mathrm{Det}(\boldsymbol{PA}) = \mathrm{Det}(\boldsymbol{U}) = |\boldsymbol{U}| \neq 0$$

$\boldsymbol{P}$ läßt sich dabei eindeutig durch einen Permutationsvektor $\boldsymbol{p}$ definieren, dessen k-te Komponente die Nummer des Einheitsvektors angibt, der in $\boldsymbol{P}$ die k-te Zeile festlegt. Der Vektor $\boldsymbol{p}^T = (1, 2, \ldots, n)$ definiert demnach als Permutationsmatrix $\boldsymbol{P}$ die Einheitsmatrix $\boldsymbol{I} = \boldsymbol{P}$. In (14) ist $\boldsymbol{L}$ eine normierte untere und $\boldsymbol{U}$ eine i.a. unnormierte obere Dreiecksmatrix, vergleiche dazu auch Seite 11.

Mit Hilfe der beschriebenen LU-Zerlegung berechnet sich nun die Inverse $\boldsymbol{A}^{-1}$ nach folgendem Algorithmus:

1. Berechne zu gegebenem regulären $\boldsymbol{A}$ die LU-Zerlegung, so daß folgende Produktzerlegung gilt:

$$\boldsymbol{PA} = \boldsymbol{LU}$$

2. Berechne $\boldsymbol{Y}$ in $\boldsymbol{LY} = \boldsymbol{PI}$ spaltenweise.

3. Berechne $\boldsymbol{X}$ in $\boldsymbol{UX} = \boldsymbol{Y}$ spaltenweise, $\longrightarrow$ $\boldsymbol{X} = \boldsymbol{A}^{-1}$

Anmerkungen:

- $\boldsymbol{P}$ wird nicht wirklich gespeichert; die genannten Zeilenvertauschungen werden lediglich in einem Permutationsvektor $\boldsymbol{p}$ festgehalten. Für $\boldsymbol{P}$ gilt:

$$\boldsymbol{P} = \boldsymbol{P}^T = \boldsymbol{P}^{-1}, \qquad |\boldsymbol{P}| = \pm 1$$

- Wir zeigen weiter, daß $\boldsymbol{X}$ tatsächlich die gesuchte Inverse ist:

$$\begin{aligned}\boldsymbol{L}(\boldsymbol{U}\boldsymbol{X}) &= \boldsymbol{L}\boldsymbol{Y} = \boldsymbol{P}\boldsymbol{I}\\ (\boldsymbol{L}\boldsymbol{U})\boldsymbol{X} &= \boldsymbol{L}\boldsymbol{Y} = \boldsymbol{P}\boldsymbol{I}\\ \boldsymbol{P}\boldsymbol{A}\boldsymbol{X} &= \boldsymbol{P}\boldsymbol{I} \mid \cdot \boldsymbol{P}^{-1}(\text{von links})\\ \boldsymbol{A}\boldsymbol{X} &= \boldsymbol{I}\end{aligned}$$

  und wegen der Eindeutigkeit der Inversen folgt: $\boldsymbol{X} = \boldsymbol{A}^{-1}$.

- In einer Matrizen–Gleichung wie $\boldsymbol{U}\boldsymbol{X} = \boldsymbol{Y}$ mit gegebenen $\boldsymbol{U}, \boldsymbol{Y}$ ist die j-te Spalte $\boldsymbol{x}_j$ der gesuchten Matrix $\boldsymbol{X}$ bestimmt **nur** durch $\boldsymbol{U}$ und durch die j-te Spalte $\boldsymbol{y}_j$ von $\boldsymbol{Y}$:

$$(15)\qquad \boldsymbol{U}\boldsymbol{x}_j = \boldsymbol{y}_j; \quad \boldsymbol{x}_j = \begin{pmatrix} x_{1j}\\ x_{2j}\\ \cdot\\ \cdot\\ x_{nj}\end{pmatrix}, \quad \boldsymbol{y}_j = \begin{pmatrix} y_{1j}\\ y_{2j}\\ \cdot\\ \cdot\\ y_{nj}\end{pmatrix}, \; j = 1, \ldots, n;$$

  Damit ist aber die Matrizen–Gleichung $\boldsymbol{U}\boldsymbol{X} = \boldsymbol{Y}$ lediglich eine Zusammenfassung von n linearen Gleichungssystemen der Form (15).

- Bedeutet nun $\boldsymbol{e}_j^{(P)}$ die j-te Spalte der Matrix $\boldsymbol{P}\boldsymbol{I}$, wobei $\boldsymbol{e}_j^{(P)}$ außer Nullen nur eine einzige Eins enthält, so berechnet sich die j-te Spalte $\boldsymbol{x}_j$ der gesuchten Inversen $\boldsymbol{A}^{-1}$ in zwei Schritten:

$$(16)\qquad \boldsymbol{L}\boldsymbol{y}_j = \boldsymbol{e}_j^{(P)}; \quad \boldsymbol{U}\boldsymbol{x}_j = \boldsymbol{y}_j; \quad j = 1, 2, \ldots, n;$$

  Danach sind für jedes j jeweils zwei lineare Gleichungssysteme zu lösen, was zunächst recht aufwendig erscheint. Da jedoch $\boldsymbol{L}$ und $\boldsymbol{U}$ Dreiecksmatrizen mit von Null verschiedenen Diagonalelementen sind, läßt sich in $\boldsymbol{L}\boldsymbol{y}_j = \boldsymbol{e}_j^{(P)}$ das erste Lösungselement y_{1j} direkt hinschreiben, und alle restlichen Elemente $y_{2j}, y_{3j}, \ldots, y_{nj}$ können sukzessive berechnet werden (**Vorwärtseinsetzen**). Entsprechend läßt sich in $\boldsymbol{U}\boldsymbol{x}_j = \boldsymbol{y}_j$ x_{nj} sofort angeben, und alle restlichen Lösungselemente können wieder sukzessiv durch **Rückwärtseinsetzen** bestimmt werden.

Mit (14),(16) steht somit ein recht effektiver Algorithmus zur näherungsweisen Berechnung von $\boldsymbol{X} = \boldsymbol{A}^{-1}$ zur Verfügung, der durch Aufruf der Prozedur

INV_LU(var A,R : Lrmatrix; var Err : integer);
exportiert durch linv

ausgewertet werden kann. Dabei erhält man auch für höhere Dimensionen n noch brauchbare Näherungen $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$, wenn $\boldsymbol{A}$ nur hinreichend gut konditioniert ist. Als Beispiel für schlecht konditionierte Matrizen wählen wir die Boothroyd/Dekker-Matrizen $\boldsymbol{D}$ [13, Seite 25]. Mit $\boldsymbol{D} = (d_{ij})$, $\boldsymbol{b} = (b_i)$ lauten die Definitionsgleichungen:

$$\begin{aligned} d_{ij} &:= \binom{n+i-1}{i-1} \cdot \binom{n-1}{n-j} \cdot \frac{n}{i+j-1}; \qquad d_{ij} \in \mathbb{N} \\ b_i &:= i; \qquad 1 \le i,j \le n; \end{aligned}$$

Nach [13] gelten für die Inverse $\boldsymbol{D}^{-1}$ die folgenden Aussagen:

$$\begin{gathered} \boldsymbol{D}^{-1} = (di_{ij}); \qquad di_{ij} := (-1)^{i+j} \cdot d_{ij} \\ \boldsymbol{D}^{-1}\boldsymbol{D} = \boldsymbol{D}\,\boldsymbol{D}^{-1} = \boldsymbol{I}, \qquad \boldsymbol{I}: \text{ Einheitsmatrix;} \\ \mathrm{Det}(\boldsymbol{D}) = \mathrm{Det}(\boldsymbol{D}^{-1}) = 1 \end{gathered}$$

Die Boothroyd/Dekker-Matrizen $\boldsymbol{D}$ erhält man durch Aufruf der Prozedur

DEKKER(var D : Lrmatrix; var b : Lrvector);
exportiert durch lmv_exap

wobei die ganzzahligen Dekker-Matrix-Elemente für $n \le 27$ exakt berechnet werden. Im Fall $n > 27$ erfolgt eine Fehlermeldung, da dann die $\boldsymbol{D}[i,j]$ im 21-stelligen Longreal-Format nicht mehr rundungsfehlerfrei darstellbar sind. Im Bereich $2 \le n \le 27$ kann man jetzt mit INV_LU(D,R,Err) Approximationen $\boldsymbol{R} \approx \boldsymbol{D}^{-1}$ der inversen Dekker–Matrizen berechnen. In folgender Tabelle sind zu gegebenem n neben der Konditionszahl $\varkappa(\boldsymbol{D})$ für jedes berechnete $\boldsymbol{R}$ zu den in (2) definierten Defekten die entsprechenden Normen $\|{}^{l}\boldsymbol{D}\|$, $\|{}^{r}\boldsymbol{D}\|$ und zum Vergleich eine Oberschranke $\varepsilon(\boldsymbol{R})$ des relativen Fehlers

$$\varepsilon[i,j] := \left| \frac{\boldsymbol{D}^{-1}[i,j] - \boldsymbol{R}[i,j]}{\boldsymbol{D}^{-1}[i,j]} \right|, \qquad \max_{i,j} \left(\varepsilon[i,j]\right) \le \varepsilon(\boldsymbol{R})$$

angegeben:

Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{D}^{-1}$ einer Dekker Matrix $\boldsymbol{D}$				
Prozeduraufruf: INV_LU(D, R, Err)				
$\varepsilon(\boldsymbol{R}), \Vert{}^l\boldsymbol{D}\Vert, \Vert{}^r\boldsymbol{D}\Vert, \varkappa(\boldsymbol{D})$ in Abhängigkeit von $n, \boldsymbol{R}$				
n	$\varepsilon(\boldsymbol{R})$	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$	$\varkappa(\boldsymbol{D})$
2	0	0	0	$3.60 \cdot 10^{+1}$
3	$1.34 \cdot 10^{-20}$	$1.50 \cdot 10^{-18}$	$1.50 \cdot 10^{-19}$	$2.03 \cdot 10^{+3}$
4	$2.18 \cdot 10^{-19}$	$2.76 \cdot 10^{-17}$	$1.80 \cdot 10^{-18}$	$1.13 \cdot 10^{+5}$
5	$1.51 \cdot 10^{-17}$	$1.06 \cdot 10^{-15}$	$7.33 \cdot 10^{-17}$	$7.29 \cdot 10^{+6}$
6	$9.13 \cdot 10^{-17}$	$4.69 \cdot 10^{-14}$	$5.78 \cdot 10^{-16}$	$4.33 \cdot 10^{+8}$
7	$3.15 \cdot 10^{-15}$	$2.67 \cdot 10^{-12}$	$2.95 \cdot 10^{-14}$	$2.83 \cdot 10^{+10}$
8	$2.75 \cdot 10^{-15}$	$4.34 \cdot 10^{-11}$	$4.09 \cdot 10^{-14}$	$1.72 \cdot 10^{+12}$
9	$3.55 \cdot 10^{-12}$	$2.25 \cdot 10^{-9}$	$7.68 \cdot 10^{-11}$	$1.13 \cdot 10^{+14}$
10	$1.05 \cdot 10^{-10}$	$7.36 \cdot 10^{-8}$	$4.16 \cdot 10^{-9}$	$6.92 \cdot 10^{+15}$
11	$5.16 \cdot 10^{-10}$	$8.98 \cdot 10^{-6}$	$1.09 \cdot 10^{-8}$	$4.52 \cdot 10^{+17}$
12	$4.65 \cdot 10^{-8}$	$1.30 \cdot 10^{-4}$	$3.07 \cdot 10^{-6}$	$2.80 \cdot 10^{+19}$
13	$4.35 \cdot 10^{-8}$	$7.03 \cdot 10^{-3}$	$7.61 \cdot 10^{-7}$	$1.83 \cdot 10^{+21}$
14	$1.88 \cdot 10^{-5}$	$1.72 \cdot 10^{-1}$	$3.77 \cdot 10^{-3}$	$1.14 \cdot 10^{+23}$
15	$1.11 \cdot 10^{-4}$	$1.19 \cdot 10^{+2}$	$2.83 \cdot 10^{-2}$	$7.42 \cdot 10^{+24}$
16	$1.77 \cdot 10^{-3}$	$3.71 \cdot 10^{+3}$	$4.00 \cdot 10^{+0}$	$4.64 \cdot 10^{+26}$
17	$1.83 \cdot 10^{-1}$	$2.35 \cdot 10^{+5}$	$3.15 \cdot 10^{+2}$	$3.02 \cdot 10^{+28}$
18	$9.62 \cdot 10^{-1}$	$3.95 \cdot 10^{+5}$	$4.07 \cdot 10^{+3}$	$1.89 \cdot 10^{+30}$
19	$1.19 \cdot 10^{+0}$	$2.34 \cdot 10^{+7}$	$1.26 \cdot 10^{+4}$	$1.23 \cdot 10^{+32}$
20	$1.19 \cdot 10^{+0}$	$7.43 \cdot 10^{+9}$	$2.63 \cdot 10^{+4}$	$7.72 \cdot 10^{+33}$
21	$1.01 \cdot 10^{+0}$	$2.30 \cdot 10^{+10}$	$8.59 \cdot 10^{+4}$	$5.01 \cdot 10^{+35}$
22	$1.01 \cdot 10^{+0}$	$1.61 \cdot 10^{+11}$	$6.13 \cdot 10^{+5}$	$3.16 \cdot 10^{+37}$
23	$1.01 \cdot 10^{+0}$	$1.51 \cdot 10^{+12}$	$4.74 \cdot 10^{+5}$	$2.05 \cdot 10^{+39}$
24	$1.01 \cdot 10^{+0}$	$4.28 \cdot 10^{+11}$	$4.59 \cdot 10^{+5}$	$1.29 \cdot 10^{+41}$
25	$1.01 \cdot 10^{+0}$	$8.05 \cdot 10^{+14}$	$2.57 \cdot 10^{+6}$	$8.34 \cdot 10^{+42}$
26	$1.01 \cdot 10^{+0}$	$2.98 \cdot 10^{+14}$	$8.21 \cdot 10^{+5}$	$5.27 \cdot 10^{+44}$
27	$1.01 \cdot 10^{+0}$	$1.66 \cdot 10^{+16}$	$1.85 \cdot 10^{+6}$	$3.41 \cdot 10^{+46}$

An Hand der relativen Fehler $\varepsilon(\boldsymbol{R})$ erkennt man zunächst, daß die mit der LU–Zerlegung berechneten Näherungsinversen $\boldsymbol{R} \approx \boldsymbol{D}^{-1}$ mit wachsendem n völlig unbrauchbar werden. Zum gleichen Ergebnis kommt man auch, wenn man die Normen $\Vert{}^l\boldsymbol{D}\Vert, \Vert{}^r\boldsymbol{D}\Vert$ der beiden durch (2) gegebenen Defekte betrachtet. Dabei fällt jedoch auf, daß mit wachsendem n die Norm $\|{}^r\boldsymbol{D}\|$ um viele Größenordnungen kleiner ausfällt, als $\|{}^l\boldsymbol{D}\|$. Man kann daher vermuten, daß eine Defektiteration zur Approximation der Inversen $\boldsymbol{D}^{-1}$ mit der Definition ${}^r\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{AR}$ die besten Ergebnisse liefern wird.

Da wir mit Hilfe der LU–Zerlegung, insbesondere bei zu schlecht konditionierten Matrizen, keine ausreichend genauen Näherungsinversen erhalten, um damit eine Einschließung der Inversen berechnen zu können, werden deshalb in den folgenden Abschnitten verschiedene Verfahren diskutiert, mit denen man schon bekannte, aber zu ungenaue berechnete Näherungsinverse noch nachträglich verbessern kann. Dabei zeigt sich, daß es im Wesentlichen zwei sehr effektive Verfahren gibt, mit denen man die in der Praxis auftretenden Fälle erfolgreich behandeln kann.

Es soll aber schon an dieser Stelle ausdrücklich betont werden, daß auch das beste Korrekturverfahren scheitert, wenn man selbst bei noch kleiner Dimension n die zu invertierende Matrix $\boldsymbol{A}$ gezielt so konstruiert, daß sie extrem schlecht konditioniert ist. Wie werden später sehen, wie man auch noch in diesen Fällen, allerdings unter **erheblichem** Rechenaufwand, mit Hilfe einer *staggered arithmetic* eine sogar maximalgenaue Näherungsinverse berechnen kann.

Abschließend noch eine Anmerkung zur Berechnung der in der vorigen Tabelle angegebenen Konditionszahlen, die bekanntlich wie folgt zu berechnen sind:

$$\varkappa := \|\boldsymbol{D}\| \cdot \|\boldsymbol{D}^{-1}\|$$

Dabei benötigt man also die Inversen $\boldsymbol{D}^{-1}$ der Boothroyd/Dekker-Matrizen, die man durch Aufruf der folgenden Prozedur erhält:

DEKKER_INV(var A : Lrmatrix); exportiert durch lmv_exap

Die Ausgangsgröße A liefert bis $n = 27$ das exakte Ergebnis: $\boldsymbol{A} = \boldsymbol{D}^{-1}$.

6 Nachträgliche Korrekturen

6.1 Überblick

Zur Verbesserung einer zu ungenau berechneten Näherungsinversen diskutieren wir drei Möglichkeiten:

1. Nach G. Schulz [10, 13] wird ein Iterationsverfahren angegeben, das i.a. zwar quadratisch konvergiert, in einem speziellen Fall jedoch nicht gegen die gesuchte Inverse. Die Gründe für das Versagen werden erläutert; Sie liefern Anhaltspunkte zur Formulierung besserer Korrekturverfahren, wie sie in den nachfolgenden Punkten beschrieben werden.

2. Das Verfahren von Rump [3] liefert in doppeltlanger Rechnung besonders bei schlechtkonditionierten Matrizen eine sehr gute Verbesserung einer z.B. durch LU–Zerlegung berechneten ersten Näherungsinversen. Wegen des größeren Aufwandes empfiehlt sich diese Methode jedoch nur am Anfang eines Korrekturverfahrens, bei dem die letzten Schritte vorzugsweise mit einer schnelleren Defektiteration durchgeführt werden sollten. An Hand der Boothroyd/Dekker Matrizen wird auf Seite 35 zunächst gezeigt, daß sich der relative Höchstfehler bei doppeltlanger Rechnung durch das Verfahren von Rump nicht verbessern läßt. Im Gegensatz dazu erhält man nach Seite 37 eine Verkleinerung der Defektnormen um viele Größenordnungen insbesondere dann, wenn man nach (2) die günstigste der beiden möglichen Defektdefinitionen wählt.

3. Es wird eine Defektiteration in einfacher und doppelter Länge angegeben, wobei der Defekt selbst mit Hilfe des in **PASCAL-XSC** verfügbaren Skalarproduktes **maximalgenau** berechnet werden kann. Bei schlechtkonditionierten Matrizen ist Konvergenz jedoch nur dann möglich, wenn die nach (2) passende Defektdefinition gewählt wird, vergl. dazu Seite 64. Im Vergleich zum Verfahren von Schulz ist die Defektiteration bei gleichem Aufwand viel leistungsfähiger, da sie die zu invertierende Matrix bei jedem Iterationsschritt direkt in die Rechnung mit einbezieht. Da als wesentliche Operationen in jedem Schritt nur zwei Matrixmultiplikationen anfallen, ist die Defektiteration deutlich schneller als das Rump'sche Verfahren und sollte stets nach diesem zur Anwendung kommen, um bei Konvergenz die Berechnung einer **maximalgenauen** Näherungsinversen zu ermöglichen.

Die unter 2. und 3. angegebenen Methoden lassen sich bei allen in der Praxis auftretenden schlechtkonditionierten Matrizen erfolgreich anwenden. Man kann jedoch sehr schlecht konditionierte Matrizen gezielt so konstruieren, daß alle hier beschriebenen Iterationsverfahren versagen. In diesen Fällen erhält man ohne Korrekturverfahren eine sogar maximalgenaue Näherungsinverse, wenn man das Verfahren der LU–Zerlegung von Seite 19 unter wesentlich größerem Aufwand in *staggered arithmetic* durchführt und das gefundene Langzahlergebnis zur nächsten Longreal–Zahl rundet.

6.2 Das Verfahren von Schulz

In [13, Seite 207,..],[10] wird eine Methode angegeben, mit der G. Schulz die Inverse einer regulären Matrix iterativ berechnet hat. Ausgehend von einer nicht zu schlechten Näherungsinversen $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ wird zunächst nach (2) die zugehörige Defektmatrix (Druckfehler in [13] Seite 208, Gleichung (11) und Zeile 5 von oben beachten!)

$$ {}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{A} \tag{17} $$

ausgewertet. Mit **PASCAL-XSC** lassen sich dabei die Elemente von ${}^{l}\boldsymbol{D}$ mit Hilfe des exakten Skalarproduktes mit nur einer einzigen Rundung auf der letzten Mantissenstelle berechnen. Die Iterationsvorschrift lautet dann:

$$ \boxed{\begin{array}{ll} \boldsymbol{M}^{(0)} := {}^{l}\boldsymbol{D}; & \boldsymbol{R}^{(1)} := \boldsymbol{R}^{(0)} + \boldsymbol{M}^{(0)}\boldsymbol{R}^{(0)} \\ \boldsymbol{M}^{(1)} := \boldsymbol{M}^{(0)}\boldsymbol{M}^{(0)}; & \boldsymbol{R}^{(2)} := \boldsymbol{R}^{(1)} + \boldsymbol{M}^{(1)}\boldsymbol{R}^{(1)} \\ \boldsymbol{M}^{(2)} := \boldsymbol{M}^{(1)}\boldsymbol{M}^{(1)}; & \boldsymbol{R}^{(3)} := \boldsymbol{R}^{(2)} + \boldsymbol{M}^{(2)}\boldsymbol{R}^{(2)} \end{array}} \tag{18} $$

Die numerische Auswertung von (17),(18) ist außerordentlich lehrreich, zumal man wegen der quadratischen Konvergenz des Verfahrens eine erfolgreiche Nachkorrektur der aus der LU–Zerlegung gewonnenen Näherungsinversen $\boldsymbol{R}^{(0)}$ erwartet. Die Iteration wird abgebrochen, wenn im k-ten Schritt gilt:

$$ \boldsymbol{R}^{(k)} \equiv \boldsymbol{R}^{(k-1)}, \qquad k = 1, 2, \ldots \tag{19} $$

Die mit der **PASCAL-XSC** BCD–Version im Longreal–Format berechneten Ergebnisse sind in nachfolgender Tabelle zusammengestellt:

Näherungsinverse $R^{(k)} \approx D^{-1}$ einer Dekker Matrix D				
Iterationsverfahren von Schulz: Abbruch, falls: $R^{(k)} = R^{(k-1)}$				
n	$D^{-1}[1,n]$	$R^{(k)}[1,n]$	|Rel. Fehler| $\leq$	k
3	1	1	0	2
5	1	1	0	2
7	1	1	0	2
9	1	1	0	2
11	1	1	0	3
13	1	1	0	3
15	1	1	0	4
17	1	1.00000000000071832072	$7.19 \cdot 10^{-13}$	6

Auf den ersten Blick scheinen die Ergebnisse sogar perfekt zu sein, da bis zur Dimension $n = 15$ das Iterationsverfahren von Schulz die inversen Elemente $\boldsymbol{D}^{-1}[1,n] = \boldsymbol{R}^{(k)}[1,n]$ exakt liefert, und der geringe relative Fehler $7.19 \cdot 10^{-13}$ im Falle $n = 17$ erscheint durchaus akzeptabel. Das nachfolgende Beispiel zeigt jedoch, daß das Iterationsverfahren selbst für eine gut konditionierte Matrix $\boldsymbol{A}$ Ergebnisse liefert, die sich von den exakten Elementen der Inversen $\boldsymbol{A}^{-1}$ sogar um viele Größenordnungen unterscheiden können:

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-41} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{A}^{-1} := \begin{pmatrix} 1 & -1 & \frac{1}{3}(1-10^{-21}) & 10^{-41} \\ 0 & 1 & -\frac{1}{3} & \frac{1}{3}\cdot 10^{-21} \\ 0 & 0 & \frac{1}{3} & -\frac{1}{3} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Als Näherung der Inversen $\boldsymbol{A}^{-1}$ betrachten wir die Matrix

$$\boldsymbol{R}^{(0)} := \begin{pmatrix} 1.01 & -1.02 & 0.333 & 10^{-41} \\ 0 & 1.01 & -0.333 & 0.333 \cdot 10^{-21} \\ 0 & 0 & 0.333 & -0.333 \\ 0 & 0 & 0 & 1 \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

die durch einen beliebigen Algorithmus gegeben sei. Mit der in (17) definierten Defektmatrix $\boldsymbol{M}$ erhält man mit der Iterationsvorschrift (18) von Schulz nach 6 Schritten das Ergebnis:

$$\boldsymbol{R}^{(6)} := \begin{pmatrix} 1 & -1 & \frac{1}{3}(1-10^{-21}) & -3.06657999999893333333 \cdot 10^{-24} \\ 0 & 1 & -\frac{1}{3}(1-10^{-21}) & 3.36306663666666663333 \cdot 10^{-22} \\ 0 & 0 & \frac{1}{3}(1-10^{-21}) & -\frac{1}{3}(1-10^{-21}) \\ 0 & 0 & 0 & 1 \end{pmatrix} \neq \boldsymbol{A}^{-1}$$

Zur kürzeren Schreibweise benutzen wir dabei die Identität:

$$\frac{1}{3} \cdot (1 - 10^{-21}) = 0.333333333333333333333$$

Der Vergleich der Matrix $\boldsymbol{R}^{(6)}$ mit der Inversen $\boldsymbol{A}^{-1}$ zeigt, daß $\boldsymbol{R}^{(6)}$ bis auf die Elemente $\boldsymbol{R}^{(6)}[1,4]$, $\boldsymbol{R}^{(6)}[2,4]$ eine maximalgenaue Approximation von $\boldsymbol{A}^{-1}$ ist. Während $\boldsymbol{R}^{(6)}[2,4]$ noch als recht grobe Näherung aufgefaßt werden kann, unterscheidet sich jedoch $\boldsymbol{R}^{(6)}[1,4] = -3.0\ldots \cdot 10^{-24}$ vom exakten Wert 10^{-41} neben dem Vorzeichen um stolze 17 Größenordnungen. Mögliche Ursachen für das Scheitern des Verfahrens von Schulz sind die unvermeidbaren Rundungsfehler im benutzten Gleitkommasystem und die sehr unterschiedlichen Größenordnungen der Elemente von $\boldsymbol{A}^{-1}$.

Diese in der Praxis nicht zu umgehenden Ursachen sind aber nicht der wirkliche Grund für das Versagen des beschriebenen Verfahrens, sondern das Scheitern ist vielmehr durch die Konstruktion des Iterations-Algorithmus selbst gegeben! In (17) wird nämlich der Bezug zur zu invertierenden Matrix $\boldsymbol{A}$ durch die i.a. fehlerhaft berechnete Defektmatrix $\boldsymbol{M}$ **nur einmal**, und zwar am Anfang des Algorithmus, hergestellt. In der nachfolgenden Iteration (18) werden dann die anfangs produzierten Rundungsfehler bei ungünstigem $\boldsymbol{A}$ in jedem Schritt verstärkt, so daß einige Elemente von $\boldsymbol{A}^{-1}$ extrem falsch berechnet werden, obwohl das Iterationsverfahren auf dem Rechner (sogar quadratisch) konvergiert. Daraus ergibt sich der folgende Merksatz:

> *Ein auf Gleitkommarechnungen aufgebautes Iterationsverfahren ist in extremen Situationen (schlecht konditionierte Matrizen) nur dann erfolgreich, wenn möglichst* **bei jedem Schritt** *der Bezug zur Wirklichkeit (Berechnung der Defektmatrix) hergestellt wird.*

Anschließend gleich ein weiterer fast trivialer Merksatz:

> *In jedem Iterationsschritt sollten möglichst wenige Gleitkomma-Operationen benutzt werden, die Rundungsfehler oder wesentliche Stellenauslöschung bewirken. (Defekte stets im Akku berechnen!)*

Durch diese guten Vorsätze motiviert, wenden wir uns im folgenden Abschnitt einem Verfahren zu, mit dem S. Rump eine zu grob berechnete Näherungsinverse, insbesondere bei schlechtkonditionierten Matrizen, sehr erfolgreich verbessern konnte.

6.3 Das Verfahren von Rump

Das Verfahren von Rump basiert auf folgender Idee:

Hat man zu einer vorgegebenen regulären, aber schlechtkonditionierten Matrix $\boldsymbol{A}$ bereits eine grobe Näherungsinverse $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ berechnet, so ist die Produktmatrix $\boldsymbol{R}^{(0)}\boldsymbol{A} \approx \boldsymbol{I}$ schon sehr viel besser konditioniert und läßt sich deshalb erfolgreicher invertieren. Wegen der einfachen Beziehung

$$\boldsymbol{A}^{-1} = (\boldsymbol{R}^{(0)}\boldsymbol{A})^{-1}\boldsymbol{R}^{(0)} \tag{20}$$

erhält man daher eine verbesserte Näherung, wenn man die neu berechnete Näherungsinverse von $\boldsymbol{R}^{(0)}\boldsymbol{A}$ von rechts mit der ursprünglichen Näherung $\boldsymbol{R}^{(0)}$ multipliziert. Das Verfahren von Rump lautet im einzelnen:

Verfahren von Rump, Version **links**:

1. Berechnung einer 0. Nährungsinversen $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer LU–Zerlegung.
2. Berechnung von $\boldsymbol{R}^{(0)} \odot \boldsymbol{A}$ im Longreal–Format.
3. Berechnung einer Näherung $\boldsymbol{S} \approx (\boldsymbol{R}^{(0)} \odot \boldsymbol{A})^{-1}$ mit Hilfe einer LU–Zerlegung von $(\boldsymbol{R}^{(0)} \odot \boldsymbol{A})$.
4. Berechnung von $\boldsymbol{R}^{(1)} := \boldsymbol{S} \odot \boldsymbol{R}^{(0)}$ im Longreal–Format; $\boldsymbol{R}^{(1)} \approx \boldsymbol{A}^{-1}$ ist im Vergleich zu $\boldsymbol{R}^{(0)}$ eine i.a. stark verbesserte Näherungsinverse.

Der Operator $\odot$ beschreibt die im Longreal–Format definierte Matrixmultiplikation, die auf dem Rechner i.a. fehlerbehaftete, aber stets maximalgenaue Ergebnisse liefert.

Natürlich gilt neben (20) völlig gleichberechtigt auch die Gleichung

$$\boldsymbol{A}^{-1} = \boldsymbol{R}^{(0)}(\boldsymbol{A}\boldsymbol{R}^{(0)})^{-1} \tag{21}$$

Bezieht man sich dann auf (21), so läßt sich das Verfahren von Rump ganz analog auch in der folgenden Version 'rechts' formulieren. Wir benutzen die Bezeichnung 'rechts', da die zu invertierende Matrix $\boldsymbol{A}$ jetzt von rechts mit $\boldsymbol{R}^{(0)}$ multipliziert wird:

Verfahren von Rump, Version **rechts**:

1. Berechnung einer 0. Nährungsinversen $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer LU–Zerlegung.

2. Berechnung von $\boldsymbol{A} \odot \boldsymbol{R}^{(0)}$ im Longreal–Format.

3. Berechnung einer Näherung $\boldsymbol{S} \approx (\boldsymbol{A} \odot \boldsymbol{R}^{(0)})^{-1}$ mit Hilfe einer LU–Zerlegung von $(\boldsymbol{A} \odot \boldsymbol{R}^{(0)})$.

4. Berechnung von $\boldsymbol{R}^{(1)} := \boldsymbol{R}^{(0)} \odot \boldsymbol{S}$ im Longreal–Format; $\boldsymbol{R}^{(1)} \approx \boldsymbol{A}^{-1}$ ist im Vergleich zu $\boldsymbol{R}^{(0)}$ eine i.a. stark verbesserte Näherungsinverse.

Die beiden Algorithmen des Rump'schen Verfahrens werden realisiert durch Aufruf der **PASCAL-XSC** Prozedur

```
INV_RUMP( var A : Lrmatrix; illcd,factor : integer;
          var R : Lrmatrix; var Err : integer);
```

exportiert durch linv

Die Eingangsgröße $\boldsymbol{A}$ ist die zu invertierende reguläre Matrix. Der Parameter illcd steuert den internen Algorithmus:

illcd=0: Die Näherung $\boldsymbol{R}$ wird nur mit einer LU–Zerlegung berechnet.

illcd=1: Das Verfahren von Rump wird angewandt.

Im Falle **illcd=1** bestimmt der Eingangsparameter factor die Version des Rump'schen Verfahrens:

factor=0: Verfahren von Rump, Version **links**, vergl. Seite 28.

factor=1: Verfahren von Rump, Version **rechts**, vergl. Seite 29

Die Ausgangsgröße $\boldsymbol{R}$ ist die berechnete Näherungsinverse, und Err liefert den Fehlercode, wobei Err=0 signalisiert, daß der Algorithmus fehlerfrei ausgewertet wurde.

Als Anwendung betrachten wir die zu invertierenden Matrizen $\boldsymbol{A}, \boldsymbol{B}$:

$$\boldsymbol{A} \quad := \quad \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix}$$

$$\boldsymbol{B} \quad := \quad \begin{pmatrix} 1 & 1 & 1+10^{-19} & 1 \\ 1 & 1+10^{-19} & 1 & 10^{-40} \\ 1+10^{-19} & 1 & 1 & 10^{-40} \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

In nachfolgender Tabelle sind für U $= \boldsymbol{A}$ bzw. U $= \boldsymbol{B}$ bzgl. der mit Hilfe von INV_RUMP(U,illcd,factor,R,Err) berechneten Näherungsinversen $\boldsymbol{R}$ die Normen der nach (2) definierten Defekte ${}^l\boldsymbol{D}, {}^r\boldsymbol{D}$ zusammengestellt:

INV_RUMP(U,illcd,factor,R,Err); $\boldsymbol{U} = \boldsymbol{A}$ bzw. $\boldsymbol{U} = \boldsymbol{B}$						
Berechnung der Defektnormen $\Vert{}^l\boldsymbol{D}\Vert$, $\Vert{}^r\boldsymbol{D}\Vert$ nach Definition (2), Seite 12						
U	illcd=0		illcd=1			
	Nur LU-Zerlegung		Rump links; factor=0		Rump rechts; factor=1	
	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$
$\boldsymbol{A}$	$8.0 \cdot 10^{-1}$	$4.0 \cdot 10^{-20}$	$4.0 \cdot 10^{+0}$	$5.6 \cdot 10^{+0}$	$8.0 \cdot 10^{-20}$	$4.0 \cdot 10^{-40}$
$\boldsymbol{B}$	$4.0 \cdot 10^{-2}$	$1.4 \cdot 10^{-2}$	$1.1 \cdot 10^{-19}$	$2.0 \cdot 10^{-1}$	$4.0 \cdot 10^{-2}$	$3.6 \cdot 10^{-4}$

Zunächst finden wir für $\boldsymbol{A}$ und $\boldsymbol{B}$ den auf Seite 13 formulierten Merksatz bestätigt, nach dem sich für ein und dieselbe Näherung $\boldsymbol{R}$ die Normen der Defekte ${}^l\boldsymbol{D}$, ${}^r\boldsymbol{D}$ um viele Größenordnungen unterscheiden können.

Bei der Matrix $\boldsymbol{A}$ fällt auf, daß das Verfahren von Rump die Norm $\Vert{}^r\boldsymbol{D}\Vert$ im Vergleich zur Approximation nur mit der LU–Zerlegung um 20 Größenordungen verbessert, wenn man die Version 'rechts' (factor=1) des Verfahrens von Rump benutzt. Im Gegensatz dazu erhält man bei der Matrix $\boldsymbol{B}$ eine entsprechende Verbesserung der Norm $\Vert{}^l\boldsymbol{D}\Vert$ um 17 Größenordnungen, wenn man die Version 'links'(factor=0) verwendet. Wir formulieren daher den zweiten Merksatz:

Wählt man beim Rump'schen Verfahren die geeignete Version, so kann im Vergleich zur einfacheren LU–Approximation die Defektnorm ganz wesentlich verkleinert werden.

Die letzte Tabelle liefert noch den für beide Matrizen $\boldsymbol{A}$, $\boldsymbol{B}$ gültigen dritten Merksatz, der sich für die numerische Anwendung als sehr nützlich erweisen wird:

> *Für die nach dem Verfahren von Rump berechnete Näherung $\boldsymbol{R}$ erhält man die kleinste Defektnorm, wenn man zur gewählten Rump–Version die entsprechende Defektdefinition aus Gleichung (2) benutzt.*

Beispielsweise erhält man also für die mit der Rump–Version 'links' berechnete Näherung $\boldsymbol{R}$ die kleinere Defektnorm, wenn sich diese auf den entsprechenden Defekt ${}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{RA}$ bezieht. Obiger Merksatz wird auch durch die Defektnormen bestätigt, die in folgender Tabelle für die Approximationen $\boldsymbol{R}$ der Boothroyd/Dekker Matrizen $\boldsymbol{D}$ zusammengestellt sind:

INV_RUMP(D,illcd,factor,R,Err); D: Dekkermatrix der Dimension n						
Berechnung der Defektnormen $\|{}^{l}\boldsymbol{D}\|$, $\|{}^{r}\boldsymbol{D}\|$ nach Definition (2) von Seite 12						
n	illcd=0		illcd=1			
	Nur LU-Zerlegung		Rump links; factor=0		Rump rechts; factor=1	
	$\|{}^{l}\boldsymbol{D}\|$	$\|{}^{r}\boldsymbol{D}\|$	$\|{}^{l}\boldsymbol{D}\|$	$\|{}^{r}\boldsymbol{D}\|$	$\|{}^{l}\boldsymbol{D}\|$	$\|{}^{r}\boldsymbol{D}\|$
3	$1.5 \cdot 10^{-18}$	$1.5 \cdot 10^{-19}$	0	0	0	0
5	$1.1 \cdot 10^{-15}$	$7.4 \cdot 10^{-17}$	$2.7 \cdot 10^{-17}$	$1.3 \cdot 10^{-17}$	$8.1 \cdot 10^{-16}$	$1.8 \cdot 10^{-16}$
7	$2.7 \cdot 10^{-12}$	$3.0 \cdot 10^{-14}$	$2.2 \cdot 10^{-16}$	$1.3 \cdot 10^{-15}$	$6.8 \cdot 10^{-13}$	$1.9 \cdot 10^{-14}$
9	$2.3 \cdot 10^{-9}$	$7.7 \cdot 10^{-11}$	$1.1 \cdot 10^{-10}$	$7.9 \cdot 10^{-10}$	0	0
11	$9.0 \cdot 10^{-6}$	$1.1 \cdot 10^{-8}$	$4.7 \cdot 10^{-6}$	$1.2 \cdot 10^{-5}$	$1.1 \cdot 10^{-5}$	$8.8 \cdot 10^{-7}$
15	$1.2 \cdot 10^{+2}$	$2.9 \cdot 10^{-2}$	$5.0 \cdot 10^{+0}$	$2.5 \cdot 10^{+2}$	$1.3 \cdot 10^{+1}$	$1.8 \cdot 10^{+1}$
17	$2.4 \cdot 10^{+5}$	$3.2 \cdot 10^{+2}$	$2.9 \cdot 10^{+4}$	$1.4 \cdot 10^{+5}$	$5.1 \cdot 10^{+4}$	$2.8 \cdot 10^{+4}$
20	$7.5 \cdot 10^{+9}$	$2.7 \cdot 10^{+4}$	$3.4 \cdot 10^{+9}$	$8.2 \cdot 10^{14}$	$8.2 \cdot 10^{10}$	$1.5 \cdot 10^{+9}$
25	$8.1 \cdot 10^{14}$	$2.6 \cdot 10^{+6}$	$6.1 \cdot 10^{13}$	$2.7 \cdot 10^{21}$	$7.3 \cdot 10^{19}$	$7.6 \cdot 10^{17}$
27	$1.7 \cdot 10^{16}$	$1.9 \cdot 10^{+6}$	$2.2 \cdot 10^{16}$	$1.3 \cdot 10^{23}$	$7.5 \cdot 10^{22}$	$7.1 \cdot 10^{20}$

Berechnet man die Approximationen $\boldsymbol{R} \approx \boldsymbol{D}^{-1}$ nur mit der LU–Zerlegung, so fällt auf, daß nach obiger Tabelle für die Defektnormen stets gilt:

$$\|{}^{r}\boldsymbol{D}\| < \|{}^{l}\boldsymbol{D}\| \tag{22}$$

Der vorherigen Tabelle entnimmt man weiter, daß durch das Verfahren von Rump bei den Dekker–Matrizen für $n \geq 11$ die Defektnormen im Vergleich zur Approximation mit der LU–Zerlegung nicht mehr wesentlich verkleinert werden, so daß gerade für die schlechter konditionierten Matrizen keine Verbesserung der Defekte erreicht wird. Auch aus Laufzeitgründen ist es daher sinnvoll, bei einfacher Rechnung die erste Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ nur mit Hilfe der LU–Zerlegung zu berechnen. Im nächsten Abschnitt werden wir zeigen, daß gerade bei den schlechtkonditionierten Matrizen eine gravierende Verbesserung der Defekte erreicht wird, wenn man das Verfahren von Rump in doppelter Länge des benutzten Datenformats anwendet.

6.3.1 Doppeltlange Näherungen

Wir wenden uns jetzt dem im Toolbox-Band I [3, Seite 192] angegebenen Algorithmus von Rump zu, nach dem eine Näherungsinverse in doppelter Länge des benutzten Datenformats berechnet wird. Dabei hofft man natürlich, den auf Seite 21 definierten relativen Fehler auf den maximalen relativen Fehler $5 \cdot 10^{-42}$ einer Grundoperation im doppeltlangen Longrealformat drücken zu können. Außerdem erwartet man, daß gerade bei den schlechtkonditionierten Matrizen auch die Defekte entscheidend verbessert werden können.

Geht man zunächst wieder von der einfachen Beziehung

$$A^{-1} = (RA)^{-1} R \tag{23}$$

aus, wobei die Matrix $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ eine mit Hilfe der LU–Zerlegung von $\boldsymbol{A}$ berechnete 0. Näherung bedeutet, so lautet der vorgeschlagene Algorithmus:

Verfahren von Rump, Version **links**:

1. Berechnung einer 0. Nährungsinversen $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer LU–Zerlegung.
2. Berechnung von $\boldsymbol{R} \odot \boldsymbol{A}$ im Longreal–Format.
3. Berechnung einer Näherung $\boldsymbol{S} \approx (\boldsymbol{R} \odot \boldsymbol{A})^{-1}$ mit Hilfe einer LU–Zerlegung von $(\boldsymbol{R} \odot \boldsymbol{A})$.
4. Exakte Auswertung von $\boldsymbol{SR} \approx (\boldsymbol{R} \odot \boldsymbol{A})^{-1} \cdot \boldsymbol{R} \approx \boldsymbol{A}^{-1}$ im Akkumulator.
5. $\boldsymbol{R}_1 := \square(\boldsymbol{SR});\qquad \boldsymbol{R}_2 := \square(\boldsymbol{SR} - \boldsymbol{R}_1);$

Die Summe $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ ist dann nach dem Toolbox-Band I die doppeltlange Näherungsinverse. Der Operator $\cdot$ definiert wie üblich die exakte Matrix–Multiplikation, während $\odot$ die maximalgenaue Multiplikation im Longrealformat bedeutet, und die Schreibweise $\square(\boldsymbol{SR})$ symbolisiert das elementeweise Runden des exakten Produkts $\boldsymbol{SR}$ zur jeweils nächstgelegenen Longreal-Zahl.

Natürlich gilt neben (23) völlig gleichberechtigt auch die Gleichung

$$\boldsymbol{A}^{-1} = \boldsymbol{R}(\boldsymbol{A}\boldsymbol{R})^{-1} \tag{24}$$

Bezieht man sich dann auf (24), so läßt sich das Verfahren von Rump ganz analog auch in der folgenden Version 'rechts' formulieren. Wir benutzen die Bezeichnung 'rechts', da die zu invertierende Matrix $\boldsymbol{A}$ jetzt von rechts mit $\boldsymbol{R}$ multipliziert wird:

Verfahren von Rump, Version **rechts**:

1. Berechnung einer 0. Nährungsinversen $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer LU–Zerlegung.
2. Berechnung von $\boldsymbol{A} \odot \boldsymbol{R}$ im Longreal–Format.
3. Berechnung einer Näherung $\boldsymbol{S} \approx (\boldsymbol{A} \odot \boldsymbol{R})^{-1}$ mit Hilfe einer LU–Zerlegung von $(\boldsymbol{A} \odot \boldsymbol{R})$.
4. Exakte Auswertung von $\boldsymbol{R}\boldsymbol{S} \approx \boldsymbol{R} \cdot (\boldsymbol{A} \odot \boldsymbol{R})^{-1} \approx \boldsymbol{A}^{-1}$ im Akkumulator.
5. $\boldsymbol{R}_1 := \square(\boldsymbol{R}\boldsymbol{S}); \qquad \boldsymbol{R}_2 := \square(\boldsymbol{R}\boldsymbol{S} - \boldsymbol{R}_1);$

Die Summe $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ ist auch jetzt wie bei der Version 'links' die doppeltlange Näherungsinverse. Die beiden Algorithmen des Rump'schen Verfahrens werden realisiert durch Aufruf der **PASCAL-XSC** Prozedur

```
INV_RUMP( var A      : Lrmatrix; factor : integer;
          var R1,R2  : Lrmatrix; var Err : integer);

          exportiert durch linv
```

Die Eingangsgröße $\boldsymbol{A}$ ist die zu invertierende reguläre Matrix. Der Eingangsparameter factor bestimmt die Version des Rump'schen Verfahrens:

factor=0: Verfahren von Rump, Version **links**, vergl. Seite 33.

factor=1: Verfahren von Rump, Version **rechts**, vergl. Seite 34

Die Ausgangsgrößen R1,R2 liefern als Summe die berechnete Näherungsinverse, und Err enthält den Fehlercode, wobei Err=0 signalisiert, daß der Algorithmus fehlerfrei ausgewertet wurde.

Der von den Matrizen $\boldsymbol{R}_1, \boldsymbol{R}_2$ abhängige relative Höchstfehler $\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$ ist entsprechend Seite 21 jetzt definiert durch:

$$\varepsilon[i,j] := \left| \frac{\boldsymbol{A}^{-1}[i,j] - \boldsymbol{R}_1[i,j] - \boldsymbol{R}_2[i,j]}{\boldsymbol{A}^{-1}[i,j]} \right|, \qquad \max_{i,j}(\varepsilon[i,j]) < \varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2);$$

Als Anwendung betrachten wir wieder die Boothroyd/Dekker Matrizen $\boldsymbol{D}$, deren Inversen $\boldsymbol{D}^{-1}$ nach Seite 21 durch die Elemente von $\boldsymbol{D}$ bis aufs Vorzeichen schon direkt bekannt sind. Der relative Höchstfehler $\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$ kann daher sehr einfach bestimmt werden. Um den Einfluß der durch $\boldsymbol{R}_2$ gegebenen doppeltlangen Rechnung besser beurteilen zu können, wird neben $\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$ auch der relative Höchstfehler $\varepsilon(\boldsymbol{R}_1, \boldsymbol{0})$ berechnet, wobei also nachträglich $\boldsymbol{R}_2 \equiv \boldsymbol{0}$ gesetzt wird. Damit ist $\varepsilon(\boldsymbol{R}_1, \boldsymbol{0})$ der relative Höchstfehler, wenn das Verfahren von Rump nur in einfacher Länge durchgeführt wird.

Die relativen Höchstfehler sind in Abhängigkeit von n für die beiden möglichen Versionen 'links' und 'rechts' in nachfolgender Tabelle zusammengestellt:

Inversion einer Dekkermatrix $\boldsymbol{D}$

Verfahren v. Rump: INV_RUMP(D,factor,R1,R2,Err)

Näherungsinverse: $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{D}^{-1}$

Relative Höchstfehler $\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$, $\varepsilon(\boldsymbol{R}_1, \boldsymbol{0})$

n	Version links, factor=0		Version rechts, factor=1	
	$\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$	$\varepsilon(\boldsymbol{R}_1, \boldsymbol{0})$	$\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$	$\varepsilon(\boldsymbol{R}_1, \boldsymbol{0})$
2	0	0	0	0
4	$4.56 \cdot 10^{-36}$	0	$3.00 \cdot 10^{-38}$	0
8	$1.97 \cdot 10^{-26}$	0	$1.90 \cdot 10^{-30}$	0
10	$4.82 \cdot 10^{-21}$	$8.75 \cdot 10^{-21}$	$1.52 \cdot 10^{-22}$	0
12	$5.48 \cdot 10^{-21}$	$7.52 \cdot 10^{-21}$	$7.94 \cdot 10^{-21}$	$9.73 \cdot 10^{-21}$
14	$4.93 \cdot 10^{-18}$	$4.93 \cdot 10^{-18}$	$8.81 \cdot 10^{-21}$	$1.08 \cdot 10^{-20}$
16	$1.05 \cdot 10^{-12}$	$1.05 \cdot 10^{-12}$	$8.38 \cdot 10^{-21}$	$9.80 \cdot 10^{-21}$
18	$1.67 \cdot 10^{-8}$	$1.67 \cdot 10^{-8}$	$1.54 \cdot 10^{-20}$	$1.42 \cdot 10^{-20}$
20	$2.51 \cdot 10^{-2}$	$2.51 \cdot 10^{-2}$	$2.43 \cdot 10^{-17}$	$2.43 \cdot 10^{-17}$
25	$4.33 \cdot 10^{+1}$	$4.33 \cdot 10^{+1}$	$3.52 \cdot 10^{-13}$	$3.52 \cdot 10^{-13}$
27	$6.21 \cdot 10^{+2}$	$6.21 \cdot 10^{+2}$	$7.16 \cdot 10^{-11}$	$7.16 \cdot 10^{-11}$

Zunächst erkennt man, daß die relativen Höchstfehler sehr stark von der jeweils benutzten Version des Rump'schen Verfahrens abhängen. Für $n = 27$ unterscheiden sich seine Werte z.B. um immerhin 11 Größenordnungen!

Vergleicht man jedoch die relativen Höchstfehler $\varepsilon(\boldsymbol{R}_1, \boldsymbol{R}_2)$, $\varepsilon(\boldsymbol{R}_1, \mathbf{0})$ miteinander, so erkennt man aus obiger Tabelle, daß beim Rump'schen Verfahren die doppeltlange Auswertung den relativen Höchstfehler unabhängig von der benutzten Version 'links' oder 'rechts' insbesondere für höhere Dimensionen nicht verbessert! Dieses zunächst enttäuschende Ergebnis ist jedoch einleuchtend, denn beispielsweise wird das Produkt $\boldsymbol{SR}$ zwar im Akkumulator rundungsfehlerfrei ausgewertet, aber die beiden Faktoren $\boldsymbol{S}, \boldsymbol{R}$ wurden vorher im Longreal-Format berechnet, so daß die Elemente dieser Matrizen schon mit einem Höchstfehler von $5 \cdot 10^{-21}$ behaftet sind. Man kann daher nicht erwarten, daß nur das zweimalige Auslesen des exakten Akku-Wertes $\boldsymbol{SR}$ eine größere Genauigkeit, d.h. einen kleineren relativen Höchstfehler liefert. Der relative Höchstfehler wird also nicht dadurch verbessert, daß man beim Verfahren von Rump den Akkumulator in doppelter Länge ausliest!

Wir bleiben beim Beispiel der Boothroyd/Dekkermatrizen $\boldsymbol{D}$ und untersuchen jetzt, wie sich die doppeltlange Auswertung des Rump'schen Verfahrens auf die entsprechenden Defekte auswirkt. Analog zu (2) definieren wir dazu die beiden möglichen Defektmatrizen:

$$\boxed{{}^r\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{A}(\boldsymbol{R}_1 + \boldsymbol{R}_2); \qquad {}^l\boldsymbol{D} := \boldsymbol{I} - (\boldsymbol{R}_1 + \boldsymbol{R}_2)A} \tag{25}$$

Mit Hilfe des Akkumulators lassen sich ${}^r\boldsymbol{D}$, ${}^l\boldsymbol{D}$ wieder maximalgenau auswerten, so daß auch ihre Gesamtnormen $\|{}^r\boldsymbol{D}\|$, $\|{}^l\boldsymbol{D}\|$ maximalgenau berechnet werden können. Für die Dekker–Matrizen $\boldsymbol{D}$ sind in nachfolgender Tabelle die Werte $\|{}^r\boldsymbol{D}\|$, $\|{}^l\boldsymbol{D}\|$ für das einfache und doppeltlange Rump'sche Verfahren bzgl. der Versionen 'links'und 'rechts' zusammengestellt.

Nach dem Aufruf der Prozedur INV_RUMP(D,factor,R1,R2,Err) erhält man die Defektnormen $\|{}^r\boldsymbol{D}\|$, $\|{}^l\boldsymbol{D}\|$ für die Auswertung des Verfahrens von Rump in einfacher Rechnung, indem man vor ihrer Auswertung die entsprechenden Matrizen $\boldsymbol{R}_2 = \mathbf{0}$ setzt. Zu jeder Version des Rump'schen Verfahrens berechnen wir nur jeweils die passende Defektnorm, da z.B. zur Version 'links' die Defektmatrix ${}^r\boldsymbol{D}$ die größere Norm liefert. Vergleichen Sie dazu bitte den gleichlautenden Merksatz von Seite 31, der sich zunächst nur auf die einfache Auswertung des Rump'schen verfahrens bezieht aber auch bei doppeltlanger Auswertung seine Gültigkeit behält.

Inversion einer Dekkermatrix D				
Verfahren v. Rump: INV_RUMP(D,factor,R1,R2,Err)				
Näherungsinverse: $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{D}^{-1}$; $\boldsymbol{R}_1 \approx \boldsymbol{D}^{-1}$				
n	Version links, factor=0		Version rechts, factor=1	
	einfach	doppelt	einfach	doppelt
	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$
2	0	0	0	0
3	0	$2.80 \cdot 10^{-38}$	0	$1.80 \cdot 10^{-39}$
4	0	$1.83 \cdot 10^{-35}$	0	$1.23 \cdot 10^{-37}$
6	0	$8.74 \cdot 10^{-30}$	0	$3.02 \cdot 10^{-32}$
8	0	$1.58 \cdot 10^{-25}$	0	$1.52 \cdot 10^{-29}$
10	$6.92 \cdot 10^{-8}$	$4.82 \cdot 10^{-20}$	0	$1.52 \cdot 10^{-21}$
12	$2.68 \cdot 10^{-4}$	$6.57 \cdot 10^{-20}$	$2.06 \cdot 10^{-4}$	$9.52 \cdot 10^{-20}$
14	$3.02 \cdot 10^{-1}$	$1.22 \cdot 10^{-19}$	$4.25 \cdot 10^{-1}$	$1.24 \cdot 10^{-19}$
16	$3.05 \cdot 10^{+2}$	$6.35 \cdot 10^{-16}$	$1.01 \cdot 10^{+3}$	$7.06 \cdot 10^{-19}$
18	$1.05 \cdot 10^{+6}$	$2.19 \cdot 10^{-11}$	$1.77 \cdot 10^{+6}$	$1.58 \cdot 10^{-15}$
20	$3.33 \cdot 10^{+9}$	$5.36 \cdot 10^{-1}$	$1.48 \cdot 10^{+9}$	$2.74 \cdot 10^{-12}$
22	$3.81 \cdot 10^{+12}$	$4.38 \cdot 10^{+4}$	$6.79 \cdot 10^{+12}$	$5.49 \cdot 10^{-9}$
24	$2.19 \cdot 10^{+15}$	$2.74 \cdot 10^{+6}$	$1.38 \cdot 10^{+16}$	$6.62 \cdot 10^{-6}$
26	$4.47 \cdot 10^{+16}$	$4.89 \cdot 10^{+8}$	$4.04 \cdot 10^{+19}$	$1.59 \cdot 10^{-2}$
27	$2.11 \cdot 10^{+16}$	$3.92 \cdot 10^{+8}$	$7.09 \cdot 10^{+20}$	$8.26 \cdot 10^{-1}$

Zunächst fällt auf, daß besonders bei größeren Dimensionen der Dekkermatrizen die doppeltlange Auswertung des Rump'schen Verfahrens mit der Version 'rechts' im Vergleich zur Version 'links' deutlich kleinere Defektnormen liefert.

Das wichtigste Ergebnis ist jedoch, daß jetzt im Gegensatz zum relativen Höchstfehler die Defektnormen bei doppeltlanger Auswertung des Vefahrens von Rump im Vergleich zur nur einfachen Auswertung ganz wesentlich verbessert werden! Dies ist insbesondere bei der Version 'rechts' der Fall, wo z.B. mit $n = 27$ eine Verkleinerung der Norm um ganze 21 Größenordungen erreicht wird, während mit der Version 'links' eine entsprechende Verbesserung nur um 8 Größenordungen erfoglt. Wir formulieren deshalb den folgenden Merksatz:

Bei doppeltlanger Auswertung und richtiger Versionswahl liefert das Rump'sche Verfahren im Vergleich zur einfachen Auswertung eine erhebliche Verbesserung der entsprechenden Defektnorm.

Man kann also erwarten, daß eine Defektiteration nach einer doppeltlangen Auswertung des Rump'schen Verfahrens auch bei schlechtkonditionierten Matrizen Näherungsinverse mit sehr kleinen Defektnormen liefern wird. Mit diesen sehr guten Näherungsinversen sind wir dann in der Lage, garantierte Einschließungen der gesuchten Inversen auch in kritischen Fällen noch erfolgreich berechnen zu können.

Bei einfacher Auswertung des Verfahrens von Rump hatten wir schon mit dem Merksatz von Seite 31 das Ergebnis gewonnen, daß man zu einer vorgegebenen Version die kleinste Defektnorm erhält, wenn man zu ihrer Berechnung die zur Version passende Defektdefinition wählt. Mit Hilfe der Boothroyd/Dekker Matrizen $\boldsymbol{D}$ wollen wir mit den Defektnormen aus nachfolgender Tabelle zeigen, daß man auch bei doppeltlanger Auswertung des Rump'schen Verfahrens das gleiche Ergebnis erhält:

Inversion einer Dekkermatrix $\boldsymbol{D}$				
Verfahren v. Rump: INV_RUMP(D,factor,R1,R2,Err)				
Näherungsinverse: $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{D}^{-1}$				
n	Version links, factor=0		Version rechts, factor=1	
	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$	$\Vert{}^l\boldsymbol{D}\Vert$	$\Vert{}^r\boldsymbol{D}\Vert$
2	0	0	0	0
5	$2.00 \cdot 10^{-20}$	$2.52 \cdot 10^{-17}$	$7.56 \cdot 10^{-16}$	$1.20 \cdot 10^{-20}$
7	$2.81 \cdot 10^{-20}$	$1.69 \cdot 10^{-15}$	$6.22 \cdot 10^{-13}$	$2.81 \cdot 10^{-20}$
9	$9.53 \cdot 10^{-21}$	$1.16 \cdot 10^{-9}$	$9.41 \cdot 10^{-12}$	$8.11 \cdot 10^{-23}$
12	$6.57 \cdot 10^{-20}$	$1.42 \cdot 10^{-4}$	$4.20 \cdot 10^{-4}$	$9.52 \cdot 10^{-20}$
14	$1.22 \cdot 10^{-19}$	$1.43 \cdot 10^{+0}$	$1.04 \cdot 10^{+0}$	$1.24 \cdot 10^{-19}$
16	$6.35 \cdot 10^{-16}$	$2.85 \cdot 10^{+4}$	$8.10 \cdot 10^{+2}$	$7.06 \cdot 10^{-19}$
19	$1.81 \cdot 10^{-6}$	$8.32 \cdot 10^{+10}$	$2.53 \cdot 10^{+8}$	$5.05 \cdot 10^{-14}$
22	$4.38 \cdot 10^{+4}$	$2.85 \cdot 10^{+19}$	$6.31 \cdot 10^{+14}$	$5.49 \cdot 10^{-9}$
25	$1.92 \cdot 10^{+7}$	$2.61 \cdot 10^{+21}$	$7.29 \cdot 10^{+19}$	$5.25 \cdot 10^{-4}$
27	$3.92 \cdot 10^{+8}$	$1.24 \cdot 10^{+23}$	$7.42 \cdot 10^{+22}$	$8.26 \cdot 10^{-1}$

In der Tat bestätigen die Defektnormen der letzten Tabelle den folgenden Merksatz:

> *Für die nach dem Verfahren von Rump berechnete doppeltlange Näherung $\boldsymbol{R}_1 + \boldsymbol{R}_2$ erhält man die kleinste Defektnorm, wenn man zur gewählten Rump–Version die entsprechende Defektdefinition aus Gleichung (25) benutzt.*

Als weitere Anwendung betrachten wir die schon auf Seite 30 angegebenen Matrizen

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix}$$

$$\boldsymbol{B} := \begin{pmatrix} 1 & 1 & 1+10^{-19} & 1 \\ 1 & 1+10^{-19} & 1 & 10^{-40} \\ 1+10^{-19} & 1 & 1 & 10^{-40} \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

Mit dem Prozeduraufruf INV_RUMP(U,factor,R1,R2,Err) berechnen wir zunächst für beide Versionen 'links' (factor=0) und 'rechts' (factor=1) die Näherungen $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{U}^{-1}$. Die zugehörigen Defektnormen $\|{}^l\boldsymbol{D}\|$, $\|{}^r\boldsymbol{D}\|$ sind zum Vergleich für beide Rump–Versionen in folgender Tabelle zusammengestellt:

INV_RUMP(U,factor,R1,R2,Err); $\boldsymbol{U} = \boldsymbol{A}$ bzw. $\boldsymbol{U} = \boldsymbol{B}$						
Berechnung der Defektnormen $\|{}^l\boldsymbol{D}\|$, $\|{}^r\boldsymbol{D}\|$ nach Definition (25), Seite 36						
U	Nur LU-Zerlegung		Rump links; factor=0		Rump rechts; factor=1	
	$\|{}^l\boldsymbol{D}\|$	$\|{}^r\boldsymbol{D}\|$	$\|{}^l\boldsymbol{D}\|$	$\|{}^r\boldsymbol{D}\|$	$\|{}^l\boldsymbol{D}\|$	$\|{}^r\boldsymbol{D}\|$
$\boldsymbol{A}$	$8.0 \cdot 10^{-1}$	$4.0 \cdot 10^{-20}$	$1.6 \cdot 10^{-20}$	$5.4 \cdot 10^{+0}$	$4.9 \cdot 10^{-20}$	$5.7 \cdot 10^{-40}$
$\boldsymbol{B}$	$4.0 \cdot 10^{-2}$	$1.4 \cdot 10^{-2}$	$1.8 \cdot 10^{-20}$	$2.3 \cdot 10^{-1}$	$5.0 \cdot 10^{-2}$	$6.5 \cdot 10^{-21}$

Man erkennt sehr deutlich, daß sich z.B. bei der Rump–Version 'rechts' die Defektnorm bei $\boldsymbol{U} = \boldsymbol{B}$ um 19 Größenordnungen verschlechtert, wenn man an Stelle von $\|{}^r\boldsymbol{D}\|$ die nicht zur Version passende Norm $\|{}^l\boldsymbol{D}\|$ berechnet. Die Tabelle zeigt weiter, daß die Übertragung der bei der LU–Zerlegung günstigsten Defektdefinition ${}^r\boldsymbol{D}$ auf die entsprechende Rump–Version 'rechts' die kleinste Norm liefert, vorausgesetzt, daß man bei dieser Version auch die gleiche Defektdefinition ${}^r\boldsymbol{D}$ wählt.

Als weitere Anwendung betrachten wir eine quadratische Blockmatrix $\boldsymbol{B}$, die durch zwei ebenfalls quadratische Matrizen $\boldsymbol{U}, \boldsymbol{V}$ gleicher Dimension n so konstruiert wird, daß die obere Hälfte von $\boldsymbol{B}$ durch die Folge $\boldsymbol{U}, \boldsymbol{V}$ und die untere Hälfte durch $\boldsymbol{V}, \boldsymbol{U}$ definiert wird; damit besitzt dann $\boldsymbol{B}$ die doppelte Dimension $2n$. Bei allen Blockmatrizen wählen wir $\boldsymbol{U} \equiv \mathbf{1}$ als konstante Matrix und $\boldsymbol{V} = <v>$ als Diagonalmatrix, wobei die Diagonalelemente durch v definiert sind und alle restlichen Elemente verschwinden. Mit $\boldsymbol{B} = \boldsymbol{B}(n, v)$ stehen uns jetzt Blockmatrizen der Dimension $2n$ zur Verfügung, die im Gegensatz zu den Boothroyd/Dekker Matrizen auch bei kleiner Dimension mit Hilfe von v beliebig schlecht konditioniert werden können. Für $|v| \ll 1$ ist nämlich die Größenordnung der Konditionszahl $\varkappa$ ungefähr gegeben durch:

$$\varkappa := \|\boldsymbol{A}\| \cdot \|\boldsymbol{A}^{-1}\| \;\sim\; |v^{-1}|, \qquad \text{falls } |v| \ll 1, \tag{26}$$

Es gilt z.B.

$$\boldsymbol{B}(4, 10^{-30}) := \begin{pmatrix} 1 & 1 & 1 & 1 & 10^{-30} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 10^{-30} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 10^{-30} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 10^{-30} \\ 10^{-30} & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 10^{-30} & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 10^{-30} & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 10^{-30} & 1 & 1 & 1 & 1 \end{pmatrix}$$

wobei die maximalgenaue Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{B}^{-1}$ gegeben ist durch:

$$\boldsymbol{R} := \begin{pmatrix} \alpha & \alpha & \alpha & \alpha & \boldsymbol{\delta} & \beta & \beta & \beta \\ \alpha & \alpha & \alpha & \alpha & \beta & \boldsymbol{\delta} & \beta & \beta \\ \alpha & \alpha & \alpha & \alpha & \beta & \beta & \boldsymbol{\delta} & \beta \\ \alpha & \alpha & \alpha & \alpha & \beta & \beta & \beta & \boldsymbol{\delta} \\ \boldsymbol{\delta} & \beta & \beta & \beta & \alpha & \alpha & \alpha & \alpha \\ \beta & \boldsymbol{\delta} & \beta & \beta & \alpha & \alpha & \alpha & \alpha \\ \beta & \beta & \boldsymbol{\delta} & \beta & \alpha & \alpha & \alpha & \alpha \\ \beta & \beta & \beta & \boldsymbol{\delta} & \alpha & \alpha & \alpha & \alpha \end{pmatrix} \approx \boldsymbol{B}^{-1}, \quad \text{mit:} \quad \begin{array}{l} \alpha := \;\; 6.25 \cdot 10^{-2} \\ \beta := -2.5 \cdot 10^{+29} \\ \delta := +7.5 \cdot 10^{+29} \end{array}$$

Für die Konditionszahl $\varkappa$ erhält man den Wert:

$$\varkappa_{\boldsymbol{B}} := \|\boldsymbol{B}\| \cdot \|\boldsymbol{B}^{-1}\| \approx \|\boldsymbol{B}\| \cdot \|\boldsymbol{R}\| = 4.8 \cdot 10^{+31}$$

Im Gegensatz dazu findet man für die Dekker Matrix $\boldsymbol{D}$ der entsprechenden Dimension $n = 8$ den viel kleineren Wert: $\varkappa_{\boldsymbol{D}} = 1.72 \cdot 10^{+12}$.

Bezüglich der Blockmatrizen $\boldsymbol{B}(n, 10^{-30})$ sind in nachfolgender Tabelle für verschiedene Dimensionen n die Defektnormen $\|{}^l\boldsymbol{D}\|$, $\|{}^r\boldsymbol{D}\|$ angegeben, die sich auf die Approximation nur mit Hilfe der LU–Zerlegung und auf die Approximation nach Rump mit den Versionen 'links' und 'rechts' beziehen.

INV_RUMP(B,factor,R1,R2,Err); $\boldsymbol{B} = \boldsymbol{B}(n, 10^{-30})$						
$\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{B}^{-1}$						
Berechnung der Defektnormen $\|{}^l\boldsymbol{D}\|$, $\|{}^r\boldsymbol{D}\|$ nach Definition (25), Seite 36						
n	Nur LU-Zerlegung		Rump links; factor=0		Rump rechts; factor=1	
	$\|{}^l\boldsymbol{D}\|$	$\|{}^r\boldsymbol{D}\|$	$\|{}^l\boldsymbol{D}\|$	$\|{}^r\boldsymbol{D}\|$	$\|{}^l\boldsymbol{D}\|$	$\|{}^r\boldsymbol{D}\|$
2	$2.0 \cdot 10^{+0}$	$4.0 \cdot 10^{-30}$	$5.0 \cdot 10^{-61}$	$7.5 \cdot 10^{-61}$	$5.0 \cdot 10^{-31}$	$1.0 \cdot 10^{-60}$
3	$1.2 \cdot 10^{+40}$	$1.8 \cdot 10^{+10}$	Err $\neq 0$	Err $\neq 0$	$1.8 \cdot 10^{+19}$	$3.6 \cdot 10^{-11}$
4	$2.4 \cdot 10^{+10}$	$3.2 \cdot 10^{+10}$	$6.9 \cdot 10^{-2}$	$1.9 \cdot 10^{+10}$	$4.8 \cdot 10^{+19}$	$7.5 \cdot 10^{-11}$
5	$4.0 \cdot 10^{+10}$	$5.0 \cdot 10^{+10}$	$2.6 \cdot 10^{-1}$	$4.5 \cdot 10^{+10}$	$2.0 \cdot 10^{+20}$	$2.6 \cdot 10^{-10}$
6	$4.8 \cdot 10^{+10}$	$6.0 \cdot 10^{-20}$	$7.6 \cdot 10^{-1}$	$1.0 \cdot 10^{+20}$	$4.6 \cdot 10^{+10}$	$2.4 \cdot 10^{-11}$
7	$8.4 \cdot 10^{+10}$	$5.6 \cdot 10^{-20}$	$8.6 \cdot 10^{-1}$	$1.7 \cdot 10^{+20}$	$6.4 \cdot 10^{+10}$	$4.2 \cdot 10^{-11}$
8	$3.2 \cdot 10^{+10}$	$9.6 \cdot 10^{-20}$	$1.6 \cdot 10^{-10}$	$6.8 \cdot 10^{+10}$	$6.8 \cdot 10^{+10}$	$4.8 \cdot 10^{-11}$
9	$1.5 \cdot 10^{+11}$	$1.3 \cdot 10^{-19}$	$2.4 \cdot 10^{+0}$	$4.1 \cdot 10^{+20}$	$6.0 \cdot 10^{+10}$	$7.2 \cdot 10^{-11}$
10	$7.2 \cdot 10^{+40}$	$6.0 \cdot 10^{+10}$	Err $\neq 0$	Err $\neq 0$	$1.1 \cdot 10^{+11}$	$6.0 \cdot 10^{-11}$
11	$4.4 \cdot 10^{+40}$	$9.7 \cdot 10^{+10}$	Err $\neq 0$	Err $\neq 0$	$6.9 \cdot 10^{+27}$	$7.6 \cdot 10^{-3}$
12	$1.2 \cdot 10^{+11}$	$1.2 \cdot 10^{+11}$	$1.6 \cdot 10^{+3}$	$5.9 \cdot 10^{+22}$	$4.0 \cdot 10^{+28}$	$4.3 \cdot 10^{-2}$

Für die Dimensionen 6 bis 9 liefert die Tabelle zunächst ein überraschendes Ergebnis, das wir wieder in einem Merksatz formulieren:

Es gibt spezielle Matrizen, bei denen die nur mit der LU–Zerlegung berechnete Näherung eine viel kleinere Defektnorm liefert, als die mit dem Verfahren von Rump berechnete, doppeltlange Näherung.

Man kann daher vermuten, daß in diesen Fällen die nur mit der LU–Zerlegung berechneten Ausgangsnäherungen mit einer anschließenden Defektiteration auch die besseren Approximationen liefern. Es wird sich jedoch zeigen, daß diese Defektiterationen stets besser konvergiern, wenn die Ausgangsnäherungen vorher mit dem Verfahren von Rump berechnet wurden.

Zur Bestimmung der besten Rump–Version liefern die Defektnormen aus obiger Tabelle den nächsten Merksatz:

> *Unterscheiden sich die Defektnormen aus der LU–Zerlegung um wenigstens eine Größenordnung und liefert z.B. ${}^{r}\boldsymbol{D}$ die kleinere Defektnorm, so ist auch die Version 'rechts' die günstigste Rump–Version.*

Haben die Defektnormen jedoch gleich Größenordnungen (n=4,5,12), so findet man die beste Rump–Version nur dadurch, daß man für beide Versionen die jeweils passende Defektnorm berechnet und beide Werte miteinander vergleicht. Das beschriebene Verfahren zur Bestimmung der Ausgangsnäherung $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ mit Hilfe der günstigsten Rump–Version wird beim Aufruf der folgenden Prozedur realisiert:

```
INV_RUMP( var A,R1,R2   : Lrmatrix;
          var factor,Err : integer);

exportiert durch linv
```

Zu gegebener Matrix $\boldsymbol{A}$ wird mit dem Verfahren von Rump eine Näherung $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ so berechnet, daß die zugehörige Defektnorm minimal wird. Die Ausgangsgröße factor liefert die bei der Approximation benutzte Rump–Version:

factor	=	0	$\Longleftrightarrow$	Rump–Version 'links' nach Seite 33
factor	=	1	$\Longleftrightarrow$	Rump–Version 'rechts' nach Seite 34

Es ist sinnvoll, dem Anwender diese Ausgangsgröße zu Verfügung zu stellen, da z.B. eine nachfolgende Defektiteration nur dann erfolgreich ist, wenn sie die gleiche, durch factor angegebene Defektdefinition benutzt. Die Ausgangsgröße Err enthält den Fehlercode, wobei Err=0 signalisiert, daß keine Fehler aufgetreten sind.

Wählt man jetzt für $\boldsymbol{A}$ die Blockmatrizen $\boldsymbol{B} = \boldsymbol{B}(n, 10^{-30})$, so liefert obige Prozedur die in nachfolgender Tabelle zusammengestellten Ergebnisse:

INV_RUMP(B,R1,R2,factor,Err)					
Blockmatrizen $\boldsymbol{B} = \boldsymbol{B}(n, 10^{-30})$					
$\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{B}^{-1}$ mit minimaler Defektnorm					
n	Defektnorm	factor	n	Defektnorm	factor
2	$1.0 \cdot 10^{-60}$	1	8	$4.8 \cdot 10^{-11}$	1
3	$3.6 \cdot 10^{-11}$	1	9	$7.2 \cdot 10^{-11}$	1
4	$7.5 \cdot 10^{-11}$	1	10	$6.0 \cdot 10^{-11}$	1
5	$2.6 \cdot 10^{-10}$	1	11	$7.6 \cdot 10^{-3}$	1
6	$2.4 \cdot 10^{-11}$	1	12	$4.3 \cdot 10^{-2}$	1
7	$4.2 \cdot 10^{-11}$	1	13	$9.8 \cdot 10^{-2}$	1

Vergleicht man die Defektnormen aus obiger Tabelle mit den entsprechenden Werten aus der Tabelle von Seite 41, so erkennt man, daß der Prozeduraufruf INV_RUMP(B,R1,R2,factor,Err) mit dem Verfahren von Rump tatsächlich die Approximation $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{B}^{-1}$ mit minimaler Defektnorm liefert.

Im nächsten Abschnitt werden wir mit der *Defektiteration* ein Methode kennenlernen, mit der man im Vergleich zum Verfahren von Rump mit viel weniger Aufwand eine doppeltlange Näherungsinverse sehr effektiv berechnen kann. Während nämlich die Defektiteration nur zwei wesentliche Matrix–Multiplikationen erfordert, verlangt das Verfahren von Rump nach Seite 33 noch die zusätzliche Inversion zweier Matrizen, was die Laufzeit doch merklich verlängert. Es wird sich zeigen, daß erst die richtige Kombination beider Verfahren auch bei schlechtkonditionierten Matrizen zu sehr guten Ergebnissen führt. Dabei ist nach unseren bisherigen Ergebnissen darauf zu achten, daß man zunächst bei doppeltlanger Rechnung mit dem Verfahren von Rump die Approximation mit minimaler Defektnorm berechnet, um dann bei der folgenden Defektiteration die mit dem Parameter factor gelieferte passende Defektnorm zu benutzen.

6.4 Defektiteration

Mit Hilfe der Defektmethode lassen sich bei linearen Gleichungssystemen und auch bei der Matrixinversion unvermeidbare Rundungsfehler bei den Gleitkommarechnungen nachträglich korrigieren. Dies kann in der **PASCAL-XSC** Umgebung sehr effektiv realisiert werden, da die Defekte mit Hilfe des exakten Skalarproduktes mit nur einer einzigen Rundung auf der letzten Mantissenstelle berechnet werden können.

6.4.1 Iteration in einfacher Länge

Für die Inverse $\boldsymbol{A}^{-1}$ gilt die Definitionsgleichung: $\boldsymbol{I} - \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{0}$ und durch Einsetzen der Näherung $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ erhält man an Stelle der Null-Matrix $\boldsymbol{0}$ die Defektmatrix ${}^l\boldsymbol{D}^{(0)}$:

$$ {}^l\boldsymbol{D}^{(0)} := \boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{A} \tag{27} $$

Der obere linke Index ^{l} kennzeichnet, daß in (27) die Matrix $\boldsymbol{A}$ von links mit $\boldsymbol{R}^{(0)}$ multipliziert wird. Der Ansatz $\boldsymbol{A}^{-1} = \boldsymbol{R}^{(0)} + \boldsymbol{Z}$ liefert mit den obigen Beziehungen für die exakte Korrektur $\boldsymbol{Z}$ die Gleichung:

$$ {}^l\boldsymbol{D}^{(0)} = \boldsymbol{Z}\boldsymbol{A} \tag{28} $$

die durch Multiplikation von rechts mit $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ näherungsweise gelöst wird durch:

$$ \boldsymbol{Z} \approx {}^l\boldsymbol{D}^{(0)}\boldsymbol{R}^{(0)} \tag{29} $$

Damit ist die Summe

$$ \boldsymbol{R}^{(0)} + {}^l\boldsymbol{D}^{(0)}\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1} $$

eine Verbesserung der Ausgangsnäherung $\boldsymbol{R}^{(0)}$, falls $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ hinreichend gut erfüllt ist. Sonst wird nämlich die Matrix $\boldsymbol{Z}$ aus (28) nicht genau genug durch die Korrektur ${}^l\boldsymbol{D}^{(0)}\boldsymbol{R}^{(0)}$ approximiert!

Wiederholt man die beschriebene Korrektur, indem man mit der verbesserten Näherung $\boldsymbol{R}^{(1)}$ nach (27) den Defekt ${}^l\boldsymbol{D}^{(1)} := \boldsymbol{I} - \boldsymbol{R}^{(1)}\boldsymbol{A}$ neu berechnet und damit den Bezug zur Wirklichkeit wiederherstellt, so erhält man das folgende, schon lange bekannte Verfahren der Defektiteration, wobei

$$ \boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1} \tag{30} $$

eine hinreichend gute Ausgangsnäherung bedeutet, die man z.B. mit Hilfe einer LU-Zerlegung berechnen kann:

(31)

Defektiteration, Version **links**:

$$\begin{array}{ll} {}^{l}\boldsymbol{D}^{(0)} := \boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{A}; & \boldsymbol{R}^{(1)} := \boldsymbol{R}^{(0)} + {}^{l}\boldsymbol{D}^{(0)}\boldsymbol{R}^{(0)} \\ {}^{l}\boldsymbol{D}^{(1)} := \boldsymbol{I} - \boldsymbol{R}^{(1)}\boldsymbol{A}; & \boldsymbol{R}^{(2)} := \boldsymbol{R}^{(1)} + {}^{l}\boldsymbol{D}^{(1)}\boldsymbol{R}^{(1)} \\ {}^{l}\boldsymbol{D}^{(2)} := \boldsymbol{I} - \boldsymbol{R}^{(2)}\boldsymbol{A}; & \boldsymbol{R}^{(3)} := \boldsymbol{R}^{(2)} + {}^{l}\boldsymbol{D}^{(2)}\boldsymbol{R}^{(2)} \end{array}$$

Anmerkungen:

1. Das Verfahren wird erfolgreich abgebrochen, wenn innerhalb einer vorgegebenen Schrittzahl der Fall $\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)}$ eintritt.

2. Das Verfahren wird mit dem Fehlercode 4 abgebrochen, wenn für $\nu \geq 2$ der Zehnerexponent eines Elements der Defektmatrix ${}^{l}\boldsymbol{D}^{(\nu)}$ den Wert 50 übersteigt. In diesem Fall ist eine Konvergenz des Iterationsverfahrens i.a. nicht mehr zu erwarten.

3. Gilt $\boldsymbol{A}^{-1}[i,j] = 0$ und $\boldsymbol{R}^{(0)}[i,j] \neq 0$, so endet das Verfahren in einer Endlosschleife, da die Zahl Null auf einem Rechner durch die Näherungen $\boldsymbol{R}^{(\nu)}$ nicht wirklich erreicht werden kann. Setzt man jedoch $\boldsymbol{R}^{(\nu)}[i,j] = 0$, wenn sich die Zehnerexponenten der von Null verschiedenen Werte $\boldsymbol{R}^{(\nu-1)}[i,j], \boldsymbol{R}^{(\nu)}[i,j]$ für $\nu \geq 3$ um mehr als 4 verkleinern, so wird das Verfahren erfolgreich abgeschlossen.

4. In (31) wird, im Gegensatz zum Verfahren von Schulz in (18), bei jedem Iterationsschritt der Defekt mit Hilfe der zu invertierenden Matrix $\boldsymbol{A}$ neu berechnet. Man kann daher erwarten, daß alle Elemente der $\boldsymbol{R}^{(\nu)}$ mit wachsendem ν im Rahmen des benutzten Datenformats (Longreal oder real) gegen die Elemente von $\boldsymbol{A}^{-1}$ konvergieren, zumal die Defekte ${}^{l}\boldsymbol{D}^{(\nu)}$ mit dem Akkumulator von **PASCAL-XSC** maximalgenau berechnet werden.

5. Vergleicht man das Verfahren von Schulz in (18) mit dem obigen Verfahren (31), so erkennt man, daß in jedem Iterationsschritt jeweils zwei Matrix-Multiplikationen durchzuführen sind, so daß sich bei gleicher Schrittzahl die Laufzeiten nicht wesentlich unterscheiden werden.

Benutzt man an Stelle von (27) die völlig gleichwertige Defektdefinition

$$ {}^{r}\boldsymbol{D}^{(0)} := \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(0)}, \tag{32}$$

so führen die analogen Überlegungen von Seite 44 zu der zweiten Formulierung der Defektiteration:

Defektiteration, Version **rechts**:

$$\begin{aligned} {}^{r}\boldsymbol{D}^{(0)} &:= \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(0)}; & \boldsymbol{R}^{(1)} &:= \boldsymbol{R}^{(0)} + \boldsymbol{R}^{(0)} \cdot {}^{r}\boldsymbol{D}^{(0)} \\ {}^{r}\boldsymbol{D}^{(1)} &:= \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(1)}; & \boldsymbol{R}^{(2)} &:= \boldsymbol{R}^{(1)} + \boldsymbol{R}^{(1)} \cdot {}^{r}\boldsymbol{D}^{(1)} \\ {}^{r}\boldsymbol{D}^{(2)} &:= \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(2)}; & \boldsymbol{R}^{(3)} &:= \boldsymbol{R}^{(2)} + \boldsymbol{R}^{(2)} \cdot {}^{r}\boldsymbol{D}^{(2)} \end{aligned} \tag{33}$$

Die beiden möglichen Versionen (31),(33) der Defektiteration werden realisiert beim Aufruf der Prozedur

```
INV_DEF( var A : Lrmatrix; factor : integer;
         var R : Lrmatrix; var Err : integer);
```

exportiert durch linv

$\boldsymbol{A}$ ist die zu invertierende Matrix, und $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ ist die Eingangsgröße, die von der durch Defektiteration i.a. verbesserten Näherung überschrieben wird. Der Eingangsparameter factor bestimmt dabei die Version der Defektiteration:

$$\begin{aligned} \text{factor} &= 0 \iff \text{Version 'links' nach (31)} \\ \text{factor} &= 1 \iff \text{Version 'rechts' nach (33)} \end{aligned}$$

Als erstes Beispiel betrachten wir die gutkonditionierte Matrix $\boldsymbol{A}$, die nach dem Verfahren von Schulz nicht invertiert werden konnte, vergl. Seite 26:

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-41} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \boldsymbol{A}^{-1} := \begin{pmatrix} 1 & -1 & \frac{1}{3}(1-10^{-21}) & 10^{-41} \\ 0 & 1 & -\frac{1}{3} & \frac{1}{3}\cdot 10^{-21} \\ 0 & 0 & \frac{1}{3} & -\frac{1}{3} \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

und wählen als grobe Eingangsnäherung der Inversen $\boldsymbol{A}^{-1}$ wie auf Seite 26

$$\boldsymbol{R}^{(0)} := \begin{pmatrix} 1.01 & -1.02 & 0.333 & 10^{-41} \\ 0 & 1.01 & -0.333 & 0.333 \cdot 10^{-21} \\ 0 & 0 & 0.333 & -0.333 \\ 0 & 0 & 0 & 1 \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

Der Prozeduraufruf INV_DEF(A,factor,R,Err) liefert dann mit factor=0,1 für $\boldsymbol{A}^{-1}$ nach ebenfalls 6 Iterationsschritten im Gegensatz zum Verfahren von Schulz jetzt die **maximalgenaue** Näherungsinverse:

$$\boldsymbol{R}^{(6)} := \begin{pmatrix} 1 & -1 & \frac{1}{3}(1-10^{-21}) & 10^{-41} \\ 0 & 1 & -\frac{1}{3}(1-10^{-21}) & \frac{1}{3} \cdot 10^{-21}(1-10^{-21}) \\ 0 & 0 & \frac{1}{3}(1-10^{-21}) & -\frac{1}{3}(1-10^{-21}) \\ 0 & 0 & 0 & 1 \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

Wegen factor=0,1 erhält man also bei diesem Beispiel für beide Versionen der Defektiteration das gleiche maximalgenaue Ergebnis.

Wir kommen jetzt zu einem Beispiel, mit dem sich das auf Seite 45 angegebene Abbruchkriterium $\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)}$ nachträglich begründen läßt. Das Beispiel zeigt weiter, daß mit dem obigen Kriterium nicht mit jeder Version der Defektiteration Konvergenz erreicht wird. Die zu invertierende Matrix lautet:

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix}$$

besitzt die maximalgenaue Näherungsinverse

$$\boldsymbol{R}_A := \begin{pmatrix} -10^{20}+1 & 0 & 10^{20} & -1+10^{-20} \\ -10^{20}+1 & 10^{20} & 0 & -1+10^{-20} \\ 1-10^{-20} & 0 & 0 & -1+10^{-20} \\ 2 \cdot 10^{20}-2 & -10^{20} & -10^{20} & 3-2 \cdot 10^{-20} \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

Für den Prozeduraufruf INV_DEF(A,factor,R,Err) wählen wir die Eingangsnäherung

$$\boldsymbol{R}^{(0)} = \begin{pmatrix} -10^{20}+2 & 0 & 10^{20}-1.1 & -1+2 \cdot 10^{-20} \\ -10^{20}+1 & 10^{20} & -0.1 & -1+10^{-20} \\ 1-10^{-20} & 0 & 10^{-21} & -1+10^{-20} \\ 2 \cdot 10^{20}-3 & -10^{20} & -10^{20}+1.2 & 3-3 \cdot 10^{-20} \end{pmatrix} \neq \boldsymbol{R}_A \tag{34}$$

Mit der Version links (31) (factor=0) liefert die Defektiteration mit dem Fehlercode Err=4 nach maximal 15 Iterationsschritten das Ergebnis:

$$
{}^{l}\boldsymbol{R}^{(15)} = \begin{pmatrix} -10^{20}+1 & 0 & 10^{20} & -1+10^{-20} \\ -10^{20}+1 & 10^{20} & 10^{-41} - 1.3 \cdot 10^{-60} & -1+10^{-20} \\ 1-10^{-20} & 0 & 0 & -1+10^{-20} \\ 2 \cdot 10^{20} - 2 & -10^{20} & -10^{20} & 3 - 2 \cdot 10^{-20} \end{pmatrix} \neq \boldsymbol{R}_A
$$

Im Sinne des Abbruchkriteriums ${}^{l}\boldsymbol{R}^{(k)} = {}^{l}\boldsymbol{R}^{(k-1)}$ liegt damit keine Konvergenz vor, da die Elemente ${}^{l}\boldsymbol{R}^{(\nu)}[2,3]$ nicht verschwinden; sie pendeln vielmehr um einen von Null verschiedenen Wert mit minimalen Abweichungen auf den letzten Mantissenstellen. Im Gegensatz dazu werden jedoch die zugehörigen Defektnormen

$$
\|{}^{l}\boldsymbol{D}^{(0)}\| = 8 \cdot 10^{-1}, \quad \|{}^{l}\boldsymbol{D}^{(1)}\| = \ldots = \|{}^{l}\boldsymbol{D}^{(15)}\| = 8 \cdot 10^{-20}
$$

konstant. Eine Abbruchbedigung $\|{}^{l}\boldsymbol{D}^{(k)}\| = \|{}^{l}\boldsymbol{D}^{(k-1)}\|$ an Stelle des auf Seite 45 angegebenen Kriteriums ${}^{l}\boldsymbol{R}^{(k)} = {}^{l}\boldsymbol{R}^{(k-1)}$ würde daher bei diesem Beispiel Konvergenz vortäuschen und wäre damit wegen ${}^{l}\boldsymbol{R}^{(15)} \neq \boldsymbol{R}_A$ unbrauchbar! In sehr seltenen Ausnahmesituationen kann jedoch bei einer Defektiteration auf dem Rechner das Abbruchkriterium $\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)}$ erfüllt sein, ohne daß dabei $\boldsymbol{R}^{(k)}$ die gesuchte, **maximalgenaue** Näherungsinverse ist. In einem solchen Fall wäre dann allerdings wegen der Aussage

$$
\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)} \quad \Longrightarrow \quad \|\boldsymbol{D}^{(k)}\| = \|\boldsymbol{D}^{(k-1)}\|
$$

auch das Abbruchkriterium $\|\boldsymbol{D}^{(k)}\| = \|\boldsymbol{D}^{(k-1)}\|$ erfüllt, mit der gleichen Folge $\boldsymbol{R}^{(k)} \neq \boldsymbol{R}_A$. Da sich aus der Gleichung $\|\boldsymbol{D}^{(k)}\| = \|\boldsymbol{D}^{(k-1)}\|$ nicht immer die Beziehung $\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)}$ ergibt, formulieren wir den Merksatz:

> *Bei der Defektiteration darf das Abbruchkriterium nicht mit den Defektnormen sondern nur mit Hilfe der Folge der Approximationen $\{\boldsymbol{R}^{(\nu)}\}$ formuliert werden:* $\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)} \rightsquigarrow$ *Iterationsabbruch. In sehr seltenen Fällen gilt jedoch:* $\boldsymbol{R}^{(k)} \neq \boldsymbol{R}_A$, *wobei* $\boldsymbol{R}_A \approx \boldsymbol{A}^{-1}$ *die maximalgenaue Näherungsinverse bedeutet.*

Liegt in Ausnahmefällen die Situation $\boldsymbol{R}^{(k)} = \boldsymbol{R}^{(k-1)} \neq \boldsymbol{R}_A$ vor, so wird man versuchen, entweder mit Hilfe einer doppeltlangen Defektiteration oder durch *staggered arithmetic* eine genauere Näherungsinverse zu berechnen.

Da der Parameter factor=0 mit der Prozedur INV_DEF(A,factor,R,Err) wegen Err=4 keine Konvergenz liefert, wählen wir jetzt factor=1 (Version rechts, (33)) und erhalten mit der gleichen, in (34) definierten Eingangsnäherung $\boldsymbol{R}^{(0)}$ mit nur drei Iterationsschritten die maximalgenaue Näherungsinverse ${}^{r}\boldsymbol{R}^{(3)} = \boldsymbol{R}_A$. Die zugehörigen Defektnormen sind gegeben durch:

$$\|{}^{r}\boldsymbol{D}^{(0)}\| = 4 \cdot 10^{-20}, \quad \|{}^{r}\boldsymbol{D}^{(1)}\| = 4 \cdot 10^{-21}, \quad \|{}^{r}\boldsymbol{D}^{(2)}\| = 4 \cdot 10^{-40}, \quad \|{}^{r}\boldsymbol{D}^{(3)}\| = 4 \cdot 10^{-40};$$

Bitte beachten Sie, daß sich die zur gleichen Eingangsnäherung $\boldsymbol{R}^{(0)}$ in (34) gehörigen Defeknormen $\|{}^{l}\boldsymbol{D}^{(0)}\| = 8 \cdot 10^{-1}$ und $\|{}^{r}\boldsymbol{D}^{(0)}\| = 4 \cdot 10^{-20}$ um 19 Größenordnungen unterscheiden und vergleichen Sie dazu auch den Merksatz von Seite 13.

Das letzte Beispiel hat gezeigt, daß die Defektiteration nur mit der Version rechts (factor=1) Konvergenz liefert. Im Gegensatz dazu erhält man im nächsten Beispiel Konvergenz nur mit der Version links (factor=0). Die zu invertierende Matrix lautet jetzt:

$$\boldsymbol{B} := \begin{pmatrix} 1 & 1 & 1+10^{-19} & 1 \\ 1 & 1+10^{-19} & 1 & 10^{-40} \\ 1+10^{-19} & 1 & 1 & 10^{-40} \\ 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{besitzt die maximalgenaue Näherungsinverse}$$

$$\boldsymbol{R}_B := \begin{pmatrix} -5 \cdot 10^{18} + \frac{1}{4} & -5 \cdot 10^{18} + \frac{1}{4} & +5 \cdot 10^{18} + \frac{1}{4} & +5 \cdot 10^{18} - \frac{1}{4} \\ -5 \cdot 10^{18} + \frac{1}{4} & +5 \cdot 10^{18} + \frac{1}{4} & -5 \cdot 10^{18} + \frac{1}{4} & +5 \cdot 10^{18} - \frac{1}{4} \\ 10^{+19} & 0 & 0 & -10^{+19} \\ -\frac{1}{2} + \frac{1}{4} \cdot 10^{-19} & -\frac{1}{2} + \frac{1}{4} \cdot 10^{-19} & -\frac{1}{2} + \frac{1}{4} \cdot 10^{-19} & \frac{3}{2} - 2 \cdot 10^{-20} \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

Für den Prozeduraufruf INV_DEF(B,factor,R,Err) ist die Eingangsnäherung gegeben durch:

$$\boldsymbol{R}^{(0)} = \begin{pmatrix} -5 \cdot 10^{18} + 0.5 & -5 \cdot 10^{18} + 0.5 & 5 \cdot 10^{18} & 5 \cdot 10^{18} - 0.5 \\ -5 \cdot 10^{18} & 5 \cdot 10^{18} & -5 \cdot 10^{18} + 0.5 & 5 \cdot 10^{18} \\ 10^{19} & \frac{2}{3} \cdot 10^{-2} \cdot (1 + 5 \cdot 10^{-21}) & 0 & -10^{19} \\ -0.5 + 10^{-21} & -0.5 - 10^{-21} & -0.5 + 5 \cdot 10^{-20} & 1.5 \end{pmatrix}$$

Mit dem Eingangsparameter factor=0 wählen wir bei der Defektiteration die Version links, d.h. nach (2) die Defekdefinition ${}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}$ und erhalten nach nur 4 Iterationsschritten das maximalgenaue Ergebnis: ${}^{l}\boldsymbol{R}^{(4)} = \boldsymbol{R}_B$. Es gilt: $\frac{2}{3} \cdot 10^{-2} \cdot (1 + 5 \cdot 10^{-21}) = 6.66666666666666666667 \cdot 10^{-3}$.

Für die Approximationen ${}^{l}\boldsymbol{R}^{(\nu)}$ sind die zugehörigen Defektnormen in nachfolgender Tabelle zusammengestellt:

INV_DEF(B,0,R,Err); Err=0							
$\boldsymbol{R} = \boldsymbol{R}^{(0)}$; Defektnormen $\|{}^l\boldsymbol{D}^{(\nu)}\|$ zu den Approximationen ${}^l\boldsymbol{R}^{(\nu)}$							
ν	$\|{}^l\boldsymbol{D}^{(\nu)}\|$	ν	$\|{}^l\boldsymbol{D}^{(\nu)}\|$	ν	$\|{}^l\boldsymbol{D}^{(\nu)}\|$	ν	$\|{}^l\boldsymbol{D}^{(\nu)}\|$
0	$2.67 \cdot 10^{-2}$	1	$1.78 \cdot 10^{-4}$	2	$7.91 \cdot 10^{-9}$	3	$1.57 \cdot 10^{-17}$
4	$1.00 \cdot 10^{-19}$						

Wählt man jedoch factor=1, d.h. bei der Defektiteration die Version rechts, so erhält man mit der gleichen Eingangsnäherung $\boldsymbol{R}^{(0)}$ wegen Err=4 keine Konvergenz. Nach maximal 15 Schritten wird die Iteration mit dem folgenden Ergebnis abgebrochen:

$$
{}^r\boldsymbol{R}^{(15)} = \begin{pmatrix} -5\cdot 10^{18}+\frac{1}{4} & -5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}-\frac{1}{4} \\ -5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}+\frac{1}{4} & -5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}-\frac{1}{4} \\ 10^{+19} & 5.3\ldots\cdot 10^{-23} & -3.3\ldots\cdot 10^{-22} & -10^{+19} \\ -\frac{1}{2}+\frac{1}{4}\cdot 10^{-19} & -\frac{1}{2}+\frac{1}{4}\cdot 10^{-19} & -\frac{1}{2}+\frac{1}{4}\cdot 10^{-19} & \frac{3}{2}-2\cdot 10^{-20} \end{pmatrix} \neq \boldsymbol{R}_B
$$

Konvergenz kommt nicht zustande, da z.B. die ${}^r\boldsymbol{R}^{(\nu)}[3,2]$ nicht verschwinden, sondern um einen festen Wert pendeln, wobei sich für größere ν nur noch die beiden letzten Mantissenstellen ändern. Wie beim letzten Beispiel werden jedoch auch jetzt die entsprechenden Defektnormen konstant, wie man nachfolgender Tabelle entnehmen kann:

INV_DEF(B,1,R,Err); Err=4							
$\boldsymbol{R} = \boldsymbol{R}^{(0)}$; Defektnormen $\|{}^r\boldsymbol{D}^{(\nu)}\|$ zu den Approximationen ${}^r\boldsymbol{R}^{(\nu)}$							
ν	$\|{}^r\boldsymbol{D}^{(\nu)}\|$	ν	$\|{}^r\boldsymbol{D}^{(\nu)}\|$	ν	$\|{}^r\boldsymbol{D}^{(\nu)}\|$	ν	$\|{}^r\boldsymbol{D}^{(\nu)}\|$
0	$2.67 \cdot 10^{-2}$	4	$1.09 \cdot 10^{-10}$	8	$5.88 \cdot 10^{-19}$	12	$1.03 \cdot 10^{-19}$
1	$3.12 \cdot 10^{-4}$	5	$8.29 \cdot 10^{-13}$	9	$1.03 \cdot 10^{-19}$	13	$1.01 \cdot 10^{-19}$
2	$8.65 \cdot 10^{-7}$	6	$1.11 \cdot 10^{-14}$	10	$1.01 \cdot 10^{-19}$	14	$1.01 \cdot 10^{-19}$
3	$1.54 \cdot 10^{-8}$	7	$1.57 \cdot 10^{-16}$	11	$1.03 \cdot 10^{-19}$	15	$1.01 \cdot 10^{-19}$

Es fällt auf, daß im Gegensatz zum letzten Beispiel die obigen Tabellenwerte $\|{}^l\boldsymbol{D}^{(0)}\|$, $\|{}^r\boldsymbol{D}^{(0)}\|$ jetzt übereinstimmen. Allerdings sind diese Werte nicht wirklich gleich, da in den Tabellen aus Platzgründen die Defektnormen stets zur zweiten Nachkommastelle aufgerundet wurden. Tatsächlich gilt für die Defektnormen der Eingangsnäherung $\boldsymbol{R}^{(0)}$ die Ungleichung: $\|{}^l\boldsymbol{D}^{(0)}\| < \|{}^r\boldsymbol{D}^{(0)}\|$. Mit den Ergebnissen der beiden letzten Beispiele läßt sich folgende Vermutung aussprechen:

Vermutung:

> Die Defektiteration ist mit derjenigen Version 'links' oder 'rechts' besonders erfolgreich, die zur Eingangsnäherung $\boldsymbol{R}^{(0)}$ mit der entsprechenden Defektdefinition ${}^{l}\boldsymbol{D}$ oder ${}^{r}\boldsymbol{D}$ die kleinere Norm liefert.
>
> Gilt also wie im letzten Beispiel $\|{}^{l}\boldsymbol{D}^{(0)}\| < \|{}^{r}\boldsymbol{D}^{(0)}\|$, so wird die Defektiteration der Version 'links' besonders gut konvergieren. Dabei kann es sogar vorkommen, daß die andere Version 'rechts' nicht nur schlechter sondern wie im letzten Beispiel überhaupt nicht konvergiert.

In ganz seltenen Ausnahmefällen wird diese Vermutung jedoch widerlegt, wie mit folgender Matrix gezeigt werden kann:

$$\boldsymbol{C} := \begin{pmatrix} 10^{-10} & 10^{-20} & 1 & -10^{-20} \\ 1 & 3+10^{-20} & 3 & 4 \\ 1 & 3 & 2 & 4 \\ 10^{-10} & 0.1 & 1 & -10^{-20}\cdot(1+10^{-20}) \end{pmatrix}$$

Berechnet man jetzt mit INV_LU(C,R,Err) eine Näherungsinverse $\boldsymbol{R} = \boldsymbol{R}^{(0)}$, so erhält man für die zugehörigen Defektnormen die folgenden Ergebnisse: $\|{}^{l}\boldsymbol{D}^{(0)}\| = 2.4\cdot 10^{-10}$ und $\|{}^{r}\boldsymbol{D}^{(0)}\| = 1.2\cdot 10^{-17}$. Nach obiger Vermutung müßte nun die Version 'rechts' der Defektiteration Konvergenz liefern. Tatsächlich ist aber nur die Version 'links' erfolgreich, d.h. nur INV_DEF(C,0,R,Err) berechnet mit der Eingangsnäherung $\boldsymbol{R} = \boldsymbol{R}^{(0)}$ die maximalgenaue Ausgangsnäherung $\boldsymbol{R}_C \approx \boldsymbol{C}^{-1}$, mit der $\boldsymbol{R} = \boldsymbol{R}^{(0)}$ überschrieben wird.

Da in der Praxis obige Vermutung fast immer bestätigt wird, arbeitet die Prozedur

INV(var A,R : Lrmatrix; var factor,Err : integer); exportiert durch linv

wie folgt: Zunächst wird mit Hilfe einer LU–Zerlegung von A eine Näherungsinverse $\boldsymbol{R}^{(0)}$ berechnet und dazu die beiden Defektnormen $\|{}^{l}\boldsymbol{D}^{(0)}\|$, $\|{}^{r}\boldsymbol{D}^{(0)}\|$. Nach obiger Vermutung wählt man dann die entsprechende Version der Defektiteration. Führt dieses Verfahren nicht zum Ziel, versucht man Konvergenz mit Hilfe der anderen Version zu erreichen. Im Konvergenzfall (Err=0) liefert die Ausgangsgröße factor die dabei intern gewählte Versionsnummer:

factor=0: Defektiteration, Version **links** nach (31), Seite 45

factor=1: Defektiteration, Version **rechts** nach (33), Seite 46

Der Prozeduraufruf INV(U,R,factor,Err) liefert mit den ab Seite 47 behandelten Matrizen $\boldsymbol{U} = \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ die in folgender Tabelle zusammengestellten Ergebnisse:

INV(U,R,factor,Err), $\boldsymbol{R} \approx \boldsymbol{U}^{-1}$				
$\boldsymbol{U}$	$\boldsymbol{R}$	Err	factor $\longleftrightarrow$	Version
$\boldsymbol{A}$	$\boldsymbol{R}_A$	0	1	rechts
$\boldsymbol{B}$	$\boldsymbol{R}_B$	0	0	links
$\boldsymbol{C}$	$\boldsymbol{R}_C$	0	0	links

$\boldsymbol{R} = \boldsymbol{R}_C$ bedeutet, daß mit $\boldsymbol{R}$ die zur exakten Inversen $\boldsymbol{C}^{-1}$ maximalgenaue Näherung berechnet wurde. Danach liefert die Prozedur INV(U,R,factor,Err) für die drei Beispiele mit Hilfe der jeweils angegebenen Version nur maximalgenaue Ergebnisse.

Abschließend noch ein Hinweis, warum die Vermutung von Seite 51 nicht immer erfüllt wird. Berechnet man nämlich die Eingangsnäherung $\boldsymbol{R}^{(0)}$ nur mit Hilfe der LU–Zerlegung von $\boldsymbol{A}$, so kommt dabei eine rechts- oder linksseitige Multiplikation von $\boldsymbol{A}$ mit einer Näherung nicht vor, so daß dadurch auch eine Vorentscheidung für die geeignete Defektdefinition nicht erfolgt. Genau dies geschieht jedoch, wenn $\boldsymbol{R}^{(0)}$ bei doppeltlanger Rechnung mit dem Verfahren von Rump bestimmt wird, vergl. dazu die Seiten 33,34. Nach dem Aufruf der Prozedur INV_RUMP(var A,R1,R2,factor,Err) liegt dann mit der intern gewählten und durch factor beschriebenen Rump–Version (vergl. Seite 42) auch die bevorzugte Defektdefinition fest, mit der dann eine anschließende Defektiteration in doppelter Länge durchzuführen ist.

Bevor wir uns einer solchen doppeltlangen Defektiteration zuwenden, wollen wir uns im nächsten Absschnitt mit der Frage beschäftigen, unter welchen Voraussetzungen die bisher betrachteten Defektiterationen konvergieren und welche Konvergenzordnung gegebenenfalls vorliegt. Dabei werden wir uns besonders dafür interessieren müssen, wie eine mögliche Konvergenz von den beiden durch (2) gegebenen Defektdefinitionen abhängt und wie das Konvergenzverhalten durch die unvermeidbaren Rundungsfehler in den Gleitpunktrechnungen beeinflußt wird.

6.4.2 Konvergenzbetrachtungen

Bevor wir uns mit der Konvergenz der Defektiteration befassen, soll zunächst gezeigt werden, daß es noch einen ganz anderen Zugang zu den Iterationsvorschriften (31),(33) gibt. Ist $\boldsymbol{A}$ die zu invertierende, quadratische, reguläre Matrix, so besitzen die beiden Fixpunktgleichungen

$$\boldsymbol{R} = (2\boldsymbol{I} - \boldsymbol{RA})\boldsymbol{R} \tag{35}$$

$$\boldsymbol{R} = \boldsymbol{R}(2\boldsymbol{I} - \boldsymbol{AR}) \tag{36}$$

jeweils den einzigen von $\mathbf{0}$ verschiedenen Fixpunkt $\boldsymbol{R}^* = \boldsymbol{A}^{-1}$. Die zugehörigen Iterationsgleichungen lauten dann:

$$\boldsymbol{R}^{(k+1)} := (2\boldsymbol{I} - \boldsymbol{R}^{(k)}\boldsymbol{A})\boldsymbol{R}^{(k)} \tag{37}$$

$$\boldsymbol{R}^{(k+1)} := \boldsymbol{R}^{(k)}(2\boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(k)}), \qquad k = 0, 1, 2, \ldots \tag{38}$$

Mit den beiden nach (2) möglichen Defektdefinitionen ergeben sich daraus die schon in (31),(33) angegebenen Iterationsgleichungen:

$${}^{l}\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{R}^{(k)}\boldsymbol{A}; \qquad \boldsymbol{R}^{(k+1)} := \boldsymbol{R}^{(k)} + {}^{l}\boldsymbol{D}^{(k)} \cdot \boldsymbol{R}^{(k)} \tag{39}$$

$${}^{r}\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(k)}; \qquad \boldsymbol{R}^{(k+1)} := \boldsymbol{R}^{(k)} + \boldsymbol{R}^{(k)} \cdot {}^{r}\boldsymbol{D}^{(k)} \tag{40}$$

Nun zur Frage der Konvergenz. Zunächst fällt auf, daß die Gleichungen (37),(38) in $\boldsymbol{R}^{(k)}$ nichtlinear sind, so daß die im linearen Fall so einfachen Konvergenzsätze[2] [11, Seite 583] hier nicht zur Anwendung kommen. Da die Fixpunktgleichungen (35),(36) neben dem hier nicht interessierenden Fixpunkt $\mathbf{0}$ nur den einen Fixpunkt $\boldsymbol{R}^* = \boldsymbol{A}^{-1}$ besitzen, schreiben wir die in (2) gegebenen Defektdefinitionen in der Form:

$$\boldsymbol{R}^{(k)} := \boldsymbol{A}^{-1}(\boldsymbol{I} - {}^{r}\boldsymbol{D}^{(k)}) \quad \text{mit} \quad {}^{r}\boldsymbol{D}^{(k)} = \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(k)} \tag{41}$$

$$\boldsymbol{R}^{(k)} := (\boldsymbol{I} - {}^{l}\boldsymbol{D}^{(k)})\boldsymbol{A}^{-1} \quad \text{mit} \quad {}^{l}\boldsymbol{D}^{(k)} = \boldsymbol{I} - \boldsymbol{R}^{(k)}\boldsymbol{A} \tag{42}$$

Setzt man jetzt die rechten Seiten von $\boldsymbol{R}^{(k)}$ in einer der beiden Gleichungen (37) oder (38) ein, so erhält man:

$$\boldsymbol{R}^{(k+1)} = \boldsymbol{A}^{-1}(\boldsymbol{I} - \{{}^{r}\boldsymbol{D}^{(k)}\}^2), \qquad \boldsymbol{R}^{(k+1)} = (\boldsymbol{I} - \{{}^{l}\boldsymbol{D}^{(k)}\}^2)\boldsymbol{A}^{-1} \tag{43}$$

Bei jedem Iterationsschritt werden damit die Defektmatrizen quadriert, was **quadratische Konvergenz** bedeutet; dies führt zum nächsten Merksatz:

---

[2] $\boldsymbol{X}^{(k+1)} := \boldsymbol{TX}^{(k)} + \boldsymbol{C} \iff \boldsymbol{x}_j^{(k+1)} := \boldsymbol{T}\boldsymbol{x}_j^{(k)} + \boldsymbol{c}_j, \; j = 1, 2, \ldots n;$

Falls die Defektiteration konvergiert, konvergiert sie quadratisch.

Bleibt noch die Frage, unter welchen Voraussetzungen die Defektiteration tatsächlich Konvergenz liefert. Zur Klärung schreiben wir dazu die Ausgangsnäherung $\boldsymbol{R}^{(0)}$ in der Form:

$$\begin{array}{llll} (44) & \boldsymbol{R}^{(0)} := \boldsymbol{A}^{-1}(\boldsymbol{I} - {}^r\boldsymbol{D}^{(0)}) & \text{mit} & {}^r\boldsymbol{D}^{(0)} = \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(0)} \\ (45) & \boldsymbol{R}^{(0)} := (\boldsymbol{I} - {}^l\boldsymbol{D}^{(0)})\boldsymbol{A}^{-1} & \text{mit} & {}^l\boldsymbol{D}^{(0)} = \boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{A} \end{array}$$

und durch Einsetzen in (37),(38) erhält man mit $\sigma(k) := 2^k$ die Beziehungen:

$$(46) \qquad \boldsymbol{R}^{(k)} = \boldsymbol{A}^{-1}(\boldsymbol{I} - \{{}^r\boldsymbol{D}^{(0)}\}^{\sigma(k)}), \quad \boldsymbol{R}^{(k)} = (\boldsymbol{I} - \{{}^l\boldsymbol{D}^{(0)}\}^{\sigma(k)})\boldsymbol{A}^{-1}$$

Danach konvergieren die $\boldsymbol{R}^{(k)}$ genau dann gegen die gesuchte Inverse $\boldsymbol{A}^{-1}$, wenn die Potenzen der Ausgangs-Defekte ${}^r\boldsymbol{D}^{(0)}$ und ${}^l\boldsymbol{D}^{(0)}$ gegen die Nullmatrix konvergieren, und dies ist genau dann der Fall, wenn der Spektralradius ϱ dieser Defekte kleiner Eins ist [13, Seite 16]. Dabei versteht man unter dem Spektralradius einer Matrix das Maximum der Beträge seiner Eigenwerte. Wir zeigen zunächst:

$$(47) \qquad \varrho({}^l\boldsymbol{D}^{(0)}) = \varrho({}^r\boldsymbol{D}^{(0)})$$

Zum Beweis benutzen wir die Tatsache, daß für zwei quadratische Matrizen $\boldsymbol{A}, \boldsymbol{B}$ gleicher Dimension die Eigenwerte von $\boldsymbol{AB}$ und $\boldsymbol{BA}$ übereinstimmen, während die zugehörigen Eigenvektoren i.a. verschieden sind:

$$\begin{array}{ccc} (\boldsymbol{AB} - \mu_i\boldsymbol{I})\boldsymbol{x}_i = \boldsymbol{0}, & (\boldsymbol{BA} - \mu_i\boldsymbol{I})\boldsymbol{y}_i = \boldsymbol{0}, & i = 1, 2, \ldots \\ & \mu_i = 1 - \lambda_i \quad \rightsquigarrow & \\ [(\boldsymbol{I} - \boldsymbol{AB}) - \lambda_i\boldsymbol{I}]\boldsymbol{x}_i = \boldsymbol{0}, & [(\boldsymbol{I} - \boldsymbol{BA}) - \lambda_i\boldsymbol{I}]\boldsymbol{y}_i = \boldsymbol{0}, & i = 1, 2, \ldots \end{array}$$

Danach haben die Defekte ${}^l\boldsymbol{D}^{(0)}, {}^r\boldsymbol{D}^{(0)}$ gleiche Eigenwerte, woraus die Behauptung (47) direkt folgt. Wir fassen die Ergebnisse in einem weiteren Merksatz zusammen:

Mit einer beliebigen Näherung $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$ konvergieren die Folgen (39) und (40) der $\boldsymbol{R}^{(k)}$ bei rundungsfehlerfreier Rechnung genau dann (quadratisch) gegen die Inverse $\boldsymbol{A}^{-1}$, wenn der Spektralradius des Defekts $\boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(0)}$ oder $\boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{A}$ kleiner Eins ist.

Wegen (47) liegt also im Falle $\varrho({}^l\boldsymbol{D}^{(0)}) = \varrho({}^r\boldsymbol{D}^{(0)}) < 1$ bei exakter Rechnung Konvergenz stets bei **beiden** Defektiterationen (39),(40) vor. Dies ist überraschend, da wir auf Seite 47 mit der Matrix $\boldsymbol{A}$ ein Beispiel kennengelernt hatten, bei dem die Defektiteration bei Gleitkommarechnung mit der in (34) gegebenen Näherung $\boldsymbol{R}^{(0)}$ nur mit der Defektdefinition rechts konvergent war! In nachfolgender Tabelle sind für diese Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix}$$

mit der maximalgenauen Näherungsinversen

$$\boldsymbol{R}_A := \begin{pmatrix} -10^{20}+1 & 0 & 10^{20} & -1+10^{-20} \\ -10^{20}+1 & 10^{20} & 0 & -1+10^{-20} \\ 1-10^{-20} & 0 & 0 & -1+10^{-20} \\ 2\cdot 10^{20}-2 & -10^{20} & -10^{20} & 3-2\cdot 10^{-20} \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

und der in (34) gegebenen Eingangsnäherung

$$\boldsymbol{R}^{(0)} = \begin{pmatrix} -10^{20}+2 & 0 & 10^{20}-1.1 & -1+2\cdot 10^{-20} \\ -10^{20}+1 & 10^{20} & -0.1 & -1+10^{-20} \\ 1-10^{-20} & 0 & 10^{-21} & -1+10^{-20} \\ 2\cdot 10^{20}-3 & -10^{20} & -10^{20}+1.2 & 3-3\cdot 10^{-20} \end{pmatrix} \neq \boldsymbol{R}_A$$

die Defektnormen angegeben, wenn die Defektiteration bei exakter Rechnung mit Mathematica oder bei Gleitkommarechnung mit Hilfe der **PASCAL-XSC** Prozedur INV_DEF(A,factor,R,Err) durchgeführt wird. Die beiden in Gleichung (2) angegebenen Defektdefinitionen werden in INV_DEF durch die Parameterwerte factor=0,1 berücksichtigt, vergl. Seite 46.

Defektiteration nach (39),(40)				
mit der Eingangsnäherung $\boldsymbol{R}^{(0)} \approx \boldsymbol{A}^{-1}$				
${}^l\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{R}^{(k)}\boldsymbol{A}$, ${}^r\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}^{(k)}$, $k = 0, 1, 2, \ldots$				
Spektralradius $\varrho({}^l\boldsymbol{D}^{(0)}) = \varrho({}^r\boldsymbol{D}^{(0)}) = 1.091\ldots\cdot 10^{-20}$				
	Mathematica		INV_DEF(A,factor,R,Err)	
k	$\Vert{}^l\boldsymbol{D}^{(k)}\Vert$	$\Vert{}^r\boldsymbol{D}^{(k)}\Vert$	$\Vert{}^l\boldsymbol{D}^{(k)}\Vert$	$\Vert{}^r\boldsymbol{D}^{(k)}\Vert$
0	$8.00\cdot 10^{-1}$	$4.00\cdot 10^{-20}$	$8.00\cdot 10^{-1}$	$4.00\cdot 10^{-20}$
1	$4.00\cdot 10^{-21}$	$4.01\cdot 10^{-40}$	$8.00\cdot 10^{-20}$	$4.00\cdot 10^{-21}$
2	$5.20\cdot 10^{-61}$	$5.25\cdot 10^{-80}$	$8.00\cdot 10^{-20}$	$4.00\cdot 10^{-40}$
3	$7.39\cdot 10^{-141}$	$7.45\cdot 10^{-160}$	$8.00\cdot 10^{-20}$	$4.00\cdot 10^{-40}$
4	$1.49\cdot 10^{-300}$	$1.38\cdot 10^{-321}$	Err = 4	Err = 0

Betrachtet man zunächst die mit Mathematica exakt berechneten Werte $\|{}^l\boldsymbol{D}^{(k)}\|, \|{}^r\boldsymbol{D}^{(k)}\|$, so erkennt man recht deutlich die quadratische Konvergenz, wobei die Defektnormen $\|{}^r\boldsymbol{D}^{(k)}\|$ wegen des sehr viel kleineren Startwertes $\|{}^r\boldsymbol{D}^{(0)}\| = 4.00 \cdot 10^{-20}$ um mindestens 19 Größenordnungen kleiner ausfallen als die Defektnormen $\|{}^l\boldsymbol{D}^{(k)}\|$. Ganz anders ist jedoch die Situation bei den durch INV_DEF in Gleitkommarechnung bestimmten Defektnormen, die keine quadratische Konvergenz zeigen, obwohl die intern zu berechnenden Defekte in (39),(40) bei jedem Iterationsschritt mit Hilfe des langen Akkumulators jeweils **maximalgenau** berechnet werden! Für die starken Abweichungen von den exakten Mathematica-Rechnungen gibt es zwei wesentliche Gründe:

1. Auf dem Rechner kann die Inverse wegen des endlichen Datenformats nicht besser als durch $\boldsymbol{R}_A$ approximiert werden. Gilt nun wie in diesem Beispiel $\boldsymbol{R}_A \neq \boldsymbol{A}^{-1}$, so bleiben die beiden Defekte ${}^l\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}_A\boldsymbol{A}$ bzw. ${}^r\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}_A$ von der Nullmatrix verschieden, so daß sich die entsprechenden Normen nicht beliebig verkleinern können und damit quadratische Konvergenz verhindern.

2. Die Durchführung der Defektiteration nach den Vorschriften (39),(40) ist in Gleitkomma-Arithmetik mit unvermeidbaren Rundungsfehlern behaftet, die wie in diesem Beispiel mit der Defektdefinition ${}^l\boldsymbol{D}^{(k)}$ eine Konvergenz auf der Maschine sogar ganz unterbinden (Err=4).

Im nächsten Beispiel kommen wir nochmals auf die Matrix $\boldsymbol{B}$ von Seite 49 zurück, mit der sich der unter Punkt 2. genannte Einfluß der Rundungsfehler bei der Defektiteration besonders gut zeigen läßt. In nachfolgender Tabelle sind für diese Matrix

$$\boldsymbol{B} := \begin{pmatrix} 1 & 1 & 1+10^{-19} & 1 \\ 1 & 1+10^{-19} & 1 & 10^{-40} \\ 1+10^{-19} & 1 & 1 & 10^{-40} \\ 1 & 1 & 1 & 1 \end{pmatrix} \quad \text{mit der maximalgenauen Näherungsinversen}$$

$$\boldsymbol{R}_B := \begin{pmatrix} -5\cdot 10^{18}+\frac{1}{4} & -5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}-\frac{1}{4} \\ -5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}+\frac{1}{4} & -5\cdot 10^{18}+\frac{1}{4} & +5\cdot 10^{18}-\frac{1}{4} \\ 10^{+19} & 0 & 0 & -10^{+19} \\ -\frac{1}{2}+\frac{1}{4}\cdot 10^{-19} & -\frac{1}{2}+\frac{1}{4}\cdot 10^{-19} & -\frac{1}{2}+\frac{1}{4}\cdot 10^{-19} & \frac{3}{2}-2\cdot 10^{-20} \end{pmatrix} \approx \boldsymbol{A}^{-1}$$

und der Eingangsnäherung

$$
\boldsymbol{R}^{(0)} = \begin{pmatrix} -5\cdot 10^{18}+0.5 & -5\cdot 10^{18}+0.5 & 5\cdot 10^{18} & 5\cdot 10^{18}-0.5 \\ -5\cdot 10^{18} & 5\cdot 10^{18} & -5\cdot 10^{18}+0.5 & 5\cdot 10^{18} \\ 10^{19} & \frac{2}{3}\cdot 10^{-2}\cdot(1+5\cdot 10^{-21}) & 0 & -10^{19} \\ -0.5+10^{-21} & -0.5-10^{-21} & -0.5+5\cdot 10^{-20} & 1.5 \end{pmatrix}
$$

die Defektnormen angegeben, wenn die Defektiteration bei exakter Rechnung mit Mathematica oder bei Gleitkommarechnung mit Hilfe der **PASCAL-XSC** Prozedur INV_DEF(B,factor,R,Err) durchgeführt wird. Die beiden in Gleichung (2) angegebenen Defektdefinitionen werden in INV_DEF durch die Parameterwerte factor=0,1 berücksichtigt, vergl. Seite 46.

Defektiteration nach (39),(40)				
mit der Eingangsnäherung $\boldsymbol{R}^{(0)} \approx \boldsymbol{B}^{-1}$				
${}^{l}\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{R}^{(k)}\boldsymbol{B}, \quad {}^{r}\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{B}\boldsymbol{R}^{(k)}, \quad k = 0,1,2,\ldots$				
Spektralradius $\varrho({}^{l}\boldsymbol{D}^{(0)}) = \varrho({}^{r}\boldsymbol{D}^{(0)}) = 6.6666\ldots\cdot 10^{-3}$				
	Mathematica		INV_DEF(B,factor,R,Err)	
k	$\Vert{}^{l}\boldsymbol{D}^{(k)}\Vert$	$\Vert{}^{r}\boldsymbol{D}^{(k)}\Vert$	$\Vert{}^{l}\boldsymbol{D}^{(k)}\Vert$	$\Vert{}^{r}\boldsymbol{D}^{(k)}\Vert$
0	$2.67\cdot 10^{-2}$	$2.67\cdot 10^{-2}$	$2.67\cdot 10^{-2}$	$2.67\cdot 10^{-2}$
1	$1.78\cdot 10^{-4}$	$1.78\cdot 10^{-4}$	$1.78\cdot 10^{-4}$	$3.12\cdot 10^{-4}$
2	$7.91\cdot 10^{-9}$	$7.91\cdot 10^{-9}$	$7.91\cdot 10^{-9}$	$8.65\cdot 10^{-7}$
3	$1.57\cdot 10^{-17}$	$1.57\cdot 10^{-17}$	$1.57\cdot 10^{-17}$	$1.54\cdot 10^{-8}$
4	$6.09\cdot 10^{-35}$	$6.09\cdot 10^{-35}$	$1.00\cdot 10^{-19}$	$1.09\cdot 10^{-10}$
5	$9.28\cdot 10^{-70}$	$9.28\cdot 10^{-70}$	Err = 0	Err = 4

Es gilt $\Vert{}^{l}\boldsymbol{D}^{(0)}\Vert < \Vert{}^{r}\boldsymbol{D}^{(0)}\Vert$, aber die Defektnormen unterscheiden sich erst auf der 17. Dezimalstelle, so daß sie als praktisch gleich angesehen werden können. Daher stimmen auch die mit Mathematica exakt berechneten Defektnormen in den ersten fünf Iterationsschritten überein und zeigen die geforderte quadratische Konvergenz. Im Vergleich zum vorhergehenden Beispiel erfolgt jetzt aber die Konvergenz wegen der Ungleichung $\varrho(\boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{B}) = 6.666\ldots\cdot 10^{-3} \gg \varrho(\boldsymbol{I} - \boldsymbol{R}^{(0)}\boldsymbol{A}) = 1.091\ldots\cdot 10^{-20}$ deutlich langsamer.

Betrachten wir jetzt die mit INV_DEF(B,factor,R,Err) berechneten Defektnormen $\Vert{}^{l}\boldsymbol{D}^{(k)}\Vert$, so erkennt man anfangs zwar quadratische Konvergenz, die jedoch im 4. Iterationsschritt wegen Konvergenz auf der Maschine gegen die maximalgenaue Näherung $\boldsymbol{R}_B \neq \boldsymbol{B}^{-1}$ abbricht. Dagegen wirken sich bei der Defektiteration mit dem Defekt ${}^r\boldsymbol{D}^{(k)} := \boldsymbol{I} - \boldsymbol{B}\boldsymbol{R}^{(k)}$ die Rundungsfehler so stark aus, daß die $\|{}^r\boldsymbol{D}^{(k)}\|$ weder anfangs quadratisch konvergieren noch daß ein Grenzwert der $\boldsymbol{R}^{(k)}$ auf der Maschine gefunden wird (Err=4).

Wir kommen jetzt auf den Merksatz von Seite 54 zurück. Für die praktische Anwendung ist dieser Satz natürlich unbrauchbar, da die Berechnung des Spektralradius viel zu aufwendig ist. In der Praxis wird man daher die Iteration (39) oder (40) mit einer Ausgangsnäherung $\boldsymbol{R}^{(0)}$ einfach beginnen und dabei hoffen, daß auf der Maschine Konvergenz eintritt. Da wir jedoch schon wissen, daß es Matrizen gibt, bei denen die Defektiteration für ein spezielles $\boldsymbol{R}^{(0)}$ nur mit **einer** Defektdefinition erfolgreich ist, wäre eine solche Probiermethode immer noch zu aufwendig! Gesucht ist also ein Verfahren, mit dem man relativ schnell und sicher entscheiden kann, welche Defektdefinition erfolgreich ist oder die bessere Konvergenz liefert. In der Praxis zeigt sich nun, daß man z.B. mit der Defektdefinition ${}^r\boldsymbol{D}^{(0)}$ in fast allen Fällen eine bessere Konvergenz erhält als mit der Definition ${}^l\boldsymbol{D}^{(0)}$, wenn folgende Ungleichung erfüllt ist:

$$(48) \qquad \|{}^r\boldsymbol{D}^{(0)}\| < \|{}^l\boldsymbol{D}^{(0)}\| \quad \rightsquigarrow \quad {}^r\boldsymbol{D}^{(0)} \text{ liefert bessere Konvergenz}$$

Natürlich ist diese Vorgehensweise kein echtes Kriterium für eine bessere Konvergenz der Defektiteration (40) mit der Definition ${}^r\boldsymbol{D}^{(0)}$. Die Überprüfung obiger Ungleichung ist jedoch nicht besonders aufwendig, da lediglich ein zweiter Defekt zu berechnen ist. Die Auswahl der günstigsten Defektdefinition erfolgt dann mit einer ziemlich großen Sicherheit, so daß mit (48) ein für die Praxis sehr nützliches Auswahlverfahren gegeben ist. Falls nun in seltenen Ausnahmefällen obige Ungleichung erfüllt ist und der Defekt ${}^r\boldsymbol{D}^{(0)}$ auf der Maschine dennoch keine Konvergenz liefert, so muß man versuchen, die gewünschte Konvergenz mit der anderen Defektdefinition ${}^l\boldsymbol{D}^{(0)}$ zu erreichen. Genau diese Überlegungen werden realisiert beim Aufruf der Prozedur INV(A,R,factor,Err) von Seite 51. Wenn nun diese Prozedur wegen Err $\neq 0$ keine Konvergenz liefert, so muß die Defektiteration in doppelter Länge des benutzten Datenformats ausgeführt werden.

Im nächsten Abschnitt werden wir eine solche doppeltlange Defektiteration kennenlernen, wobei je nach Wahl der Defektdefinition wieder zwei Versionen zu unterscheiden sind. Es wird sich zeigen, daß für die Berechnung einer möglichst maximalgenauen Näherungsinversen die richtige Wahl dieser Version gerade bei schlechtkonditionierten Matrizen für das Konvergenzverhalten von entscheidender Bedeutung ist.

6.4.3 Doppeltlange Defektiteration

In diesem Abschnitt beschäftigen wir uns zunächst mit der Aufgabe, zu einer regulären Matrix $\boldsymbol{A} \in I\!R^{n\times n}$, für die eine Näherungsinverse schon bekannt ist, mit Hilfe der Defektiteration eine möglichst maximalgenaue Näherungsinverse in **doppelter** Länge des benutzten Datenformats zu berechnen, d.h. in der Approximation

$$\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1} \tag{49}$$

sind die beiden Matrizen $\boldsymbol{R}_1$, $\boldsymbol{R}_2$ gesucht, wobei die Haupt-Näherung durch $\boldsymbol{R}_1$ gegeben ist und $\boldsymbol{R}_2$ die gewünschte, doppelte Genauigkeit liefert. Zur Realisierung betrachten wir dazu den Algorithmus (31) mit dem Ziel, die in (28) angegebene Gleichung für die exakte Korrektur $\boldsymbol{Z}$ genauer zu lösen.

Für die zu approximierende Inverse $\boldsymbol{A}^{-1}$ gilt zunächst wieder ganz analog die Definitionsgleichung:

$$\boldsymbol{I} - \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{0} \tag{50}$$

Gegeben sei eine grobe Näherungsinverse $\boldsymbol{R}^{(0)}$ in doppelter Länge:

$$\boldsymbol{R}^{(0)} := \boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)} \approx \boldsymbol{A}^{-1}$$

Dann ist der zugehörige Defekt nach (50) definiert durch

$$^{l}\boldsymbol{D}^{(0)} := \boldsymbol{I} - (\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)}) \cdot \boldsymbol{A} \neq \boldsymbol{0} \tag{51}$$

Der Index ^{l} kennzeichnet, daß die Matrix $\boldsymbol{A}$ von links mit $(\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)})$ multipliziert wird. Der Ansatz $\boldsymbol{A}^{-1} = \boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)} + \boldsymbol{Z}$ liefert für die exakte Korrektur $\boldsymbol{Z}$ wie auf Seite 44 die Gleichung

$$^{l}\boldsymbol{D}^{(0)} = \boldsymbol{Z}\boldsymbol{A}, \tag{52}$$

die wir jetzt genauer lösen können, wenn von rechts mit der doppeltlangen Näherung $\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)} \approx \boldsymbol{A}^{-1}$ multipliziert wird. Für $\boldsymbol{Z}$ ergibt sich damit die Approximation $\boldsymbol{Z} \approx {}^{l}\boldsymbol{D}^{(0)}(\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)})$ und die Summe

$$^{l}\boldsymbol{R}^{(1)} := \boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)} + {}^{l}\boldsymbol{D}^{(0)}(\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)}) \tag{53}$$

ist dann eine Verbesserung der vorgegebenen nicht zu groben Ausgangsnäherung $\boldsymbol{R}^{(0)} := \boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)}$;

Anmerkungen zur Rechnerauswertung:

1. Auf dem Rechner wird die Defektmatrix ${}^l\boldsymbol{D}^{(0)}$ maximalgenau ausgewertet, indem man für jedes Element die rechte Seite von (51) rundungsfehlerfrei im Akkumulator berechnet und seinen Wert dann zur jeweils nächsten Rasterzahl des Longreal–Formats rundet:
   ${}^l\boldsymbol{D}^{(0)} := \square(\boldsymbol{I} - (\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)})\,\boldsymbol{A})$.
2. Die rechte Seite von (53) wird für jedes Matrixelement im Akkumulator (Akku) rundungsfehlerfrei berechnet, der dann zweimal zur jeweils nächsten Rasterzahl ausgelesen wird:
   ${}^l\boldsymbol{R}_1^{(1)} := \square(\text{Akku});\qquad {}^l\boldsymbol{R}_2^{(1)} := \square(\text{Akku} - {}^l\boldsymbol{R}_1^{(1)})$.
   Mit ${}^l\boldsymbol{R}^{(1)} := {}^l\boldsymbol{R}_1^{(1)} + {}^l\boldsymbol{R}_2^{(1)} \approx \boldsymbol{A}^{-1}$ haben wir damit eine verbesserte, doppeltlange Näherungsinverse berechnet.

Mit der Ausgangsnäherung $\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)} \approx \boldsymbol{A}^{-1}$ lautet damit das Iterationsverfahren zur Berechnung einer doppeltlangen Näherungsinversen:

(54) Defektiteration, Version **links**:

$$\begin{array}{rcl} {}^l\boldsymbol{D}^{(\nu)} & := & \square(\boldsymbol{I} - ({}^l\boldsymbol{R}_1^{(\nu)} + {}^l\boldsymbol{R}_2^{(\nu)})\boldsymbol{A}); \quad \nu = 0, 1, \ldots \\ Akku & := & {}^l\boldsymbol{R}_1^{(\nu)} + {}^l\boldsymbol{R}_2^{(\nu)} + {}^l\boldsymbol{D}^{(\nu)}({}^l\boldsymbol{R}_1^{(\nu)} + {}^l\boldsymbol{R}_2^{(\nu)}); \\ {}^l\boldsymbol{R}_1^{(\nu+1)} & := & \square(Akku);\ {}^l\boldsymbol{R}_2^{(\nu+1)} := \square(Akku - {}^l\boldsymbol{R}_1^{(\nu+1)}) \end{array}$$

Anmerkungen zum Algorithmus:

Wie bei allen Iterationsverfahren ist auch jetzt die Abbruchbedingung sehr sorgfältig zu wählen, wenn man korrekte Ergebnisse in vernünftigen Laufzeiten erhalten will. Zunächst ist klar, daß man die Defektiteration mit dem Fehlercode Err=0 beenden wird, wenn ${}^l\boldsymbol{D}^{(\nu)}$ verschwindet, da man dann möglicherweise sogar die exakte Inverse gefunden hat. Außerdem erfolgt ein Abbruch mit dem gleichen Fehlercode, wenn in drei aufeinanderfolgenden Iterationsschritten die Bedingung

$${}^l\boldsymbol{R}_1^{(\nu)} = {}^l\boldsymbol{R}_1^{(\nu-1)}$$

erfüllt ist. Wegen der quadratischen Konvergenz der Defektiteration kann man dann erwarten, daß danach auch die zweiten Komponenten übereinstimmen:

$${}^l\boldsymbol{R}_2^{(\nu)} = {}^l\boldsymbol{R}_2^{(\nu-1)}$$

Andererseits wird man die Iteration mit dem Fehlercode Err $\neq 0$ abbrechen, wenn die erste Bedigung innerhalb einer maximalen Schrittzahl nicht erreicht wird. Bei Durchführung der Defektiteration sind im Fall ${}^{l}\boldsymbol{D}^{(\nu)} \neq 0$ folgende Punkte zu beachten:

1. Gilt $\boldsymbol{A}^{-1}[i,j] = 0$ und ${}^{l}\boldsymbol{R}_1^{(\nu)}[i,j] \neq 0$, so kann das Verfahren in einer Endlosschleife münden, da die Zahl Null auf einem Rechner durch die Näherungen ${}^{l}\boldsymbol{R}_1^{(\nu)}$ im allgemeinen nicht erreicht wird. Setzt man daher ${}^{l}\boldsymbol{R}_1^{(\nu)}[i,j] = {}^{l}\boldsymbol{R}_2^{(\nu)}[i,j] = 0$, wenn sich die Zehnerexponenten der von Null verschiedenen Werte ${}^{l}\boldsymbol{R}_1^{(\nu-1)}[i,j], {}^{l}\boldsymbol{R}_1^{(\nu)}[i,j]$ um mehr als 8 verkleinern, so wird das Verfahren erfolgreich abgeschlossen.

2. Ist $\boldsymbol{A}^{-1}[i,j] \neq 0$ und verkleinern sich die Zehnerexponenten der von Null verschiedenen Werte ${}^{l}\boldsymbol{R}_2^{(\nu-1)}[i,j], {}^{l}\boldsymbol{R}_2^{(\nu)}[i,j]$ auch noch bei größerer Schrittzahl um mehr als 8, so signalisiert dies, daß $\boldsymbol{A}^{-1}[i,j]$ schon durch ${}^{l}\boldsymbol{R}_1[i,j]^{(\nu)}$ alleine exakt dargestellt werden kann, so daß ${}^{l}\boldsymbol{R}_2^{(\nu)}[i,j] = 0$ gesetzt werden sollte, wenn man eine möglichst maximalgenaue Näherung der Inversen berechnen will.

Die oben beschriebenen Nullsetzungen werden durch Aufruf der Prozedur CheckZeros$({}^{l}\boldsymbol{R}_1^{(\nu-1)}, {}^{l}\boldsymbol{R}_2^{(\nu-1)}, {}^{l}\boldsymbol{R}_1^{(\nu)}, {}^{l}\boldsymbol{R}_2^{(\nu)}, s)$ realisiert, wobei der Parameter s die Differenz der Zehnerexponenten bedeutet. Es zeigt sich, daß insbesondere bei den schlechtkonditionierten Dekker Matrizen eine zu starke Manipulation der Näherung ${}^{l}\boldsymbol{R}_1^{(\nu)} + {}^{l}\boldsymbol{R}_2^{(\nu)} \approx \boldsymbol{A}^{-1}$ durch CheckZeros(...) zu einem explosionsartigen Anwachsen des Defekts ${}^{l}\boldsymbol{D}^{(\nu)}$ führt. Der Aufruf dieser Prozedur erfolgt daher erst nach vier Iterationsschritten oder wenn die Iteration schon vorher erfolgreich beendet wurde. Diese Vorgehensweise liefert auch bei schlechtkonditionierten Matrizen $\boldsymbol{A}$ i.a. eine maximalgenaue oder sogar auch exakte Näherungsinverse.

Bei der Formulierung der Defektiteration kann man an Stelle von (50) auch völlig gleichwertig von der Definitionsgleichung

$$\boldsymbol{I} - \boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{0} \tag{55}$$

ausgehen. Die ganz analogen Überlegungen von Seite 59 führen dann zur zweiten Version der doppeltlangen Defektiteration:

Defektiteration, Version **rechts**:

$$
(56) \qquad
\begin{array}{rcl}
{}^r\boldsymbol{D}^{(\nu)} & := & \square(\boldsymbol{I} - \boldsymbol{A}({}^r\boldsymbol{R}_1^{(\nu)} + {}^r\boldsymbol{R}_2^{(\nu)})); \quad \nu = 0, 1, \ldots \\
Akku & := & {}^r\boldsymbol{R}_1^{(\nu)} + {}^r\boldsymbol{R}_2^{(\nu)} + ({}^r\boldsymbol{R}_1^{(\nu)} + {}^r\boldsymbol{R}_2^{(\nu)}){}^r\boldsymbol{D}^{(\nu)}; \\
{}^r\boldsymbol{R}_1^{(\nu+1)} & := & \square(Akku);\ {}^r\boldsymbol{R}_2^{(\nu+1)} := \square(Akku - {}^r\boldsymbol{R}_1^{(\nu+1)})
\end{array}
$$

Beide Versionen der Defektiteration (54),(56) werden realisiert durch Aufruf der **PASCAL-XSC** Prozedur

```
INV_DEF( var A       : Lrmatrix; factor : integer;
         var R1,R2 : Lrmatrix; var Err : integer);

            exportiert durch linv
```

$\boldsymbol{A}$ ist die gegebene, zu invertierende, quadratische Matrix. Die Eingangsgrößen R1 := $\boldsymbol{R}_1^{(0)}$ und R2 := $\boldsymbol{R}_2^{(0)}$ definieren die Näherungsinverse, und die Ausgangsgrößen R1 := $\boldsymbol{R}_1^{(k)}$ und R2 := $\boldsymbol{R}_2^{(k)}$ liefern die i.a. maximalgenaue, doppeltlange Näherung der Inversen $\boldsymbol{A}^{-1}$:

$$\mathrm{R1} + \mathrm{R2} \approx \boldsymbol{A}^{-1}$$

Der Eingangsparameter factor bestimmt dabei die vom Anwender gewünschte Version der Defektiteration:

factor=0: Defektiteration, Version **links** nach (54), Seite 60

factor=1: Defektiteration, Version **rechts** nach (56), Seite 62

Liegt nun eine Matrix $\boldsymbol{A}$ vor, deren Inverse in einfacher Rechnung mit Hilfe der Prozedur INV(A,R,factor,Err) von Seite 51 nicht approximiert werden konnte, so wird man versuchen, das Problem durch doppeltlange Rechnung zu lösen. Dazu bestimmt man zunächst durch INV_RUMP(A,R1,R2,factor,Err) von Seite 42 mit dem Vefahren von Rump eine erste Näherung $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$.

Der Ausgangsparameter factor liefert dabei diejenige Version der Defektiteration, mit der man durch INV_DEF(A,factor,R1,R2,Err) eine maximalgenaue Näherungsinverse i.a. erfolgreich berechnen kann. Liefert diese letzte Prozedur wider Erwarten den Fehlercode Err $\neq 0$, so wird die Defektiteration mit der gleichen Eingangsnäherung, jedoch mit Hilfe der anderen Version erneut durchgeführt. Die geschilderte Vorgehensweise wird realisiert beim Aufruf der **PASCAL-XSC** Prozedur

INV(var A,R1,R2 : Lrmatrix; var factor,Err : integer); exportiert durch linv

Die Eingangsgröße $\boldsymbol{A}$ ist die zu invertierende Matrix, und $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ ist die i.a. maximalgenau Näherungsinverse. Im Fall Err=0 liefert die Ausgangsgröße factor die Version, mit der die interne Defektiteration erfolgreich durchgeführt wurde:

factor=0: Defektiteration, Version **links** nach (54), Seite 60

factor=1: Defektiteration, Version **rechts** nach (56), Seite 62

Die Angabe der benutzten Version mit Hilfe des Parameters factor ist z.B. dann sinnvoll, wenn mit Hilfe der obigen Näherungsinversen $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ eine garantierte Einschließung von $\boldsymbol{A}^{-1}$ zu berechnen ist.

Als erstes Beispiel betrachten wir noch einmal die Boothroyd/Dekker Matrizen $\boldsymbol{D}$, die mit der Prozedur INV(D,R,factor,Err) bei einfacher Rechnung nur bis zur Dimenison $n = 15$ erfolgreich invertiert werden können. Für den Bereich $16 \leq n \leq 27$ schalten wir daher durch Aufruf der genannten Prozedur INV(A,R1,R2,factor,Err) auf doppeltlange Rechnung um. Mit diesem Beispiel wird gezeigt, wie wichtig die richtige Versionswahl ist, mit der in der genannten Prozedur durch INV_DEF(D,factor,R1,R2,Err) die interne Defektiteration durchgeführt wird. In nachfolgender Tabelle haben wir dazu die benötigten Iterationsschrittzahlen angegeben, die bei den zwei möglichen Versionen erforderlich sind, dabei bedeutet $k = \infty$, daß auf der Maschine keine Konvergenz erreicht wurde.

Bitte beachten Sie, daß durch INV_RUMP(D,R1,R2,factor,Err) in der obigen Prozedur INV(D,R1,R2,factor,Err) durch factor die i.a. günstigste Version der Defektiteration fest vorgeschrieben wird. In der nachfolgenden Tabelle sind zum Vergleich auch diejenigen Ergebnisse angegeben, die man mit Hilfe der jeweils anderen Version erhalten würde.

Inversion der Dekker–Matrizen D					
Berechnung der 1. Näherung durch: INV_RUMP(D,R1,R2,factor,Err)					
Vergleich der Schrittzahlen k der beiden möglichen Defektiterationen durch Aufruf von INV_DEF(D,factor,R1,R2,Err) factor=0 $\leftrightarrow$ Version links, factor=1 $\leftrightarrow$ Version rechts;					
n	Version links	Version rechts	n	Version links	Version rechts
	Schrittzahl k	Schrittzahl k		Schrittzahl k	Schrittzahl k
15	4	4	23	∞	4
16	4	4	24	∞	4
20	6	4	25	∞	4
21	6	4	26	∞	5
22	∞	4	27	∞	5

Zunächst erkennt man, daß die beiden Versionen bis $n = 16$ völlig gleichwertig sind. Bis $n = 21$ erhält man in beiden Fällen zwar noch Konvergenz, jedoch benötigt die ungünstigere Version links (factor=0) zwei zusätzliche Iterationsschritte, und für $n \geq 22$ erhält man Konvergenz nur noch mit Hilfe der Version rechts (factor=1), die auch durch INV(D,R1,R2,factor,Err) mit factor stets als die intern benutzte Version angegeben wird. Mit diesen Ergebnissen formulieren wir den folgenden Merksatz:

Ist eine mit dem Verfahren von Rump berechnete Ausgangsnäherung vorgegeben, so ist bei schlechtkonditionierten Matrizen für eine nachfolgende Defektiteration die Wahl der richtigen Version links oder rechts von entscheidender Bedeutung.

Für ein weiteres Beispiel definieren wir die Matrix:

$$\boldsymbol{A}(\varepsilon) \;:=\; \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & \varepsilon \end{pmatrix} \quad \text{mit der Inversen:}$$

$$\boldsymbol{A}^{-1}(\varepsilon) \;:=\; \begin{pmatrix} 1 & -1 & \frac{1}{3}(1-10^{-21}) & 10^{-81}/\varepsilon \\ 0 & 1 & -\frac{1}{3} & \frac{1}{3}\cdot 10^{-21}/\varepsilon \\ 0 & 0 & \frac{1}{3} & -\frac{1}{3}/\varepsilon \\ 0 & 0 & 0 & 1/\varepsilon \end{pmatrix}$$

Wir wählen $\varepsilon = (1 + 10^{-10}) \cdot 10^{-40}$ und erreichen damit, daß z.B. die Mantisse von $\boldsymbol{A}^{-1}(\varepsilon)[1,4]$ aus 10-ziffrigen Blöcken aufgebaut wird, die abwechselnd jeweils nur die Ziffer 9 bzw. 0 enthalten, wodurch die Inversion deutlich erschwert wird. Zusätzlich wird $\boldsymbol{A}(\varepsilon)$ durch die obige Wahl von ε sehr schlecht konditioniert, da die Konditionszahl etwa die gleiche Größenordnung besitzt wie $1/\varepsilon \approx 10^{+40}$. Trotzdem liefert der Prozeduraufruf INV(A,R1,R2,factor,Err) die folgende, doppeltlange und **maximalgenaue** Näherung:

$$\mathrm{R1} \;:=\; \begin{pmatrix} 1 & -1 & \frac{1}{3}(1-10^{-21}) & +9.9999999990\ldots010\cdot10^{-42} \\ 0 & 1 & -\frac{1}{3}(1-10^{-21}) & +3.3333333330\ldots003\cdot10^{+18} \\ 0 & 0 & \frac{1}{3}(1-10^{-21}) & -3.3333333330\ldots003\cdot10^{+39} \\ 0 & 0 & 0 & +9.9999999990\ldots010\cdot10^{+39} \end{pmatrix} = \boldsymbol{R}_1$$

$$\mathrm{R2} \;:=\; \begin{pmatrix} 0 & 0 & 0 & -9.9999999990\ldots010\cdot10^{-72} \\ 0 & 0 & -\frac{1}{3}(1-10^{-21})\cdot10^{-21} & +3.3333333300\ldots033\cdot10^{-03} \\ 0 & 0 & \frac{1}{3}(1-10^{-21})\cdot10^{-21} & -3.3333333300\ldots033\cdot10^{+18} \\ 0 & 0 & 0 & -9.9999999990\ldots010\cdot10^{+09} \end{pmatrix} = \boldsymbol{R}_2$$

Damit haben wir auch für eine ziemlich schlecht konditionierte Matrix $\boldsymbol{A}(\varepsilon)$ mit Hilfe einer doppeltlangen Defektiteration der Version 'links' (factor=0) die Inverse $\boldsymbol{A}^{-1}(\varepsilon)$ maximalgenau berechnet.

In nachfolgender Tabelle haben wir für die schon auf Seite 40 definierten Blockmatrizen $\boldsymbol{B} = \boldsymbol{B}(n, v)$ die Werte der Ausgangsgrößen factor,Err beim Aufruf der Prozedur INV(B,R1,R2,factor,Err) zusammengestellt:

INV(B,R1,R2,factor,Err)					
$\boldsymbol{B} = \boldsymbol{B}(n, 10^{-30})$; $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{B}^{-1}$					
factor=0 $\leftrightarrow$ Defektiteration, Version links					
factor=1 $\leftrightarrow$ Defektiteration, Version rechts					
Err = 0*: Approximation nicht maximalgenau					
n	Err	factor	n	Err	factor
1	0	1	7	0*	1
2	0	1	8	0	1
3	0*	1	9	3	
4	0	1	10	0	1
5	0	1	11	0*	1
6	0*	0	12	0*	1

Für $n = 3$ liegt auf der Maschine zwar Konvergenz vor, aber die Approximation $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{B}^{-1}(3, 10^{-30})$ wird nicht maximalgenau berechnet. Wir formulieren daher den Merksatz:

> *In konstruierten Fällen kann die Defektiteration auf der Maschine zwar konvergieren, ohne daß dabei aber die Inverse maximalgenau approximiert wird.*

In diesen Fällen wird man eine Einschließung der Inversen i.a. nur erhalten, wenn man den Algorithmus der LU–Zerlegung recht zeitaufwendig in *staggered arithmetic* durchführt.

Für Kontrollrechnungen wird neben einer Blockmatrix $\boldsymbol{B}$ selbst auch noch die doppeltlange Approximation ihrer Inversen angegeben. Mit $n = 4$ ist die Blockmatrix $\boldsymbol{B}$ der Dimension 8 gegeben durch:

$$\boldsymbol{B}(4, 10^{-30}) := \begin{pmatrix} 1 & 1 & 1 & 1 & 10^{-30} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 10^{-30} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 10^{-30} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 10^{-30} \\ 10^{-30} & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 10^{-30} & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 10^{-30} & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 10^{-30} & 1 & 1 & 1 & 1 \end{pmatrix}$$

und die mit INV(B,R1,R2,factor,Err) maximalgenaue berechnete Näherungsinverse $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{B}^{-1}(4, 10^{-30})$ lautet:

$$\boldsymbol{R1} := \begin{pmatrix} \alpha & \alpha & \alpha & \alpha & \boldsymbol{\delta} & \beta & \beta & \beta \\ \alpha & \alpha & \alpha & \alpha & \beta & \boldsymbol{\delta} & \beta & \beta \\ \alpha & \alpha & \alpha & \alpha & \beta & \beta & \boldsymbol{\delta} & \beta \\ \alpha & \alpha & \alpha & \alpha & \beta & \beta & \beta & \boldsymbol{\delta} \\ \boldsymbol{\delta} & \beta & \beta & \beta & \alpha & \alpha & \alpha & \alpha \\ \beta & \boldsymbol{\delta} & \beta & \beta & \alpha & \alpha & \alpha & \alpha \\ \beta & \beta & \boldsymbol{\delta} & \beta & \alpha & \alpha & \alpha & \alpha \\ \beta & \beta & \beta & \boldsymbol{\delta} & \alpha & \alpha & \alpha & \alpha \end{pmatrix} \quad \text{mit:} \quad \begin{aligned} \alpha &:= 6.25 \cdot 10^{-2} \\ \beta &:= -2.5 \cdot 10^{+29} \\ \delta &:= +7.5 \cdot 10^{+29} \end{aligned}$$

$$
\boldsymbol{R2} := \begin{pmatrix} \mu & \mu & \mu & \mu & \nu & \nu & \nu & \nu \\ \mu & \mu & \mu & \mu & \nu & \nu & \nu & \nu \\ \mu & \mu & \mu & \mu & \nu & \nu & \nu & \nu \\ \mu & \mu & \mu & \mu & \nu & \nu & \nu & \nu \\ \nu & \nu & \nu & \nu & \mu & \mu & \mu & \mu \\ \nu & \nu & \nu & \nu & \mu & \mu & \mu & \mu \\ \nu & \nu & \nu & \nu & \mu & \mu & \mu & \mu \\ \nu & \nu & \nu & \nu & \mu & \mu & \mu & \mu \end{pmatrix} \quad \text{mit:} \quad \begin{array}{l} \mu := \ 3.90625 \cdot 10^{-63} \\ \nu := -1.5625 \cdot 10^{-32} \end{array}
$$

Die Ergebisse lassen vermuten, daß die Inverse einer Blockmatrix selbst wieder die Struktur einer Blockmatrix besitzt.

Im letzten Beispiel betrachten wir noch einmal die schon auf Seite 47 behandelte Matrix:

$$
\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix} \quad \text{besitzt die maximalgenaue Näherungsinverse}
$$

$$
\boldsymbol{R}_A := \begin{pmatrix} -10^{20}+1 & 0 & 10^{20} & -1+10^{-20} \\ -10^{20}+1 & 10^{20} & 0 & -1+10^{-20} \\ 1-10^{-20} & 0 & 0 & -1+10^{-20} \\ 2\cdot 10^{20}-2 & -10^{20} & -10^{20} & 3-2\cdot 10^{-20} \end{pmatrix} \approx \boldsymbol{A}^{-1}
$$

Mit dem Prozeduraufruf INV(A,R1,R2,factor,Err) erhält man die maximalgenaue Näherungsinverse $\boldsymbol{R}_1 + \boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$, wobei factor=1 angibt, daß bei der internen Defektiteration die Version 'rechts' zur Anwendung kam. Die Ergebnisse lauten:

$$
\begin{aligned}
\boldsymbol{R}_1 &= \begin{pmatrix} -10^{20}+1 & 0 & 10^{20} & -1+10^{-20} \\ -10^{20}+1 & 10^{20} & 0 & -1+10^{-20} \\ 1-10^{-20} & 0 & 0 & -1+10^{-20} \\ 2\cdot 10^{20}-2 & -10^{20} & -10^{20} & 3-2\cdot 10^{-20} \end{pmatrix} = \boldsymbol{R}_A \\
\boldsymbol{R}_2 &= \begin{pmatrix} -10^{-20}+2\cdot 10^{-40} & 0 & 0 & -2\cdot 10^{-40}+3\cdot 10^{-60} \\ -10^{-20}+2\cdot 10^{-40} & 0 & 0 & -2\cdot 10^{-40}+3\cdot 10^{-60} \\ 2\cdot 10^{-40}-3\cdot 10^{-60} & 0 & 0 & -2\cdot 10^{-40}+3\cdot 10^{-60} \\ 2\cdot 10^{-20}-5\cdot 10^{-40} & 0 & 0 & 5\cdot 10^{-40}-7\cdot 10^{-60} \end{pmatrix}
\end{aligned}
$$

7 Sehr schlecht konditionierte Matrizen

In den vorangegangenen Abschnitten haben wir zu einer gegebenen, regulären Matrix $\boldsymbol{A}$ eine i.a. maximalgenaue Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ nach folgendem Schema berechnet:

1. Man bestimmt mit Hilfe der LU–Zerlegung eine Näherung $\boldsymbol{R}^{(0)}$ der Inversen $\boldsymbol{A}^{-1}$ in einfacher Länge des benutzten Datenformats. Eine nachfolgende Defektiteration liefert dann die gesuchte, i.a. maximalgenaue Näherungsinverse, wobei die benutzte Defektdefinition angegeben wird. Führt dieses Verfahren bei schlechtkonditionierten Matrizen nicht zum Ziel, so rechnet man in doppelter Länge:

2. Mit dem Verfahren von Rump bestimmt man in doppelter Länge des benutzten Datenformats eine erste Näherung $\boldsymbol{R}_1^{(0)} + \boldsymbol{R}_2^{(0)}$ der Inversen, und eine nachfolgende Defektiteration liefert dann die i.a. maximalgenaue Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$, wobei auch jetzt die dabei benutzte Defektdefinition angegeben wird.

Bei einfacher Rechnung erfolgt die Auswertung mit der **PASCAL-XSC** Prozedur INV(A,R,factor,Err) von Seite 51 und bei doppeltlanger Rechnung mit Hilfe der Prozedur INV(A,R1,R2,factor,Err) von Seite 63. In vielen eindrucksvollen Beispielen, z.B. auf den Seiten 64,67, wurde dabei gezeigt, daß die letzte Prozedur auch bei schlechtkonditionierten Matrizen noch erfolgreich arbeitet. Andererseits zeigt die Tabelle auf Seite 65, daß mit der gleichen Prozedur die Inversion der auf Seite 40 definierten Blockmatrizen $\boldsymbol{B} = \boldsymbol{B}(n, 10^{-30})$ z.B. für die Dimensionen $n = 3, 9$ auch bei doppeltlanger Rechnung nicht mehr maximalgenau erfolgt. In diesem Zusammenhang zitieren wir daher den folgenden Merksatz:

> *Ein auf Gleitkommarechnungen aufgebautes Iterationsverfahren, das selbst bei kritischen Anwendungen noch hervorragend konvergiert, wird völlig unbrauchbar, wenn das Problem nur hinreichend schlecht konditioniert ist.*

Liegt nun der Fall vor, daß INV(A,R1,R2,factor,Err) von Seite 63 die Inverse $\boldsymbol{A}^{-1}$ nicht mehr maximalgenau approximiert, so bleibt nur noch die Möglichkeit, die notwendigen Rechnungen sehr zeitaufwendig in *staggered arithmetic* durchzuführen.

7.1 Approximation in staggered arithmetic

Wir wollen hier nicht alle Einzelheiten der *staggered arithmetic* angeben, da man diese in den Beschreibungen der Hilfsmodule `lstagg` bzw. `lstgmv` in [2] ausführlich nachlesen kann. Das Ergebnis s einer Rechnung in *staggered arithmetic* ist gegeben durch:

$$s := s_1 + s_2 + \ldots + s_p + [S_1, S_2], \qquad p \geq 1,$$

wobei der Genauigkeitsparameter p mit Hilfe der **PASCAL-XSC** Prozedur

SET_LSTAGG_PREC(p : integer); exportiert durch `type_def`

vor der eigentlichen Rechnung festzulegen ist. Die p Summanden s_i sind in unserem Fall vom Typ Longreal und sollen sich nicht überlappen. Das zusätzliche Fehler-Intervall $[S_1, S_2]$ mit den Longrealgrenzen S_1, S_2 ermöglicht dann eine Einschließung des exakten Wertes a durch das schmale Intervall s, wobei dann die Genauigkeit durch etwa $(p+1) \cdot 21$ Dezimalziffern gegeben ist. Wird nun das exakte Ergebnis b durch ein entsprechendes Intervall t eingeschlossen, so gilt z.B. $a \cdot b \in s \odot t$, wenn $\odot$ den vom Modul LSTAGG exportierten Multiplikations-Operator bedeutet. Damit sind Operationen in *staggered arithmetic* reine Intervalloperationen, wobei die bekannten *wrapping*-Effekte bei hinreichend großem p so weit nach hinten verlagert werden, daß sie in der Praxis i.a. keine Rolle mehr spielen. Ein weiterer Vorteil der *staggered arithmetic* besteht darin, daß ein Intervall, dessen Genauigkeit durch ca. $(p+1) \cdot 21$ Dezimalziffern gegeben ist, mit nur $(p+2)$ Longreal-Zahlen darstellbar ist!

Formuliert man nun die LU–Zerlegung einer zu invertierenden Matrix $\boldsymbol{A}$ mit anschließender Vorwärts- und Rückwärts-Substitution (vergl. Seite 19,...) in *staggered arithmetic*, so erhält man automatisch eine Einschließung der gesuchten Inversen $\boldsymbol{A}^{-1}$. Diese Vorgehensweise wird realisiert beim Aufruf der **PASCAL-XSC** Prozedur

INV(var A: Lrmatrix; var S: LStaggrmatrix; var Err : integer); exportiert durch linv

Die Eingangsgröße $\boldsymbol{A}$ ist die zu invertierende Matrix, und die Ausgangsgröße $\boldsymbol{S}$ liefert eine garantierte Einschließung der gesuchten Inversen $\boldsymbol{A}^{-1} \in \boldsymbol{S}$. Der Genauigkeitsparameter p ist dabei mit Hilfe der schon genannten Prozedur SET_LSTAGG_PREC(p: integer) vorher entsprechend festzulegen. Beachten Sie bitte, daß der Fehlercode Err=0 lediglich eine fehlerfreie LU–Zerlegung von $\boldsymbol{A}$ garantiert und nicht etwa die Berechnung einer maximalgenauen Näherungsinversen; letztere erhält man nur mit einem hinreichend großem Genauigkeitsparameter p.

Rundet man jetzt nach Aufruf der Prozedur INV(A,S,Err) alle Mittelpunkte der mit *staggered arithmetic* berechneten Elemente von $\boldsymbol{S}$ durch Aufruf der Prozedur TRANS(S,R1,R2) ins doppeltlange Longreal–Format, so erhält man mit $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ eine i.a. maximalgenaue Näherungsinverse. In Ausnahmesituationen fallen die Intervalle der $\boldsymbol{S}[i,j]$ auch bei noch so großem p zu breit aus, so daß durch die Prozedur TRANS(S,R1,R2) nicht mehr alle Summen $\boldsymbol{R1}[i,j] + \boldsymbol{R2}[i,j]$ maximalgenau bestimmt werden können, d.h. die Näherung $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ kann in Ausnahmefällen auch bei großen p-Werten relativ ungenau ausfallen. Der beschriebene Algorithmus wird realisiert beim Aufruf der **PASCAL-XSC** Prozedur

INV_ST(var A,R1,R2 : Lrmatrix; var Err : integer); exportiert durch linv

Die Summe $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ ist dann im Falle Err=0 die möglicherweise maximalgenaue Näherung in doppelter Länge des Longreal-Formats, wenn man vorher den Genauigkeitsparameter p geeignet gewählt hat.

Als erstes Beispiel betrachten wir die sehr schlecht konditionierte Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & 10^{-100} \end{pmatrix}; \qquad \varkappa = 4.10 \cdot 10^{+100}$$

Mit $p = 5$ erhält man mit Err=0 die maximalgenaue, doppeltlange Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$:

$$\boldsymbol{R1} := \begin{pmatrix} 0.75 & -0.75 & +0.25 & +7.5 \cdot 10^{+18} \\ 0.25 & 0.75 & -0.25 & +3.33\ldots 33 \cdot 10^{+78} \\ -0.25 & 0.25 & +0.25 & -3.33\ldots 33 \cdot 10^{+99} \\ 0 & 0 & 0 & 10^{+100} \end{pmatrix}$$

$$
\boldsymbol{R2} := \begin{pmatrix} +1.875\cdot 10^{-22} & -1.875\cdot 10^{-22} & -1.875\cdot 10^{-22} & +1.875\cdot 10^{-3} \\ +6.25\cdot 10^{-23} & -6.25\cdot 10^{-23} & -6.25\cdot 10^{-23} & +3.33\ldots 33\cdot 10^{+57} \\ -6.25\cdot 10^{-23} & +6.25\cdot 10^{-23} & +6.25\cdot 10^{-23} & -3.33\ldots 33\cdot 10^{+78} \\ 0 & 0 & 0 & 0 \end{pmatrix}
$$

Im folgenden Beispiel wählen wir die ebenfalls sehr schlecht konditionierte Matrix:

$$
\boldsymbol{B} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 1 & 0 & 0 & 10^{-100} \end{pmatrix}; \qquad \varkappa = 8.34\cdot 10^{+81}
$$

In nachfolgender Tabelle sind die durch den Aufruf INV_ST(B,R1,R2,Err) mit dem Genauigkeitsparameter $p = 10$ maximalgenau berechneten Elemente $\boldsymbol{R1}[i,j]$ bzw. $\boldsymbol{R2}[i,j]$ zusammengestellt:

INV_ST(B,R1,R2,Err);	Genauigkeitsparameter $p = 10$
Maximalgenaue Näherung: $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{B}^{-1}$	
$\boldsymbol{R1}[1,1] = +9.99999999999999999867\cdot 10^{-20}$	$\boldsymbol{R1}[1,2] = -9.99999999999999999867\cdot 10^{-20}$
$\boldsymbol{R1}[1,3] = +3.33333333333333333289\cdot 10^{-20}$	$\boldsymbol{R1}[1,4] = +9.99999999999999999867\cdot 10^{-1}$
$\boldsymbol{R1}[2,1] = -3.33333333333333333289\cdot 10^{+59}$	$\boldsymbol{R1}[2,2] = +3.33333333333333333289\cdot 10^{+59}$
$\boldsymbol{R1}[2,3] = -1.11111111111111111096\cdot 10^{+59}$	$\boldsymbol{R1}[2,4] = +4.44444444444444444385\cdot 10^{+59}$
$\boldsymbol{R1}[3,1] = +3.33333333333333333289\cdot 10^{+80}$	$\boldsymbol{R1}[3,2] = -3.33333333333333333289\cdot 10^{+80}$
$\boldsymbol{R1}[3,3] = +1.11111111111111111096\cdot 10^{+80}$	$\boldsymbol{R1}[3,4] = -4.44444444444444444385\cdot 10^{+80}$
$\boldsymbol{R1}[4,1] = -9.99999999999999999867\cdot 10^{+80}$	$\boldsymbol{R1}[4,2] = +9.99999999999999999867\cdot 10^{+80}$
$\boldsymbol{R1}[4,3] = -3.33333333333333333289\cdot 10^{+80}$	$\boldsymbol{R1}[4,4] = +1.33333333333333333316\cdot 10^{+81}$
$\boldsymbol{R2}[1,1] = -3.33333333333333315522\cdot 10^{-41}$	$\boldsymbol{R2}[1,2] = +3.33333333333333315522\cdot 10^{-41}$
$\boldsymbol{R2}[1,3] = -4.44444444444444438463\cdot 10^{-41}$	$\boldsymbol{R2}[1,4] = -3.33333333333333315522\cdot 10^{-22}$
$\boldsymbol{R2}[2,1] = +1.11111111111111105174\cdot 10^{+38}$	$\boldsymbol{R2}[2,2] = -1.11111111111111105174\cdot 10^{+38}$
$\boldsymbol{R2}[2,3] = -1.85185185185185187179\cdot 10^{+38}$	$\boldsymbol{R2}[2,4] = +7.40740740740740820049\cdot 10^{+37}$
$\boldsymbol{R2}[3,1] = -1.11111111111111105174\cdot 10^{+59}$	$\boldsymbol{R2}[3,2] = +1.11111111111111105174\cdot 10^{+59}$
$\boldsymbol{R2}[3,3] = +1.85185185185185187179\cdot 10^{+59}$	$\boldsymbol{R2}[3,4] = -7.40740740740740820049\cdot 10^{+58}$
$\boldsymbol{R2}[4,1] = +3.33333333333333315522\cdot 10^{+59}$	$\boldsymbol{R2}[4,2] = -3.33333333333333315522\cdot 10^{+59}$
$\boldsymbol{R2}[4,3] = +4.44444444444444438463\cdot 10^{+59}$	$\boldsymbol{R2}[4,4] = -4.77777777777777775399\cdot 10^{+60}$

Beachten Sie bitte, daß mit $p = 10$ etwa $(p+1)\cdot 21 = 231$ interne Dezimalstellen zur Berechnung der Näherungsinversen benötigt werden.

8 Einschließung der Inversen

In den letzten Abschnitten wurden Verfahren vorgestellt, mit denen man zu einer gegebenen regulären Matrix $\boldsymbol{A}$ fast immer eine maximalgenaue Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ iterativ berechnen konnte. In der Tabelle auf Seite 65 haben wir jedoch mit den Blockmatrizen $\boldsymbol{B}(n, 10^{-30})$ für $n = 3, 6, 7, 11, 12$ Beispiele kennengelernt, bei denen wir auf dem Rechner zwar Konvergenz erhalten, allerdings nicht bei allen Komponenten gegen diejenige Rasterzahl, die dem jeweils exakten Wert am nächsten liegt. Damit liefern die bisherigen Iterationsverfahren auch bei Konvergenz auf der Maschine lediglich Punktnäherungen $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$, über deren Genauigkeit gesicherte Aussagen nicht möglich sind! Um die gefundenen Ergebnisse jedoch wirklich mathematisch beurteilen zu können, stellen wir in den nachfolgenden Abschnitten Algorithmen vor, mit denen man **garantierte** und möglichst enge **Einschließungen** aller Elemente der Inversen berechnen kann; behandelt werden:

1. Ein Iterationsverfahren, bei dem für eine Korrektur $\boldsymbol{Y}$ aus dem Ansatz $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ eine Fixpunkt–Gleichung aufgestellt wird, deren Fixpunkt nach dem Satz von Brouwer und den Ergebnissen von S.M. Rump garantiert eingeschlossen werden kann.

2. Ein von G. Alefeld und J. Herzberger angegebenes Iterationsverfahren [1], bei dem eine schon bekannte Einschließung der Inversen i.a. weiter verbessert wird.

3. Ein sehr aufwendiges Verfahren, bei dem der Crout'sche Algorithmus zur LU–Zerlegung in *staggered arithmetic* intervallmäßig ausgewertet wird, womit man dann eine Einschließung der Inversen automatisch erhält.

Neben den genannten Verfahren kann man eine Einschließung der Inversen $\boldsymbol{A}^{-1}$ auch dadurch gewinnen, daß man die Lösungsvektoren der n linearen Gleichungssysteme (7) auf Seite 16 mit Hilfe des Fixpunktsatzes von Brouwer einschließt. Da man hierzu in den Toolbox–Bänden I,II [3, 4] alles Wesentliche nachlesen kann, wird hier auf eine weitere Diskussion verzichtet. In den folgenden Abschnitten werden wir uns daher nur mit den drei ersten Einschließungsverfahren beschäftigen. Eine zentrale Rolle spielt dabei die in Punkt 1. genannte Fixpunktiteration, die bei Bedarf auch in doppelter Länge durchgeführt werden kann.

8.1 Einschließung der Korrekturmatrix

Zu einer gegeben Matrix $\boldsymbol{A} \in I\!R^{n\times n}$ sei eine möglichst maximalgenaue Punktnäherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ bereits berechnet. Mit dem Ansatz

$$\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y} \tag{57}$$

und dem Defekt ${}^{r}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{AR}$ folgt aus $\boldsymbol{AA}^{-1} = \boldsymbol{I}$ für die Korrektur $\boldsymbol{Y}$ die einfache Beziehung:

$$\boldsymbol{AY} = {}^{r}\boldsymbol{D} \quad \Longleftrightarrow \quad \boldsymbol{Ay}_j = {}^{r}\boldsymbol{d}_j, \quad j = 1, 2, \ldots, n; \tag{58}$$

Wie auf Seite 16 reduziert sich damit die Berechnung von $\boldsymbol{Y}$ auf die Lösung n linearer Gleichungssysteme, wobei $\boldsymbol{y}_j$ bzw. ${}^{r}\boldsymbol{d}_j$ der jeweils j-te Spaltenvektor von $\boldsymbol{Y}$ bzw. ${}^{r}\boldsymbol{D}$ ist. Aus Laufzeitgründen werden wir in der Praxis diese n Systeme jedoch nicht einzeln lösen! Für die Matrix $\boldsymbol{Y}$ werden wir vielmehr eine Fixpunktgleichung aufstellen, aus der sich dann wieder n vektorielle Iterationsgleichung ergeben, die dann nach S.M. Rump die Grundlage zur Einschließung der n Lösungsvektoren $\boldsymbol{y}_j$ bilden. Im nächsten Abschnitt werden daher die benötigten Ergebnisse der Rump'schen Methode zusammengestellt.

8.1.1 Ergebnisse von Rump

Gegeben sei das folgende lineare Gleichungssystem

$$\boldsymbol{Ax} = \boldsymbol{b}; \qquad \boldsymbol{A} \in I\!R^{n\times n} \quad \text{und} \quad \boldsymbol{x}, \boldsymbol{b} \in I\!R^{n};$$

Die nachfolgende Gleichungskette

$$\begin{aligned} \boldsymbol{0} &= \boldsymbol{R}(\boldsymbol{b} - \boldsymbol{Ax}) \\ \boldsymbol{x} &= \boldsymbol{Rb} + \boldsymbol{x} - \boldsymbol{RAx} \\ \boldsymbol{x} &= \boldsymbol{Rb} + (\boldsymbol{I} - \boldsymbol{RA})\boldsymbol{x} \end{aligned}$$

führt dann zur Fixpunktgleichung $\boldsymbol{x} = \boldsymbol{Rb} + (\boldsymbol{I} - \boldsymbol{RA})\boldsymbol{x}$, aus der sich die folgende Iterationsgleichung ergibt:

$$[\boldsymbol{x}]^{(\nu+1)} := \boldsymbol{Rb} + {}^{l}\boldsymbol{D}[\boldsymbol{x}]^{(\nu)}, \qquad {}^{l}\boldsymbol{D} := (\boldsymbol{I} - \boldsymbol{RA}), \quad \nu = 0, 1, \ldots \tag{59}$$

Findet man nun im Laufe einer solchen Iteration einen speziellen Index k, für den die Intervallvektoren $[\boldsymbol{x}]^{(k+1)}, [\boldsymbol{x}]^{(k)}$ die folgende Voraussetzung[3]

$$[\boldsymbol{x}]^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{x}]^{(k)} \tag{60}$$

[3] Wegen $\overset{\circ}{\subset}$ muß $[\boldsymbol{x}]^{(k+1)}$ ganz im Innern von $[\boldsymbol{x}]^{(k)}$ enthalten sein, wobei $[\boldsymbol{x}]^{(k)}$ entweder als Iterierte aus (59) oder sonst irgendwie gegeben ist.

erfüllen, so gelten nach Rump [5, Seite 119] die Aussagen:

- Die Matrizen $\boldsymbol{A}, \boldsymbol{R}$ sind regulär
- $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$ besitzt eine eindeutige Lösung $\boldsymbol{x} \in [\boldsymbol{x}]^{(k+1)}$

Die Intervall–Addition und die zwei Multiplikationen sind in (59) **exakt** auszuführen, so daß man $[\boldsymbol{x}]^{(k+1)}$ auf einem Rechner mit fehlerbehafteten Gleitkomma–Operationen i.a. nicht berechnen kann. Für die Anwendung stellt sich damit das Problem, die Voraussetzung $[\boldsymbol{x}]^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{x}]^{(k)}$ auf einem Rechner überprüfen zu können. Zur Lösung werden wir mit einem Intervallvektor $[\boldsymbol{x}]^{(k)}$ des Gleitkommasystems die rechte Seite von (59) intervallmäßig auswerten:

$$[\boldsymbol{x}]_{\diamond}^{(k+1)} := \boldsymbol{R} \mathbin{\diamond\!\!\!\cdot}\, \boldsymbol{b} \mathbin{\diamond\!\!\!\!+} [{}^l\boldsymbol{D}] \mathbin{\diamond\!\!\!\cdot}\, [\boldsymbol{x}]^{(k)}; \qquad [{}^l\boldsymbol{D}] := \diamond(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{A})$$

Die bekannte Einschließungseigenschaft der Intervallrechnung garantiert nun die Beziehung $[\boldsymbol{x}]^{(k+1)} \subseteq [\boldsymbol{x}]_{\diamond}^{(k+1)}$ und dies führt zu folgendem Ergebnis:

$$[\boldsymbol{x}]_{\diamond}^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{x}]^{(k)} \quad \Longrightarrow \quad [\boldsymbol{x}]^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{x}]^{(k)}$$

Wenn also auf einem Rechner die Bedingung

$$[\boldsymbol{x}]_{\diamond}^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{x}]^{(k)} \tag{61}$$

mit Hilfe einer vorhandenen Intervallarithmetik realisiert wurde, so gilt damit auch die Voraussetzung (60) und schließlich:

$$\boldsymbol{x} \in [\boldsymbol{x}]^{(k+1)} \subseteq [\boldsymbol{x}]_{\diamond}^{(k+1)} \quad \Longrightarrow \quad \boldsymbol{x} \in [\boldsymbol{x}]_{\diamond}^{(k+1)} \tag{62}$$

Trotz fehlerbehafteter Gleitkommaoperationen kann man daher auf einem Rechner die Verifikationsvoraussetzung (60) erfolgreich überprüfen und damit garantierte Einschließungen der eindeutigen Lösung eines linearen Gleichungssystems berechnen.

Um Überschätzungen bei der Intervallauswertung von $[\boldsymbol{x}]_{\diamond}^{(k+1)}$ zu vermeiden, sollte man Einschließungen der Defektmatrix ${}^l\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}$ grundsätzlich nur mit Hilfe des langen Akkumulators berechnen. Mit $\boldsymbol{R}\boldsymbol{A} \approx \boldsymbol{I}$ erhält man nämlich bei naiver Intervallauswertung von ${}^l\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}$ ohne den Akkumulator wegen starker Auslöschungseffekte viel zu große Überschätzungen!

Wir wollen jetzt die Ergebnisse von Rump auch in Matrixform schreiben, da es ja stets möglich ist, eine Matrixgleichung

$$(63) \qquad \boldsymbol{AX} = \boldsymbol{B}; \qquad \boldsymbol{A}, \boldsymbol{X}, \boldsymbol{B} \in I\!R^{n \times n}$$

als n voneinander unabhängige lineare Gleichungssysteme aufzufassen. Sind daher $\boldsymbol{x} = \boldsymbol{x}_j$ bzw. $\boldsymbol{b} = \boldsymbol{b}_j$ die Spaltenvektoren der Matrizen $\boldsymbol{X}$ bzw. $\boldsymbol{B}$, so kann man nach (59) die n Iterationsgleichungen:

$$(64) \qquad [\boldsymbol{x}]_j^{(\nu+1)} := \boldsymbol{R}\boldsymbol{b}_j + {}^l\boldsymbol{D}[\boldsymbol{x}]_j^{(\nu)}, \qquad j = 1, 2, \ldots, n$$

wieder als Matrix–Iterationsgleichung zusammenfassen:

$$(65) \qquad [\boldsymbol{X}]^{(\nu+1)} := \boldsymbol{RB} + {}^l\boldsymbol{D}[\boldsymbol{X}]^{(\nu)}, \quad {}^l\boldsymbol{D} := (\boldsymbol{I} - \boldsymbol{RA}), \quad \nu = 0, 1, \ldots$$

Findet man nun im Laufe einer solchen Iteration einen speziellen Index k, für den die Intervallmatrizen $[\boldsymbol{X}]^{(k+1)}, [\boldsymbol{X}]^{(k)}$ die folgende Voraussetzung[4]

$$(66) \qquad [\boldsymbol{X}]^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{X}]^{(k)}$$

erfüllen, so gelten nach Rump die entsprechenden Aussagen:

- Die Matrizen $\boldsymbol{A}, \boldsymbol{R}$ sind regulär
- $\boldsymbol{AX} = \boldsymbol{B}$ besitzt eine eindeutige Lösung $\boldsymbol{X} \in [\boldsymbol{X}]^{(k+1)}$

Die in (65) auftretenden Intervalloperationen sind wieder **exakt** auszuführen, so daß die Verifikations–Voraussetzung (66) auf einem Gleitkommarechner nicht direkt überprüft werden kann. Wenn jedoch die Intervallmatrix $[\boldsymbol{X}]_\diamond^{(k+1)}$ mit Hilfe einer vorhandenen Matrix–Intervallarithmetik nach folgender Gleichung

$$(67) \qquad [\boldsymbol{X}]_\diamond^{(k+1)} := \boldsymbol{R} \diamond \boldsymbol{B} \mathbin{\diamond\!\!\!\!+} [{}^l\boldsymbol{D}] \diamond [\boldsymbol{X}]^{(k)}, \quad [{}^l\boldsymbol{D}] := \diamond(\boldsymbol{I} - \boldsymbol{RA}),$$

berechnet wird, so ist ganz analog zu den Überlegungen von Seite 74 die Verifikationsvoraussetzung (66) automatisch erfüllt, wenn auf dem Rechner die Bedingung:

$$(68) \qquad [\boldsymbol{X}]_\diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{X}]^{(k)}$$

realisiert ist. Damit haben wir das Ergebnis:

$$(69) \qquad [\boldsymbol{X}]_\diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{X}]^{(k)} \implies \boldsymbol{X} \in [\boldsymbol{X}]_\diamond^{(k+1)} \text{ ist eindeutig.}$$

[4]Wegen $\overset{\circ}{\subset}$ muß $[\boldsymbol{X}]^{(k+1)}$ ganz im Innern von $[\boldsymbol{X}]^{(k)}$enthalten sein.

8.1.2 Einschließung der Korrektur auf der Maschine

Wir kommen wieder zum Ansatz (57) auf Seite 73 zurück:

$$\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$$

Es gilt dann: $(\boldsymbol{R}+\boldsymbol{Y})\boldsymbol{A} = \boldsymbol{I}$ bzw. $\boldsymbol{RA} - \boldsymbol{I} + \boldsymbol{YA} = \boldsymbol{0}$. Mit dem Defekt ${}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{RA}$ folgt: $\boldsymbol{YA} = {}^{l}\boldsymbol{D}$ und nach Multiplikation von rechts mit $(\boldsymbol{R}+\boldsymbol{Y}) = \boldsymbol{A}^{-1}$ erhält man schließlich:

$$\boldsymbol{Y} = {}^{l}\boldsymbol{D}(\boldsymbol{R}+\boldsymbol{Y}) \tag{70}$$

Mit dem Defekt ${}^{r}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{AR}$ findet man ganz analog die Beziehung:

$$Y = \boldsymbol{R} \cdot {}^{r}D + {}^{l}DY \tag{71}$$

Damit haben wir für die Korrekturmatrix $\boldsymbol{Y}$ als Lösung der Matrixgleichung $\boldsymbol{AY} = {}^{r}\boldsymbol{D}$ die zwei Fixpunktgleichungen

$$\begin{array}{llll} (72) & \boldsymbol{Y} = f(\boldsymbol{Y}) & \text{mit} & f(\boldsymbol{Y}) = {}^{l}\boldsymbol{D}(\boldsymbol{R}+\boldsymbol{Y}) \\ (73) & \boldsymbol{Y} = f(\boldsymbol{Y}) & \text{mit} & f(\boldsymbol{Y}) = \boldsymbol{R} \cdot {}^{r}\boldsymbol{D} + {}^{l}\boldsymbol{DY} \end{array}$$

gefunden, aus denen sich die beiden folgenden Iterationsgleichungen ergeben:

$$\begin{array}{llll} (74) & [\boldsymbol{Y}]^{(\nu+1)} := {}^{l}\boldsymbol{D}(\boldsymbol{R}+[\boldsymbol{Y}]^{(\nu)}), & & {}^{l}\boldsymbol{D} := I - \boldsymbol{RA}, \\ (75) & [\boldsymbol{Y}]^{(\nu+1)} := \boldsymbol{R} \cdot {}^{r}\boldsymbol{D} + {}^{l}\boldsymbol{D}[\boldsymbol{Y}]^{(\nu)}, & & \nu = 0,1,2\ldots \end{array}$$

Vergleicht man (75) mit (65), so erkennt man die völlige Identität der beiden Iterationsgleichungen, so daß das Ergebnis (69) auf die Korrekturmatrix $\boldsymbol{Y}$ direkt übertragbar ist:

$$[\boldsymbol{Y}]_{\diamond}^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]^{(k)} \implies \boldsymbol{Y} \in [\boldsymbol{Y}]_{\diamond}^{(k+1)} \text{ ist eindeutig.} \tag{76}$$

Jetzt bleibt nur noch zu klären, wie die Intervallmatrix $[\boldsymbol{Y}]_{\diamond}^{(k+1)}$ auf einem Rechner auszuwerten ist. Beachten Sie bitte dazu, daß die Funktionen $f(\boldsymbol{Y})$ in (72),(73) wirklich identisch sind und daher die Iterierten $[\boldsymbol{Y}]^{(\nu+1)}$ in (74) und (75) jeweils die **gleichen** Intervallmatrizen liefern! Prinzipiell ist es daher zunächst gleichgültig, ob die bei der Ergebnisverifikation auf dem Rechner auszuwertende Größe $[\boldsymbol{Y}]_{\diamond}^{(\nu+1)}$ mit Hilfe der rechten Seite von (74) oder (75) berechnet wird. In der Praxis entscheidet man sich natürlich für die rechte

Seite in (74), da hier eine Matrixmultiplikation eingespart wird und damit auch unnötige Intervallüberschätzungen vermieden werden. Die Intervall–Auswertung erfolgt also auf dem Rechner durch:

$$(77)\qquad [\boldsymbol{Y}]_{\diamond}^{(\nu+1)} := [{}^{l}\boldsymbol{D}] \diamond\!\!\!\cdot\, (\boldsymbol{R} \diamond\!\!\!+ [\boldsymbol{Y}]_{\diamond}^{(\nu)}), \qquad [{}^{l}\boldsymbol{D}] := \diamond(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}),$$

wobei der Startwert gegeben ist durch:

$$(78)\qquad [\boldsymbol{Y}]_{\diamond}^{(0)} := [{}^{l}\boldsymbol{D}] \diamond\!\!\!\cdot\, \boldsymbol{R}.$$

Natürlich muß ein solcher Startwert die gesuchte Lösung $\boldsymbol{Y}$ nicht enthalten, er sollte dieser Matrix jedoch möglichst nahekommen, um eine lange Iteration zu vermeiden. Geht man von der exakten Beziehung $\boldsymbol{Y} = {}^{l}\boldsymbol{D} \cdot \boldsymbol{A}^{-1}$ aus, so erhält man daraus mit $\boldsymbol{A}^{-1} \approx \boldsymbol{R}$ für $[\boldsymbol{Y}]_{\diamond}^{(0)}$ die obige Näherung, die um so besser ausfällt, je besser die Approximation $\boldsymbol{A}^{-1} \approx \boldsymbol{R}$ berechnet wurde.[5]

Wird nun mit einer Intervallmatrix $[\boldsymbol{Y}]_{\diamond}^{(k+1)}$ die Verifikationsvoraussetzung in (76) erfüllt, so gilt $\boldsymbol{Y} \in [\boldsymbol{Y}]_{\diamond}^{(k+1)}$, und nach (57) wird die Inverse $\boldsymbol{A}^{-1}$ eingeschlossen durch:

$$(79)\qquad \boldsymbol{A}^{-1} \in \boldsymbol{R} \diamond\!\!\!+ [\boldsymbol{Y}]_{\diamond}^{(k+1)}$$

In der Praxis kann der Fall eintreten, daß die Bedingung in (76) entweder gar nicht oder nur für große k-Werte erfüllt wird, weil die jeweils neu berechneten $[\boldsymbol{Y}]_{\diamond}^{(\nu+1)}$ der gesuchten Lösung $\boldsymbol{Y}$ zwar immer näher kommen, diese aber nicht wirklich einfangen können. Dies läßt sich i.a. jedoch vorzeitig erzwingen, wenn man vor jedem Iterationsschritt die alte Iterierte $[\boldsymbol{Y}]_{\diamond}^{(\nu)}$ mit Hilfe der blow–Funktion ein wenig aufbläht. Nähere Einzelheiten findet man dazu z.B. in [3, Abschnitt 3.8].

Die bisherige Diskussion der Fixpunktiteration zur Einschließung der Inversen bezog sich nach (70) auf die Defektdefinition ${}^{l}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}$. Im Abschnitt *Konvergenzbetrachtungen* auf Seite 53 hatten wir gefunden, daß der Erfolg einer Defektiteration zur Berechnung einer möglichst maximalgenauen Punktnäherung ganz wesentlich von der in (2) getroffenen Wahl der Defektdefinition abhängen kann. Um den Einfluß der beiden verschiedenen Defektdefinitionen auch bei der jetzt zu behandelnden Einschließung der Inversen untersuchen zu können, werden zunächst die entsprechenden Iterationsgleichungen für die Defektdefinition ${}^{r}\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{A}\boldsymbol{R}$ angegeben. Mit dem Ansatz

[5] Die Approximation $\boldsymbol{A}^{-1} \approx \boldsymbol{R}$ erfolgt i.a. maximalgenau.

$\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ ergibt sich dann ganz analog die Fixpunktgleichung zu:

(80) $$\boldsymbol{Y} = g(\boldsymbol{Y}) \quad \text{mit} \quad g(\boldsymbol{Y}) = (\boldsymbol{R} + \boldsymbol{Y}) \cdot {}^r\boldsymbol{D} \qquad \leadsto$$

(81) $$[\boldsymbol{Y}]_\diamond^{(0)} := \boldsymbol{R} \diamond [{}^r\boldsymbol{D}]$$

(82) $$[\boldsymbol{Y}]_\diamond^{(\nu+1)} := (\boldsymbol{R} \diamond [\boldsymbol{Y}]_\diamond^{(\nu)}) \diamond [{}^r\boldsymbol{D}]; \qquad \nu = 0, 1, 2, \ldots$$

Nachdem wir bzgl. der beiden verschiedenen Defektdefinitionen ${}^l\boldsymbol{D}$ bzw. ${}^r\boldsymbol{D}$ mit (77) und (82) die entsprechenden Iterationsgleichungen zur Einschließung der Korrektur $\boldsymbol{Y}$ angegeben haben, untersuchen wir jetzt die folgende Frage:

> Zu einer gegebenen Matrix $\boldsymbol{A}$ sei bei einfacher Rechnung mit Hilfe der Prozedur INV(A,R,factor,Err) von Seite 51 durch $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ eine i.a. maximalgenaue Näherungsinverse erfolgreich berechnet (Err=0), wobei mit factor=0 bzw. factor=1 die dabei benutzte Defektdefinition ${}^l\boldsymbol{D}$ bzw. ${}^r\boldsymbol{D}$ angegeben wird.
> **Muß man die gleiche durch factor bestimmte Defektdefinition benutzen, um danach auch die Korrektur $\boldsymbol{Y}$ erfolgreich einschließen zu können?**

Da man diese Frage vermutlich nicht rein theoretisch klären kann, wird hier eine Antwort gegeben, die nur durch die numerische Erfahrung begründet wird:

> *Nach Aufruf der Prozedur INV(A,R,factor,Err) erhält man für die aus dem Ansatz $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ bestimmte Korrektur $\boldsymbol{Y}$ und damit auch für die gesuchte Inverse $\boldsymbol{A}^{-1}$ mit der durch factor vorgegebenen Defektdefinition in fast allen Fällen eine bessere Einschließung als mit Hilfe der noch verbleibenden anderen Defektdefinition.*

Die folgenden Beispiele sollen den obiger Merksatz bestätigen. In den dazu angegebenen Tabellen findet man zum Vergleich die Einschließungen der inversen Elemente $\boldsymbol{A}^{-1}[i,j]$, die mit Hilfe der beiden möglichen Defektdefinitionen nach (77),(78) bzw. (81),(82) berechnet wurden.

Beachten Sie bitte, daß nach erfolgreicher Durchführung des jeweiligen Verifikationsschrittes $[\boldsymbol{Y}]_\diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]^{(k)}$ die nach (77) und (82) berechneten Intervallmatrizen $[\boldsymbol{Y}]_\diamond^{(k+1)}$ die gesuchte Korrekturmatrix $\boldsymbol{Y}$ **garantiert** einschließen! Wir untersuchen mit den folgenden Beispielen lediglich, welche Defektdefinition die bessere Einschließung liefert.

1. Beispiel: Wir betrachten die schon auf Seite 47 definierte Matrix

$$
\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-20} & 1 \\ 1 & 1+10^{-20} & 1 & 1 \\ 1+10^{-20} & 1 & 1 & 1 \\ 1 & 1 & 10^{-40} & 1 \end{pmatrix}
$$

INV(A,R,factor,Err); danach Berechnung einer Einschließung von		
$\boldsymbol{Y}$ bzw. $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ nach (77),(78) bzw. (81),(82). Die jeweils erste Einschließung		
wird mit der durch INV(A,R,factor,Err) festgelegten Defektdefinition ${}^r\boldsymbol{D}$ berechnet.		
$\boldsymbol{A}^{-1}[1,2] \in$	$[+0.00000000000000000000 \cdot 10^{+00}, +0.00000000000000000000 \cdot 10^{+00}]$	${}^r\boldsymbol{D}$
	$[-1.10000000000000000002 \cdot 10^{-20}, +1.10000000000000000002 \cdot 10^{-20}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[1,3] \in$	$[+1.00000000000000000000 \cdot 10^{+20}, +1.00000000000000000000 \cdot 10^{+20}]$	${}^r\boldsymbol{D}$
	$[+9.99999999999999999999 \cdot 10^{+19}, +1.00000000000000000001 \cdot 10^{+20}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[2,2] \in$	$[+1.00000000000000000000 \cdot 10^{+20}, +1.00000000000000000000 \cdot 10^{+20}]$	${}^r\boldsymbol{D}$
	$[+9.99999999999999999999 \cdot 10^{+19}, +1.00000000000000000001 \cdot 10^{+20}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[2,3] \in$	$[+0.00000000000000000000 \cdot 10^{+00}, +0.00000000000000000000 \cdot 10^{+00}]$	${}^r\boldsymbol{D}$
	$[-1.10000000000000000002 \cdot 10^{-20}, +1.10000000000000000002 \cdot 10^{-20}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,2] \in$	$[+0.00000000000000000000 \cdot 10^{+00}, +0.00000000000000000000 \cdot 10^{+00}]$	${}^r\boldsymbol{D}$
	$[-4.79999999999999999998 \cdot 10^{-335}, +4.79999999999999999998 \cdot 10^{-335}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,3] \in$	$[+0.00000000000000000000 \cdot 10^{+00}, +0.00000000000000000000 \cdot 10^{+00}]$	${}^r\boldsymbol{D}$
	$[-4.79999999999999999998 \cdot 10^{-335}, +4.79999999999999999998 \cdot 10^{-335}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,2] \in$	$[-1.00000000000000000000 \cdot 10^{+020}, -1.00000000000000000000 \cdot 10^{+020}]$	${}^r\boldsymbol{D}$
	$[-1.00000000000000000001 \cdot 10^{+20}, -9.99999999999999999999 \cdot 10^{+19}]$	${}^l\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,3] \in$	$[-1.00000000000000000000 \cdot 10^{+20}, -1.00000000000000000000 \cdot 10^{+20}]$	${}^r\boldsymbol{D}$
	$[-1.00000000000000000001 \cdot 10^{+20}, -9.99999999999999999999 \cdot 10^{+19}]$	${}^l\boldsymbol{D}$

Bemerkungen:

- Bezüglich aller restlichen Elemente von $\boldsymbol{A}^{-1}$ liefern die Defekte ${}^r\boldsymbol{D}$ und ${}^l\boldsymbol{D}$ jeweils maximalgenaue Einschließungen. Damit erhält man bei diesem Beispiel für die $\boldsymbol{A}^{-1}[i,j]$ stets die besseren Einschließungen, wenn man die mit factor=1 bestimmte Defektdefinition ${}^r\boldsymbol{D}$ benutzt.

- Aus obiger Tabelle ergeben sich mit dem Defekt ${}^r\boldsymbol{D}$ die folgenden Ergebnisse: $\boldsymbol{A}^{-1}[2,3] = \boldsymbol{A}^{-1}[3,2] = \boldsymbol{A}^{-1}[3,3] = 0$. Für diese Elemente erhält man mit dem anderen Defekt ${}^l\boldsymbol{D}$ Einschließungen, welche den exakten Wert Null als Mittelpunkt enthalten. Daß dies jedoch nicht immer eintreten muß, zeigt das 3. Beispiel auf Seite 91.

2. Beispiel: Wir betrachten die schon auf Seite 30 definierte Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1+10^{-19} & 1 \\ 1 & 1+10^{-19} & 1 & 10^{-40} \\ 1+10^{-19} & 1 & 1 & 10^{-40} \\ 1 & 1 & 1 & 1 \end{pmatrix}$$

INV(A,R,factor,Err); danach Berechnung einer Einschließung von
$\boldsymbol{Y}$ bzw. $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ nach (77),(78) bzw. (81),(82). Die jeweils erste Einschließung
wird mit der durch INV(A,R,factor,Err) festgelegten Defektdefinition $^l\boldsymbol{D}$ berechnet.

$\boldsymbol{A}^{-1}[3,1] \in$	$[+1.00000000000000000000 \cdot 10^{+19}, +1.00000000000000000000 \cdot 10^{+19}]$	$^l\boldsymbol{D}$
	$[+9.99999999999999999999 \cdot 10^{+18}, +1.00000000000000000001 \cdot 10^{+19}]$	$^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,2] \in$	$[+0.00000000000000000000 \cdot 10^{+00}, +0.00000000000000000000 \cdot 10^{+00}]$	$^l\boldsymbol{D}$
	$[-2.75000000000000000017 \cdot 10^{-21}, +2.75000000000000000017 \cdot 10^{-21}]$	$^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,3] \in$	$[+0.00000000000000000000 \cdot 10^{+00}, +0.00000000000000000000 \cdot 10^{+00}]$	$^l\boldsymbol{D}$
	$[-2.75000000000000000017 \cdot 10^{-21}, +2.75000000000000000017 \cdot 10^{-21}]$	$^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,4] \in$	$[-1.00000000000000000000 \cdot 10^{+19}, -1.00000000000000000000 \cdot 10^{+19}]$	$^l\boldsymbol{D}$
	$[-1.00000000000000000001 \cdot 10^{+19}, -9.99999999999999999999 \cdot 10^{+18}]$	$^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,1] \in$	$[-4.99999999999999999976 \cdot 10^{-1}, -4.99999999999999999974 \cdot 10^{-1}]$	$^l\boldsymbol{D}$
	$[-4.99999999999999999976 \cdot 10^{-1}, -4.99999999999999999975 \cdot 10^{-1}]$	$^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,2] \in$	$[-4.99999999999999999976 \cdot 10^{-1}, -4.99999999999999999974 \cdot 10^{-1}]$	$^l\boldsymbol{D}$
	$[-4.99999999999999999976 \cdot 10^{-1}, -4.99999999999999999975 \cdot 10^{-1}]$	$^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,3] \in$	$[-4.99999999999999999976 \cdot 10^{-1}, -4.99999999999999999974 \cdot 10^{-1}]$	$^l\boldsymbol{D}$
	$[-4.99999999999999999976 \cdot 10^{-1}, -4.99999999999999999975 \cdot 10^{-1}]$	$^r\boldsymbol{D}$

Bemerkung: Bezüglich aller restlichen Elemente von $\boldsymbol{A}^{-1}$ liefern die Defektdefinitionen $^l\boldsymbol{D}$ und $^r\boldsymbol{D}$ jeweils die gleichen, maximalgenauen Einschließungen. Bei diesem Beispiel erhält man für die $\boldsymbol{A}^{-1}[3,j]$ mit $^l\boldsymbol{D}$ stets die besseren Einschließungen, während für die drei Elemente $\boldsymbol{A}^{-1}[4,1..3]$ mit $^r\boldsymbol{D}$ sogar die maximalgenauen Einschließungen berechnet werden. Es kann also durchaus vorkommen, daß man mit nur einer der beiden möglichen Defektdefinitionen nicht alle Einschließungen jeweils optimal bestimmt. Die folgenden Beispiele werden jedoch zeigen, daß man besonders dann recht grobe Einschließungen erhalten kann, wenn man an Stelle der durch die Prozedur INV(A,R,factor,Err) mit factor vorgegebenen Defektdefinition zur Einschließung von $\boldsymbol{A}^{-1}$ den alternativen Defekt wählt.

3. Beispiel: Wir betrachten die schon auf Seite 51 definierte Matrix

$$A := \begin{pmatrix} 10^{-10} & 10^{-20} & 1 & -10^{-20} \\ 1 & 3+10^{-20} & 3 & 4 \\ 1 & 3 & 2 & 4 \\ 10^{-10} & 0.1 & 1 & -10^{-20}\cdot(1+10^{-20}) \end{pmatrix}$$

<table>
<tr><td colspan="3">INV(A,R,factor,Err); danach Berechnung einer Einschließung von</td></tr>
<tr><td colspan="3">$\boldsymbol{Y}$ bzw. $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ nach (77),(78) bzw. (81),(82). Die jeweils erste Einschließung</td></tr>
<tr><td colspan="3">wird mit der durch INV(A,R,factor,Err) festgelegten Defektdefinition ${}^{l}\boldsymbol{D}$ berechnet.</td></tr>
<tr><td rowspan="2">$\boldsymbol{A}^{-1}[1,4] \in$</td><td>$[-7.49999999981250000072 \cdot 10^{-10}, -7.49999999981250000068 \cdot 10^{-10}]$</td><td>${}^{l}\boldsymbol{D}$</td></tr>
<tr><td>$[-7.49999999981250000071 \cdot 10^{-10}, -7.49999999981250000070 \cdot 10^{-10}]$</td><td>${}^{r}\boldsymbol{D}$</td></tr>
<tr><td rowspan="2">$\boldsymbol{A}^{-1}[2,2] \in$</td><td>$[+2.49999999943750000026 \cdot 10^{-30}, +2.49999999943750000027 \cdot 10^{-30}]$</td><td>${}^{l}\boldsymbol{D}$</td></tr>
<tr><td>$[+2.49999999939000000025 \cdot 10^{-30}, +2.49999999948500000028 \cdot 10^{-30}]$</td><td>${}^{r}\boldsymbol{D}$</td></tr>
<tr><td rowspan="2">$\boldsymbol{A}^{-1}[2,3] \in$</td><td>$[-2.49999999918750000028 \cdot 10^{-30}, -2.49999999918750000026 \cdot 10^{-30}]$</td><td>${}^{l}\boldsymbol{D}$</td></tr>
<tr><td>$[-2.49999999921000000026 \cdot 10^{-30}, -2.49999999916500000028 \cdot 10^{-30}]$</td><td>${}^{r}\boldsymbol{D}$</td></tr>
<tr><td rowspan="2">$\boldsymbol{A}^{-1}[3,1] \in$</td><td>$[+1.00000000000000000010 \cdot 10^{-19}, +1.00000000000000000011 \cdot 10^{-19}]$</td><td>${}^{l}\boldsymbol{D}$</td></tr>
<tr><td>$[+1.00000000000000000007 \cdot 10^{+19}, +1.00000000000000000041 \cdot 10^{-19}]$</td><td>${}^{r}\boldsymbol{D}$</td></tr>
<tr><td rowspan="2">$\boldsymbol{A}^{-1}[3,2] \in$</td><td>$[+9.99999999999999999999 \cdot 10^{-1}, +1.00000000000000000000 \cdot 10^{+0}]$</td><td>${}^{l}\boldsymbol{D}$</td></tr>
<tr><td>$[+9.99999999999999999999 \cdot 10^{-1}, +1.00000000000000000001 \cdot 10^{+0}]$</td><td>${}^{r}\boldsymbol{D}$</td></tr>
<tr><td rowspan="2">$\boldsymbol{A}^{-1}[3,4] \in$</td><td>$[-1.00000000000000000011 \cdot 10^{-19}, -1.00000000000000000010 \cdot 10^{-19}]$</td><td>${}^{l}\boldsymbol{D}$</td></tr>
<tr><td>$[-1.00000000000000000011 \cdot 10^{-19}, -1.00000000000000000009 \cdot 10^{-19}]$</td><td>${}^{r}\boldsymbol{D}$</td></tr>
</table>

Bemerkung: Bezüglich aller restlichen Elemente von $\boldsymbol{A}^{-1}$ liefern die Defektdefinitionen ${}^{l}\boldsymbol{D}$ und ${}^{r}\boldsymbol{D}$ jeweils die gleichen, maximalgenauen Einschließungen. Bei diesem Beispiel erhält man für $\boldsymbol{A}^{-1}[1,4]$ mit der alternativen Defektdefinition ${}^{r}\boldsymbol{D}$ zwar die maximalgenaue Einschließung, allerdings werden die Einschließungen der Elemente $\boldsymbol{A}^{-1}[2,2], \boldsymbol{A}^{-1}[2,3]$ jetzt nur noch auf zehn Dezimalziffern genau berechnet. Mit der durch factor=0 gegebenen Defektdefinition ${}^{l}\boldsymbol{D}$ erhält man also für die $\boldsymbol{A}^{-1}[i,j]$, unserem letzten Merksatz entsprechend, die deutlich besseren Einschließungen.

4. Beispiel: Wir betrachten die folgende zu invertierende Matrix

$$A := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & (1+10^{-10})\cdot 10^{-40} \end{pmatrix}$$

INV(A,R,factor,Err); danach Berechnung einer Einschließung von
$\boldsymbol{Y}$ bzw. $\boldsymbol{A}^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ nach (77),(78) bzw. (81),(82). Die jeweils erste Einschließung
wird mit der durch INV(A,R,factor,Err) festgelegten Defektdefinition ${}^{l}\boldsymbol{D}$ berechnet.

$\boldsymbol{A}^{-1}[1,1] \in$	$[+9.99999999999999999999 \cdot 10^{-1}, +1.00000000000000000001 \cdot 10^{+0}]$	${}^{l}\boldsymbol{D}$
	$[+1.00000000000000000000 \cdot 10^{+0}, +1.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[1,2] \in$	$[-1.00000000000000000001 \cdot 10^{+0}, -9.99999999999999999999 \cdot 10^{-1}]$	${}^{l}\boldsymbol{D}$
	$[-1.00000000000000000000 \cdot 10^{+0}, -1.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[1,4] \in$	$[+9.99999999990000000009 \cdot 10^{-42}, +9.999999999900000000010 \cdot 10^{-42}]$	${}^{l}\boldsymbol{D}$
	$[-1.33333333333333333334 \cdot 10^{-2}, +4.33333333333333333335 \cdot 10^{-2}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[2,1] \in$	$[-1.20000000076000000002 \cdot 10^{-297}, +1.20000000076000000002 \cdot 10^{-297}]$	${}^{l}\boldsymbol{D}$
	$[+0.00000000000000000000 \cdot 10^{+0}, +0.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[2,2] \in$	$[+9.99999999999999999999 \cdot 10^{-1}, +1.00000000000000000001 \cdot 10^{+0}]$	${}^{l}\boldsymbol{D}$
	$[+1.00000000000000000000 \cdot 10^{+0}, +1.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[2,4] \in$	$[+3.33333333330000000002 \cdot 10^{+18}, +3.333333333300000000005 \cdot 10^{+18}]$	${}^{l}\boldsymbol{D}$
	$[+3.33333333329999999999 \cdot 10^{+18}, +3.333333333300000000005 \cdot 10^{+18}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,1] \in$	$[-1.20000000036000000038 \cdot 10^{-297}, +1.20000000036000000038 \cdot 10^{-297}]$	${}^{l}\boldsymbol{D}$
	$[+0.00000000000000000000 \cdot 10^{+0}, +0.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[3,2] \in$	$[-1.20000000036000000038 \cdot 10^{-297}, +1.20000000036000000038 \cdot 10^{-297}]$	${}^{l}\boldsymbol{D}$
	$[+0.00000000000000000000 \cdot 10^{+0}, +0.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,1] \in$	$[-1.20000000000000000001 \cdot 10^{-315}, +1.20000000000000000001 \cdot 10^{-315}]$	${}^{l}\boldsymbol{D}$
	$[+0.00000000000000000000 \cdot 10^{+0}, +0.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,2] \in$	$[-1.20000000000000000001 \cdot 10^{-315}, +1.20000000000000000001 \cdot 10^{-315}]$	${}^{l}\boldsymbol{D}$
	$[+0.00000000000000000000 \cdot 10^{+0}, +0.00000000000000000000 \cdot 10^{+0}]$	${}^{r}\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,3] \in$	$[-1.20000000000000000001 \cdot 10^{-315}, +1.20000000000000000001 \cdot 10^{-315}]$	${}^{l}\boldsymbol{D}$
	$[-3.33333333333333333335 \cdot 10^{-277}, +3.33333333333333333335 \cdot 10^{-277}]$	${}^{r}\boldsymbol{D}$

Bemerkung: Bezüglich aller restlichen Elemente von $\boldsymbol{A}^{-1}$ liefern die Defektdefinitionen ${}^{l}\boldsymbol{D}$ und ${}^{r}\boldsymbol{D}$ jeweils die gleichen und mindestens hochgenauen Einschließungen. Bei diesem Beispiel erhält man mit der alternativen Defektdefinition ${}^{r}\boldsymbol{D}$ bei einigen Elementen zwar eine geringfügige Verbesserung der Einschließung, allerdings wird das Elemente $\boldsymbol{A}^{-1}[1,4]$ durch das Intervall $[-1.33\ldots \cdot 10^{-2}, +4.33\ldots \cdot 10^{-2}]$ so grob eingeschlossen, daß noch nicht einmal sein Vorzeichen festgelegt wird! Andererseits liefert die durch factor=0 vorgegebene Defektdefinition ${}^{l}\boldsymbol{D}$ für das gleiche Element $\boldsymbol{A}^{-1}[1,4]$ sogar die maximalgenaue Einschließung, so daß auch jetzt der Merksatz von Seite 78 bestätigt wird.

5. Beispiel: Wir betrachten abschließend die folgende zu invertierende Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & (1+10^{-10})\cdot 10^{-40} \end{pmatrix}$$

INV(A,R,factor,Err); danach Berechnung einer Einschließung von $\boldsymbol{Y}$ bzw. $\boldsymbol{A}^{-1} = \boldsymbol{R}+\boldsymbol{Y}$ nach (77),(78) bzw. (81),(82). Die jeweils erste Einschließung wird mit der durch INV(A,R,factor,Err) festgelegten Defektdefinition ${}^l\boldsymbol{D}$ berechnet.		
$\boldsymbol{A}^{-1}[1,4] \in$	$[-2.50000000000000000000 \cdot 10^{-3}, +1.74999999999000000001 \cdot 10^{-2}]$	${}^l\boldsymbol{D}$
	$[-2.50000000000000000001 \cdot 10^{-3}, +1.08333333333333333334 \cdot 10^{-2}]$	${}^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[2,4] \in$	$[+3.33333333300000000003 \cdot 10^{+18}, +3.33333333300000000004 \cdot 10^{+18}]$	${}^l\boldsymbol{D}$
	$[+3.33333333300000000003 \cdot 10^{+18}, +3.33333333300000000005 \cdot 10^{+18}]$	${}^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,1] \in$	$[-1.20000000000000000001 \cdot 10^{-315}, +1.20000000000000000001 \cdot 10^{-315}]$	${}^l\boldsymbol{D}$
	$[-7.50000000000000000003 \cdot 10^{-299}, +7.50000000000000000003 \cdot 10^{-299}]$	${}^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,2] \in$	$[-1.20000000000000000001 \cdot 10^{-315}, +1.20000000000000000001 \cdot 10^{-315}]$	${}^l\boldsymbol{D}$
	$[-7.50000000000000000003 \cdot 10^{-299}, +7.50000000000000000003 \cdot 10^{-299}]$	${}^r\boldsymbol{D}$
$\boldsymbol{A}^{-1}[4,3] \in$	$[-1.20000000000000000001 \cdot 10^{-315}, +1.20000000000000000001 \cdot 10^{-315}]$	${}^l\boldsymbol{D}$
	$[-7.50000000000000000003 \cdot 10^{-299}, +7.50000000000000000003 \cdot 10^{-299}]$	${}^r\boldsymbol{D}$

Bemerkung: Bezüglich aller restlichen Elemente von $\boldsymbol{A}^{-1}$ liefern die Defektdefinitionen ${}^l\boldsymbol{D}$ und ${}^r\boldsymbol{D}$ jeweils die gleichen maximalgenauen Einschließungen. Bei diesem Beispiel erhält man jedoch für $\boldsymbol{A}^{-1}[1,4]$ sowohl mit der alternativen Defektdefinition ${}^r\boldsymbol{D}$ als auch mit der durch factor=0 vorgegebenen Defektdefinition ${}^l\boldsymbol{D}$ nur eine so grobe Einschließung, daß noch nicht einmal das korrekte Vorzeichen von $\boldsymbol{A}^{-1}[1,4]$ festgelegt wird! Auf Seite 92 werden wir mit Hilfe einer *staggered arithmetic* in der Lage sein, auch für dieses Element die maximalgenaue Einschließung

$$\boldsymbol{A}^{-1}[1,4] \in [7.49999999925000000007 \cdot 10^{-42}, 7.49999999925000000008 \cdot 10^{-42}]$$

mit etwa 100 internen Dezimalstellen zu berechnen. Es fällt zusätzlich auf, daß die Elemente $\boldsymbol{A}^{-1}[4,1..3]$ durch Intervalle mit dem Mittelpunkt 0 eingeschlossen werden. Dies läßt vermuten, daß die exakten Werte dieser Elemente tatsächlich durch die Zahl Null gegeben sind. Auf der Seite 87 werden wir sehen, daß man die jetzt vorliegenden Einschließungen mit dem Verfahren von Alefeld/Herzberger noch weiter verbessern kann.

Zusammenfassung: Die letzten fünf Beipsiele bestätigen den Merksatz von Seite 78, nach dem man i.a. die besseren Einschließungen der inversen Elemente $\boldsymbol{A}^{-1}[i,j]$ erhält, wenn man zur Einschließung des Fixpunktes $\boldsymbol{Y}^*$ der Gleichungen (72) oder (80) diejenige Defektdefinition wählt, die nach dem Aufruf der Prozedur INV(A,R,factor,Err) durch den Parameter factor festgelegt wird.

Bei allen bisherigen Beispielen haben wir den Verifikationsschritt, d.h. die Einschließung des Fixpunktes $\boldsymbol{Y}^*$ mit Hilfe der durch den Parameter factor vorgegebenen Defektdefinition erfolgreich durchführen können, was natürlich nicht bedeutet, daß dies immer möglich sein muß! Um jedoch einem solchen Ausnahmefall [6] vorbeugen zu können, benutzen wir den **Algorithmus**:

1. Aufruf von INV(A,R,factor,Err) zur Berechnung einer i.a. maximalgenauen Näherungsinversen $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$. Der Parameter factor gibt die dabei benutzte Defektdefinition an, vergl. Seite 51.

2. Einschließung der Korrekturmatrix $\boldsymbol{Y}$ nach (77),(78) bzw. (81),(82) entsprechend der durch factor vorgegebenen Defektdefinition.

3. Falls die Einschließung von $\boldsymbol{Y}$ scheitert, wird der Verifikationsschritt in 2. mit Hilfe der alternativen Defektdefinition durchgeführt.

Genau dieser Algorithmus wird realisiert beim Aufruf der **PASCAL-XSC** Prozedur

```
INV( var A : Lrmatrix; illcd,Alefeld : integer;
     var X : Limatrix; var Err : integer);

exportiert durch linv
```

Im Falle Err=0 liefert $\boldsymbol{X}$ eine garantierte Einschließung der Inversen $\boldsymbol{A}^{-1}$ einer gegebenen, reellen, quadratischen Eingangsmatrix $\boldsymbol{A}$.

illcd=0 Rechnung in einfacher Länge des Longreal–Formats

illcd=1 Rechnung in doppelter Länge des Longreal–Formats, vergl. Seite 85

illcd=2 Rechnung in *staggered arithmetic*, vergl. Seite 92

[6] Dem Autor ist kein solches Beispiel bekannt.

Der Eingangsparameter Alefeld=0,1,2... bestimmt die gewünschte Anzahl der Alefeld–Iterationen, mit denen die berechnete Einschließung i.a. noch weiter verbessert werden kann, vergeichen Sie bitte dazu den entsprechenden Abschnitt auf Seite 87.

In den letzten fünf Beispielen wurden die in den jeweils ersten Zeilen der Tabellen angegebenen Einschließungen der $\boldsymbol{A}^{-1}[i,j]$ durch Aufruf der Prozedur INV(A,illcd,Alefeld,X,Err) mit den Parameterwerten illcd=0 und Alefeld=0 berechnet.

Bei doppeltlanger Rechnung (illcd=1) wird zunächst wieder durch Aufruf der Prozedur INV(A,R1,R2,factor,Err) mit $\boldsymbol{R}_1+\boldsymbol{R}_2 \approx \boldsymbol{A}^{-1}$ eine doppeltlange Näherungsinverse berechnet, wobei der Parameter factor die dabei benutzte Defektdefinition liefert, vergl. Seite 63. Falls diese Defektdefinition z.B. durch ${}^l\boldsymbol{D}$ gegeben ist, lautet die Iterationsgleichung zur Einschließung von $\boldsymbol{Y}$ analog zu (77),(78) jetzt wie folgt[7]:

$$
\begin{aligned}
[{}^l\boldsymbol{D}] &:= \ \Diamond(\boldsymbol{I} - (\boldsymbol{R}_1 + \boldsymbol{R}_2)\boldsymbol{A}) \\
[\boldsymbol{Y}]_\Diamond^{(0)} &:= [{}^l\boldsymbol{D}] \mathbin{\Diamond\!\!\!\cdot}\ (\boldsymbol{R}_1 \oplus \boldsymbol{R}_2), \quad \text{prinzipiell beliebiger Startwert} \\
[\boldsymbol{Y}]_\Diamond^{(\nu+1)} &:= [{}^l\boldsymbol{D}] \mathbin{\Diamond\!\!\!\cdot}\ \Diamond(\boldsymbol{R}_1 + \boldsymbol{R}_2 + [\boldsymbol{Y}]_\Diamond^{(\nu)}); \quad \nu = 0,1,2,\ldots
\end{aligned}
$$

Dabei bedeutet $\Diamond(...)$ die maximalgenaue Einschließung aller Elemente des geklammerten, exakten Matrixausdrucks in das benutzte Longreal–Format. Ist der Defekt durch ${}^r\boldsymbol{D}$ gegeben, so sind die Gleichungen (81),(82) analog umzuschreiben, was dem Leser überlassen bleibt. Die beschriebene Vorgehensweise wird realisiert beim Aufruf der Prozedur INV(A,illcd,Alefeld,X,Err), wenn der Eingangsparameter illcd=1 gewählt wird.

6. Beispiel: Zur Rechnung im doppeltlangen Longreal–Format betrachten wir die zu invertierende Matrix

$$
\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-43} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 0 & 1 & 0 & (1+10^{-20})\cdot 10^{-40} \end{pmatrix}
$$

Beim Aufruf der Prozedur INV(A,illcd,Alefeld,X,Err) erhält man mit illcd=0 den Fehlerkode Err=4, d.h. die Einschließung $\boldsymbol{X}$ kann im einfachen Datenformat nicht berechnet werden. Wählt man jedoch mit illcd=1 die doppeltlange Rechnung im Longreal–Format, so liefert $\boldsymbol{X}$ sogar die maximalgenaue

[7]Der Startwert $[\boldsymbol{Y}]^{(0)}$ kann aber muß den Fixpunkt $\boldsymbol{Y}^*$ natürlich nicht einschließen. Daher ist der fehlerbehaftete Operator $\oplus$ im Ausdruck für $[\boldsymbol{Y}]^{(0)}$ zulässig!

Einschließung der Inversen. Die Ergebnisse sind in folgender Tabelle zusammengestellt:

INV(A,illcd,Alefeld,X,Err)	
illcd=1, d.h. doppeltlange Rechnung; Alefeld=0; $\boldsymbol{A}^{-1}[i,j] \in \boldsymbol{X}[i,j]$	
$\boldsymbol{A}^{-1}[1,1] \in [+7.50000000000000000000 \cdot 10^{-1}, +7.50000000000000000001 \cdot 10^{-1}]$	$= \boldsymbol{X}^{-1}[1,1]$
$\boldsymbol{A}^{-1}[1,2] \in [-7.50000000000000000001 \cdot 10^{-1}, -7.50000000000000000000 \cdot 10^{-1}]$	$= \boldsymbol{X}^{-1}[1,2]$
$\boldsymbol{A}^{-1}[1,3] \in [+2.49999999999999999999 \cdot 10^{-1}, +2.50000000000000000000 \cdot 10^{-1}]$	$= \boldsymbol{X}^{-1}[1,3]$
$\boldsymbol{A}^{-1}[1,4] \in [+2.24999999999999999932 \cdot 10^{-22}, +2.24999999999999999933 \cdot 10^{-22}]$	$= \boldsymbol{X}^{-1}[1,4]$
$\boldsymbol{A}^{-1}[2,1] \in [+7.49999999999999999782 \cdot 10^{-20}, +7.49999999999999999783 \cdot 10^{-20}]$	$= \boldsymbol{X}^{-1}[2,1]$
$\boldsymbol{A}^{-1}[2,2] \in [+2.24999999999999999934 \cdot 10^{-19}, +2.24999999999999999935 \cdot 10^{-19}]$	$= \boldsymbol{X}^{-1}[2,2]$
$\boldsymbol{A}^{-1}[2,3] \in [-7.49999999999999999783 \cdot 10^{-20}, -7.49999999999999999782 \cdot 10^{-20}]$	$= \boldsymbol{X}^{-1}[2,3]$
$\boldsymbol{A}^{-1}[2,4] \in [+9.99999999999999999700 \cdot 10^{-1}, +9.99999999999999999701 \cdot 10^{-1}]$	$= \boldsymbol{X}^{-1}[2,4]$
$\boldsymbol{A}^{-1}[3,1] \in [+2.49999999999999999924 \cdot 10^{+20}, +2.49999999999999999925 \cdot 10^{+20}]$	$= \boldsymbol{X}^{-1}[3,1]$
$\boldsymbol{A}^{-1}[3,2] \in [+7.49999999999999999775 \cdot 10^{+20}, +7.49999999999999999776 \cdot 10^{+20}]$	$= \boldsymbol{X}^{-1}[3,2]$
$\boldsymbol{A}^{-1}[3,3] \in [-2.49999999999999999925 \cdot 10^{+20}, -2.49999999999999999924 \cdot 10^{+20}]$	$= \boldsymbol{X}^{-1}[3,3]$
$\boldsymbol{A}^{-1}[3,4] \in [-9.99999999999999999700 \cdot 10^{+20}, -9.99999999999999999699 \cdot 10^{+20}]$	$= \boldsymbol{X}^{-1}[3,4]$
$\boldsymbol{A}^{-1}[4,1] \in [-7.49999999999999999776 \cdot 10^{+20}, -7.49999999999999999775 \cdot 10^{+20}]$	$= \boldsymbol{X}^{-1}[4,1]$
$\boldsymbol{A}^{-1}[4,2] \in [-2.24999999999999999933 \cdot 10^{+21}, -2.24999999999999999932 \cdot 10^{+21}]$	$= \boldsymbol{X}^{-1}[4,2]$
$\boldsymbol{A}^{-1}[4,3] \in [+7.49999999999999999775 \cdot 10^{+20}, +7.49999999999999999776 \cdot 10^{+20}]$	$= \boldsymbol{X}^{-1}[4,3]$
$\boldsymbol{A}^{-1}[4,4] \in [+2.99999999999999999909 \cdot 10^{+21}, +2.99999999999999999910 \cdot 10^{+21}]$	$= \boldsymbol{X}^{-1}[4,4]$

7. Beispiel: Wir betrachten die schon auf Seite 21 definierten Boothroyd/Dekkermatrizen $\boldsymbol{D}$, deren inverse Elemente bis aufs Vorzeichen durch die $d_{i,j}$ direkt gegeben sind. Der Aufruf INV(D,illcd,Alefeld,X,Err) liefert dann mit Alefeld=0 und illcd=0 für die Ordnung $n = 2, 3, \ldots, 15$ mit $\boldsymbol{X}$ die maximalgenaue Einschließung von $\boldsymbol{D}^{-1}$. Nur mit doppeltlanger Rechnung (illcd=1) erhält man für die höheren Ordnungen $n = 16, 17, \ldots, 27$ mit $\boldsymbol{X}$ ebenfalls die maximalgenaue Einschließung der Inversen $\boldsymbol{D}^{-1} \in \boldsymbol{X}$.

Im nächsten Abschnitt wird ein von G. Alefeld und J. Herzberger [1, Seite 251] angegebenes Iterationsverfahren vorgestellt, mit dem man eine bereits berechnete Einschließung einer Inversen weiter verbessern kann. Man wird auf dieses Verfahren zurückgreifen, wenn die mit dem Fixpunktsatz von Brouwer berechnete Einschließung nicht maximalgenau berechnet werden konnte. Wir kommen damit zur Klärung des Eingangsparameters Alefeld in der Prozedur INV(A,illcd,Alefeld,X,Err) von Seite 84.

8.2 Iteration nach Alefeld und Herzberger

Zu einer gegebenen, quadratischen Matrix $\boldsymbol{A} \in I\!R^{n\times n}$ sei eine Einschließung $[\boldsymbol{X}]^{(0)} \ni \boldsymbol{A}^{-1}$ der Inversen bereits berechnet. Mit der Mittelpunktsmatrix $\boldsymbol{M}^{(k)} := \boldsymbol{M}([\boldsymbol{X}]^{(k)})$ geben G. Alefeld und J. Herzberger [1, Abschnitt 18] die folgende Iterationsgleichung an:

$$(83) \qquad [\boldsymbol{X}]^{(k+1)} := \boldsymbol{M}^{(k)} + [\boldsymbol{X}]^{(k)}(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(k)}), \quad k = 0, 1, 2 \ldots,$$

mit der man die Einschließung $[\boldsymbol{X}]^{(0)}$ i.a. weiter verbessern kann; bezüglich der Gleichung (83) gilt folgendes:

1. Unter der Voraussetzung $\boldsymbol{A}^{-1} \in [\boldsymbol{X}]^{(0)}$ enthält auch jede Iterierte $[\boldsymbol{X}]^{(k)}$ die Inverse $\boldsymbol{A}^{-1}$. Dabei muß im Beweis $\boldsymbol{M}^{(k)}$ nicht die exakte Mittelpunktsmatrix sein, sondern kann sogar beliebig gewählt werden. Es genügt daher, auf dem Rechner für $\boldsymbol{M}^{(k)}$ eine maximalgenaue Punkt–Approximation der exakten Mittelpunktsmatrix zu benutzen!

2. Die Folge der $[\boldsymbol{X}]^{(k)}$ konvergiert gegen die Inverse $\boldsymbol{A}^{-1}$ genau dann, wenn der Spektralradius von $(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(0)})$ kleiner als 1 ist.

3. Nach Punkt 1. liegt es nahe, nach jedem Iterationsschritt zunächst den Durchschnitt $[\boldsymbol{X}]^{(k+1)} := [\boldsymbol{X}]^{(k+1)} \cap [\boldsymbol{X}]^{(k)}$ mit der vorangegangenen Iterierten zu bilden und dann mit der eventuell verbesserten Einschließung $[\boldsymbol{X}]^{(k+1)}$ für $\boldsymbol{A}^{-1}$ die Iteration fortzusetzen. Bei diesem modifizierten Verfahren wird Konvergenz jedoch erst garantiert, wenn **für alle** Punktmatrizen $\boldsymbol{P} \in [\boldsymbol{X}]^{(0)}$ der Spektralradius von $|\boldsymbol{I} - \boldsymbol{A}\boldsymbol{P}|$ kleiner als 1 ist. Da die Iterationsgleichung (83) nur für eine zusätzliche Verbesserung einer schon berechneten Einschließung zur Anwendung kommt, ist die letzte Konvergenzbedingung in allen praktischen Fällen erfüllt.

Auf dem Rechner muß die Iterationsgleichung (83) intervallmäßig ausgewertet werden. Mit dem Hinweis aus Punkt 3. lauten dann die Iterationsgleichungen:

$$\begin{aligned}
(84) \qquad [\boldsymbol{X}]_\diamond^{(k+1)} &:= \boldsymbol{M}^{(k)} \mathbin{\diamond\!\!\!\!+} [\boldsymbol{X}]_\diamond^{(k)} \mathbin{\diamond\!\!\!\!\cdot} \diamond(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(k)}) \\
(85) \qquad [\boldsymbol{X}]_\diamond^{(k+1)} &:= [\boldsymbol{X}]_\diamond^{(k+1)} \cap [\boldsymbol{X}]_\diamond^{(k)}, \qquad k = 0, 1, 2 \ldots
\end{aligned}$$

Dabei sind $\mathbin{\diamond\!\!\!\!+}$, $\mathbin{\diamond\!\!\!\!\cdot}$ maximalgenaue Intervalloperatoren, und die maximalgenaue Intervallmatrix $\diamond(...)$ ist dadurch definiert, daß der Ausdruck $(...)$

komponentenweise zunächst im Akkumulator exakt zu berechnen und dann zur jeweils nächsten Rasterzahl auf- bzw. abzurunden ist. Mit dem Startwert $[\boldsymbol{X}]_\diamond^{(0)}$, der durch die Fixpunkteinschließung von Brouwer geliert wird, erfolgen dann nach (84),(85) soviele Iterationsschritte, wie durch den integer-Parameter Alefeld beim Aufruf der Prozedur INV(A,illcd,Alefeld,X,Err) von Seite 84 vorgegeben werden.

Bevor wir zu ersten numerischen Beispielen kommen, soll noch geklärt werden, warum die Fixpunkteinschließung nach Brouwer/Rump dem Verfahren von Alefeld/Herzberger vorzuziehen ist.

8.2.1 Vergleich der Iterationsverfahren

Um die Iterationen der beiden letzten Abschnitte vergleichen zu können, werden die entsprechenden Gleichungen (77) und (84) noch einmal angegeben:

$$(86) \qquad [\boldsymbol{Y}]_\diamond^{(k+1)} := [{}^l\boldsymbol{D}] \odot (\boldsymbol{R} \oplus [\boldsymbol{Y}]_\diamond^{(k)}); \qquad \text{(Brouwer)}$$

$$(87) \qquad [\boldsymbol{X}]_\diamond^{(k+1)} := \boldsymbol{M}^{(k)} \oplus [\boldsymbol{X}]_\diamond^{(k)} \odot \diamond(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(k)}) \qquad \text{(Alefeld)}$$

Da die in den Iterationsgleichungen auftretenden Intervallrechnungen größere Laufzeiten erfordern und erfahrungsgemäß zu unerwünschten Aufblähungseffekten führen, wird man von einem effektiven Algorithmus verlangen, daß Intervalloperationen und aufwendige Matrixmultiplikationen möglichst vermieden werden. Vergleicht man daraufhin die Iterationsgleichungen (86),(87), so muß man dem ersten Verfahren mit der Iterationsgleichung (86) aus folgenden Gründen den Vorzug geben:

1. Während in beiden Gleichungen bei jedem Iterationsschritt jeweils eine Multiplikation zweier Intervall–Matrizen auszuführen ist, wird in Gleichung (86) die Intervall–Matrix $[{}^l\boldsymbol{D}]$ als Einschließung des Defekts ${}^l\boldsymbol{D} := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}$ zu Beginn des Iterationsverfahrens nur einmal berechnet, während in Gleichung (87) die Intervallmatrix $\diamond(\boldsymbol{I} - \boldsymbol{A}\boldsymbol{M}^{(k)})$ bei jedem Iterationsschritt recht aufwendig und neu zu berechnen ist.

2. Das erste Einschließungsverfahren ist auch deshalb vorzuziehen, weil die notwendige Punktapproximation $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer sehr gut konvergierenden Defektiteration ohne Intervallrechnung in fast allen praktischen Fällen sehr effektiv und meist maximalgenau berechnet wird, so daß die für die Fixpunkteinschließung notwendige Beziehung $[\boldsymbol{Y}]_\diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]_\diamond^{(k)}$ meist schon nach zwei oder drei Iterationsschritten erreicht werden kann.

3. Ein weiterer Nachteil des Verfahrens von Alefeld und Herzberger besteht auch noch darin, daß die Bedingung $\boldsymbol{A}^{-1} \in [\boldsymbol{X}]_\diamond^{(0)}$ zu erfüllen ist, was nur unter einschränkenden Voraussetzungen erreicht werden kann und daher i.a. zu merklichen Aufblähungen des Startintervalls $[\boldsymbol{X}]_\diamond^{(0)}$ führt.

Die Methode von Alefeld/Herzberger hat aber z.B. dann ihre Berechtigung, wenn die mit anderen Verfahren berechnete Einschließung der Inversen nicht für alle Komponenten maximalgenau bestimmt werden konnte und zu grobe Einschließungen weiter zu verbessern sind.

8.2.2 Numerische Beispiele

1. Beispiel: Wir betrachten die Matrix $\boldsymbol{A}$ von Seite 85, die wir nur im Element $\boldsymbol{A}[1,4]$ wie folgt abändern:

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-62} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 0 & 1 & 0 & (1+10^{-20}) \cdot 10^{-40} \end{pmatrix}$$

Mit den Parameterwerten illcd=1 und Alefeld=0 liefert der Prozeduraufruf INV(A,illcd,Alefeld,X,Err) mit $\boldsymbol{X}$ bis auf $\boldsymbol{X}[1,4]$ genau die gleichen Einschließungen der inversen Elemente, wie sie in der Tabelle auf Seite 86 angegeben sind. Die Einschließungen $\boldsymbol{X}[1,4]$ von $\boldsymbol{A}^{-1}[1,4]$ ergeben sich für die Parameterwerte Alefeld=0,1 zu:

INV(A,illcd,Alefeld,X,Err)	
illcd=1, d.h. doppeltlange Rechnung; Alefeld=0,1; $\boldsymbol{A}^{-1}[1,4] \in \boldsymbol{X}[1,4]$	
$\boldsymbol{A}^{-1}[1,4] \in [2.0187499999999999999 \cdot 10^{-41}, 2.6708333333333333325 \cdot 10^{-41}]$	Alefeld=0
$\boldsymbol{A}^{-1}[1,4] \in [2.2499999999999999932 \cdot 10^{-41}, 2.2499999999999999933 \cdot 10^{-41}]$	Alefeld=1

Damit ist die mit Alefeld=0 berechnete erste recht grobe Einschließung das Ergebnis der Fixpunktiteration nach Brouwer im doppeltlangen Longreal–Format, und diese Einschließung ist dann der Startwert $[\boldsymbol{X}]_\diamond^{(0)}$ auf der rechten Seite von (84), wenn mit dem Parameterwert Alefeld=1 die Alefeld–Iteration (84),(85) genau einmal durchgeführt wird. Das Ergebnis $[\boldsymbol{X}]_\diamond^{(1)}$ der linken Seite von (85) ist dann die stark verbesserte und sogar maximalgenaue Einschließung von $\boldsymbol{A}^{-1}[1,4] \in [2.2499\ldots9932, 2.2499\ldots9933] \cdot 10^{-41}$.

2. Beispiel: Wir betrachten die schon auf Seite 83 behandelte Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & (1+10^{-10})\cdot 10^{-40} \end{pmatrix}$$

Die Inverse $\boldsymbol{A}^{-1}$ soll durch die Matrix $\boldsymbol{X}$ eingeschlossen werden, wobei $\boldsymbol{X}$ mit Hilfe der Prozedur INV(A,illcd,Alefeld,X,Err) jetzt bei doppeltlanger Rechnung (illcd=1) zu bestimmen ist. In nachfolgender Tabelle sind die zunächst für Alefeld=0 berechneten, nicht maximalgenauen Einschließungen $\boldsymbol{X}[i,j]$ angegeben. Zusätzlich findet man noch deren Nachbesserungen, die durch die Alefeld–Iteration mit den Parameterwerten Alefeld=0,1,...,6 berechnet wurden:

INV(A,illcd,Alefeld,X,Err); $\boldsymbol{A}^{-1}[i,j] \in \boldsymbol{X}[i,j]$; Alefeld=0,1,...,6;		
illcd=1, d.h. doppeltlange Rechnung bei Brouwer's Fixpunktiteration		
$\boldsymbol{A}^{-1}[i,j]$	$\boldsymbol{X}[i,j]$	Alefeld
$\boldsymbol{A}^{-1}[1,4]$	$\in [-1.44791662387604165329 \cdot 10^{-24}, +2.05208337602395834673 \cdot 10^{-24}]$	0,1,..,6
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-1.20000000000000000001 \cdot 10^{-375}, +1.20000000000000000001 \cdot 10^{-375}]$	0
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-3.00000000000000000003 \cdot 10^{-397}, +3.00000000000000000003 \cdot 10^{-397}]$	1
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-7.50000000000000000008 \cdot 10^{-419}, +7.50000000000000000008 \cdot 10^{-419}]$	2
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-1.87500000000000000002 \cdot 10^{-440}, +1.87500000000000000002 \cdot 10^{-440}]$	3
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-4.68750000000000000005 \cdot 10^{-462}, +4.68750000000000000005 \cdot 10^{-462}]$	4
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-1.17187500000000000002 \cdot 10^{-483}, +1.17187500000000000002 \cdot 10^{-483}]$	5
$\boldsymbol{A}^{-1}[4,1..3]$	$\in [-2.92968750000000000005 \cdot 10^{-505}, +2.92968750000000000005 \cdot 10^{-505}]$	6

Zunächst erkennt man, daß die recht grobe Einschließung des inversen Elements $\boldsymbol{A}^{-1}[1,4]$ auch durch mehrfache Alefeld–Iterationen nicht weiter verbessert werden konnte. Im Gegensatz dazu sind die Einschließungen der Elemente $\boldsymbol{A}^{-1}[4,1..3]$ bis Alefeld=6 deutlich verbessert worden. Dabei fällt auf, daß alle diese Einschließungsintervalle den Mittelpunkt Null besitzen, so daß man die $\boldsymbol{A}^{-1}[4,1..3] = 0$ versuchsweise durch die Punktintervalle $[0,0]$ einschließen könnte[8]. Mir ist jedoch kein einfaches Verfahren bekannt, für eine so konstruierte Intervallmatrix $\boldsymbol{X}$, die in diesem Fall ja nicht mit einem selbstverifizierenden Inversionsalgorithmus bestimmt wurde, den Beweis $\boldsymbol{A}^{-1} \in \boldsymbol{X}$ zu führen. Es genügt jedenfalls nicht zu verlangen, daß die Bedingungen

$$\boldsymbol{I} \in \boldsymbol{AX} \quad \wedge \quad \boldsymbol{I} \in \boldsymbol{XA}$$

[8]Die exakten Werte $\boldsymbol{A}^{-1}[4,1..3] = 0$ wurden mit Mathematica symbolisch berechnet.

erfüllt sind. Setzt man nämlich $\boldsymbol{P} := \boldsymbol{AX}$, so gilt nach Voraussetzung $\boldsymbol{I} \in \boldsymbol{P}$, und nach Multiplikation mit $\boldsymbol{A}^{-1}$ erhält man zunächst:

$$\boldsymbol{A}^{-1}\boldsymbol{P} = \boldsymbol{A}^{-1}(\boldsymbol{AX})$$

Wegen $\boldsymbol{I} \in \boldsymbol{P}$ folgt dann $\boldsymbol{A}^{-1} \in \boldsymbol{A}^{-1}\boldsymbol{P}$ und damit auch: $\boldsymbol{A}^{-1} \in \boldsymbol{A}^{-1}(\boldsymbol{AX})$, und der Beweis für $\boldsymbol{A}^{-1} \in \boldsymbol{X}$ wäre fertig, wenn oben rechts das Assoziativgesetz gelten würde! Nach [1, Seite 151] ist aber das Assoziativgesetz bei der Matrixmultiplikation schon dann nicht mehr erfüllt, wenn einer der drei Faktoren eine Intervallmatrix ist. Außerdem lassen sich für $\boldsymbol{X}$ Beispiele angeben, nach denen die beiden Voraussetzungen von Seite 90 erfüllt sind, ohne daß die Inverse $\boldsymbol{A}^{-1}$ in $\boldsymbol{X}$ enthalten ist!

3. Beispiel: Wir betrachten die Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 \cdot 10^{-40} & 200 & 200 \\ 1 & 10^{-40} & 30 & 30 \\ 10^{+20} & 2 \cdot 10^{+20} & 3 \cdot 10^{-20} & 4 \cdot 10^{+20} \end{pmatrix}$$

und wollen zeigen, daß hier das Element $\boldsymbol{A}^{-1}[1,4] = 0$ nicht durch Intervalle eingeschlossen wird, welche die Zahl Null als Mittelpunkt enthalten[9].

INV(A,illcd,Alefeld,X,Err); $\boldsymbol{A}^{-1}[1,4] = 0 \in \boldsymbol{X}[1,4]$; Alefeld=0,1,...,21;	
illcd=0, d.h. einfache Rechnung bei Brouwer's Fixpunktiteration	
$\boldsymbol{A}^{-1}[1,4]$ $\boldsymbol{X}[1,4]$	Alefeld
$\boldsymbol{A}^{-1}[1,4] \in [-1.23999999999999999999 \cdot 10^{-60}, +2.26000000000000000004 \cdot 10^{-60}]$	0
$\boldsymbol{A}^{-1}[1,4] \in [-2.00000000000000000000 \cdot 10^{-81}, +5.00000000000000000000 \cdot 10^{-81}]$	1
$\boldsymbol{A}^{-1}[1,4] \in [-2.00000000000000000000 \cdot 10^{-122}, +5.00000000000000000000 \cdot 10^{-122}]$	3
$\boldsymbol{A}^{-1}[1,4] \in [-2.00000000000000000000 \cdot 10^{-163}, +5.00000000000000000000 \cdot 10^{-163}]$	5
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-205}, +2.00000000000000000000 \cdot 10^{-205}]$	7
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-246}, +2.00000000000000000000 \cdot 10^{-246}]$	9
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-287}, +2.00000000000000000000 \cdot 10^{-287}]$	11
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-328}, +2.00000000000000000000 \cdot 10^{-328}]$	13
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-369}, +2.00000000000000000000 \cdot 10^{-369}]$	15
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-451}, +2.00000000000000000000 \cdot 10^{-451}]$	19
$\boldsymbol{A}^{-1}[1,4] \in [-5.00000000000000000000 \cdot 10^{-492}, +2.00000000000000000000 \cdot 10^{-492}]$	21

[9]Der exakte Wert $\boldsymbol{A}^{-1}[1,4] = 0$ wurden mit Mathematica symbolisch berechnet.

8.3 Einschließung mit staggered arithmetic

Wir haben schon mehrere Beispiele kennengelernt, bei denen die bisherigen Verfahren entweder keine oder viel zu grobe Einschließungen der Inversen $\boldsymbol{A}^{-1}$ geliefert haben. So erhält man z.B. zur Matrix $\boldsymbol{A}$ von Seite 83 für das Element $\boldsymbol{A}^{-1}[1,4]$ mit der Prozedur INV(A,illcd,Alefeld,X,Err) sowohl bei einfacher (illcd=0) als auch bei doppeltlanger Rechnung (illcd=1) nur so grobe Einschließungen, daß noch nicht einmal das richtige Vorzeichen von $\boldsymbol{A}^{-1}[1,4]$ festgelegt wird. Da auch eine nachträgliche Alefeld–Iteration keine wesentliche Verbesserung bringt, bleibt nur noch die Möglichkeit, die LU–Zerlegung von $\boldsymbol{A}$ nach dem Crout'schen Algorithmus [3, Seite 176] mit Hilfe der schon auf Seite 69 angesprochenen *staggered arithmetic* auszuwerten, wodurch man dann eine Einschließung der Inversen $\boldsymbol{A}^{-1}$ automatisch erhält. Wegen der langen Laufzeiten wird man dieses mit dem Eingangsparameter illcd=2 vorbestimmte Verfahren bei sehr schlecht konditionierten Matrizen jedoch erst dann anwenden, wenn alle anderen bisher behandelten Einschließungsmethoden versagt haben.

8.3.1 Numerische Beispiele

Die Einschließung der Inversen einer gegebenen Matrix $\boldsymbol{A} \in I\!R^{n\times n}$ wird berechnet mit Hilfe der schon auf Seite 84 angegebenen **PASCAL-XSC** Prozedur

```
INV( var A : Lrmatrix; illcd,Alefeld : integer;
     var X : Limatrix; var Err : integer);

           exportiert durch linv
```

Die Eingangsgröße $\boldsymbol{A}$ ist die zu invertierende, quadratische Matrix, und die Ausgangsgröße $\boldsymbol{X}$ liefert die garantierte Einschließung von $\boldsymbol{A}^{-1}$. Mit illcd=2 wird die Rechnung in *staggered arithmetic* ausgeführt, wobei der interaktiv zu wählende Genauigkeitsparameter p etwa $(p+1)\cdot 21$ interne Dezimalstellen festlegt und aus Laufzeitgründen möglichst klein sein sollte. Der Fehlerkode Err=0 bedeutet, daß die Einschließung $\boldsymbol{X}$ von $\boldsymbol{A}^{-1}$ nach dem Crout'schen Algorithmus erfolgreich und i.a. maximalgenau durchgeführt werden konnte.

1. Beispiel: Wir betrachten noch einmal die auf Seite 83 angegebene Matrix

$$\boldsymbol{A} := \begin{pmatrix} 1 & 1 & 10^{-21} & -10^{-81} \\ 0 & 1 & 1 & \frac{1}{3}(1-10^{-21}) \\ 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & (1+10^{-10})\cdot 10^{-40} \end{pmatrix}$$

für die wir eine maximalgenaue Einschließung des inversen Elements $\boldsymbol{A}^{-1}[1,4]$ bisher nicht berechnen konnten. Mit *staggered arithmetic* (illcd=2) und dem Genauigkeitsparemeter $p = 4$ liefert die Prozedur INV(A,illcd,Alefeld,X,Err) das Ergebnis:

$$\boldsymbol{A}^{-1}[1,4] \in [7.49999999925000000007 \cdot 10^{-42}, 7.49999999925000000008 \cdot 10^{-42}]$$

Beachten Sie bitte, daß wir jetzt mit $p = 4$ etwa $(p+1)\cdot 21 = 105$ interne Dezimalstellen benötigen, um obige Einschließung maximalgenau berechnen zu können!

2. Beispiel: Wir betrachten jetzt die schon auf Seite 40 definierten Blockmatrizen $\boldsymbol{B} = \boldsymbol{B}(n,v)$ der Ordnung $2n$. In nachfolgender Tabelle sind die Werte des Fehlerkodes Err für $v = 10^{-30}$ in Abhängigkeit von n, illcd, p und Alefeld zusammengestellt, die sich beim Aufruf der Prozedur

INV(B,illcd,Alefeld,X,Err)

von Seite 84 ergeben:

Fehlerkode der Prozedur: INV(B,illcd,Alefeld,X,Err)					
Mit illcd=2 liefert p etwa $(p+1)\cdot 21$ interne Dezimalstellen.					
$\boldsymbol{B} = \boldsymbol{B}(n,v)$, $v := 10^{-30}$					
n	illcd	p	Alefeld	Err	Einschließung
2	0		0	0	maximalgenau
3	0		0	4	
	1		0,1	0	zu grob
	2	4	0	0	maximalgenau
4	0		0	4	
	1		0	0	maximalgenau
5	0		0	4	
	1		0	0	maximalgenau

Fehlerkode der Prozedur: INV(B,illcd,Alefeld,X,Err)					
Mit illcd=2 liefert p etwa $(p+1)\cdot 21$ interne Dezimalstellen.					
$\boldsymbol{B} = \boldsymbol{B}(n, v), \quad v := 10^{-30}$					
n	illcd	p	Alefeld	Err	Einschließung
6	0		0	4	
	1		0,1	0	zu grob
	2	4	0	0	maximalgenau
7	0		0	4	
	1		0,1	0	zu grob
	2	4	0	0	maximalgenau
8	0		0,1	0	hochgenau
	1		0	0	maximalgenau
9	0		0	4	
	1		0	3	
	2	4	0	0	maximalgenau
10	0		0	4	
	1		0	0	maximalgenau
11	0		0	4	
	1		1,2	0	hochgenau
	2	4	0	0	maximalgenau
12	0		0	4	
	1		0,1	0	zu grob
	2	4	0	0	maximalgenau
13	0		0	4	
	1		0,1	0	zu grob
	2	4	0	0	maximalgenau

Dabei signalisiert Err=0, daß der Algorithmus zur Einschließung der Inversen $\boldsymbol{B}^{-1}$ erfolgreich durchgeführt werden konnte, was jedoch nicht bedeutet, daß die Ergebnisse alle maximalgenau ausfallen müssen. p ist der interaktiv zu bestimmende Genauigkeitsparameter, der aus Laufzeitgründen möglichst klein zu wählen ist. Bitte beachten Sie, daß z.B. mit $n = 13$ eine Blockmatrix der Ordnung $2 \cdot 13 = 26$ definiert wird. Aus der Tabelle ergibt sich, daß in 7 von 12 Fällen eine maximalgenaue Einschließung nur in *staggered arithmetic* möglich ist.

9 Intervallmatrizen

Bisher hatten wir die Aufgabe betrachtet, zu einer vorgegebenen regulären Punktmatrix $\boldsymbol{A} \in R^{n\times n}$ eine garantierte und möglichst maximalgenaue Einschließung $\boldsymbol{X} \in IR^{n\times n}$ ihrer Inversen $\boldsymbol{A}^{-1} \in \mathbb{R}^{n\times n}$ zu berechnen: $\boldsymbol{A}^{-1} \in \boldsymbol{X}$. Dabei bedeutet $R^{n\times n}$ die Menge aller quadratischen Gleitkomma-Matrizen, und mit $IR^{n\times n}$ bezeichnen wir ganz entsprechend die Menge aller quadratischen Intervall-Gleitkomma-Matrizen, jeweils mit der Dimension n. Das benutzte Gleitkommaraster ist dabei z.B. durch das Longreal-Format der BCD-Version von **PASCAL-XSC** vorgegeben [8].

Wegen der bisherigen Bedingung $\boldsymbol{A} \in R^{n\times n}$ kann daher eine Einschließung der Inversen mit den bisherigen Methoden schon dann nicht mehr berechnet werden, wenn z.B. $\boldsymbol{A}[1,1] = \sqrt{2} \notin R \subset \mathbb{R}$ vorgegeben ist. In diesem Fall müßte man das bisherige Punkt-Element $A[1,1]$ durch das Intervall

$$[1.41421356237309504880, 1.41421356237309504881] \ni \sqrt{2}$$

ersetzen, welches unter **PASCAL-XSC** mit Hilfe der Anweisung [8]

```
b := sqrt( Lintval(2) );  { var b : Linterval; }
```

berechnet wurde und damit die irrationale Zahl $\sqrt{2}$ garantiert enthält. Das angesprochene Problem veranlaßt uns daher, die zu invertierende Matrix $\boldsymbol{A}$ als Intervall-Matrix zu wählen, was unter **PASCAL-XSC** mit Hilfe des folgenden Programmteils

```
program test(input,output);
use LMVI_ARI;
var A : Limatrix[1..n,1..n];
begin                   end.
```

sehr einfach realisiert werden kann. Damit sind alle Elemente $\boldsymbol{A}[i,j]$ der quadratischen Matrix $\boldsymbol{A}$ vom Typ `Linterval`, wobei diese Intervall-Elemente jeweils so zu wählen sind, daß die gegebenen rationalen oder irrationalen Matrixelemente stets **maximalgenau** eingeschlossen werden, um bei den späteren Intervallrechnungen die bekannten Überschätzungen möglichst zu vermeiden. Die Zahl $1/4$ wird demnach als rationale Zahl durch das Punktintervall $[0.25, 0.25]$ mit dem Durchmesser 0 sogar exakt eingeschlossen.

9.1 Begriffsbildung

Wir betrachten also jetzt eine Intervallmatrix $\boldsymbol{A}$, deren Elemente $\boldsymbol{A}[i,j]$ durch Intervalle vom Typ `Linterval` gegeben sind. Ist der Durchmesser auch nur eines dieser Intervalle von Null verschieden, so sind mit $\boldsymbol{A}$ unendlich viele verschiedene Punktmatrizen $\boldsymbol{A}_P$ gegeben, wobei jedes $\boldsymbol{A}_P$ ein Element von $\boldsymbol{A}$ ist:

$$\boldsymbol{A}_P \in \boldsymbol{A}, \quad \text{falls} \quad \boldsymbol{A}_P[i,j] \in \boldsymbol{A}[i,j], \quad i,j = 1,2,\ldots,n$$

Unter $\boldsymbol{A}^{-1}$ verstehen wir dann die Menge aller inversen Matrizen $\boldsymbol{A}_P^{-1}$, wobei vorausgesetzt wird, daß zu jeder Punktmatrix $\boldsymbol{A}_P$ ihre Inverse $\boldsymbol{A}_P^{-1}$ auch wirklich existiert.

9.2 Der Inversionsalgorithmus

Gegeben sei eine quadratische Intervallmatrix $\boldsymbol{A}$, wobei vorausgesetzt wird, daß zu jeder Punktmatrix $\boldsymbol{A}_P \in \boldsymbol{A}$ auch ihre Inverse $\boldsymbol{A}_P^{-1}$ existiert. Gesucht ist jetzt ein Algorithmus, der eine Intervallmatrix $\boldsymbol{X}$ liefert, welche die Menge $\boldsymbol{A}^{-1}$ aller inversen Punktmatrizen $\boldsymbol{A}_P^{-1}$ garantiert und zudem möglichst eng einschließt:

$$\boldsymbol{A}^{-1} \subseteq \boldsymbol{X}; \qquad \texttt{var A,X : Lintervall[1..n,1..n];}$$

Die obige Bezeichnung *möglichst eng* weist darauf hin, daß die Intervallmatrix $\boldsymbol{X}$ auch Punktmatrizen enthalten kann, die keine Inverse einer Punktmatrix $\boldsymbol{A}_P \in \boldsymbol{A}$ ist. Da $\boldsymbol{X}$ u.a. durch Intervallrechnungen bestimmt wird, sind solche Überschätzungen i.a. nicht zu vermeiden.

Glücklicherweise können wir nun zur Berechnung von $\boldsymbol{X}$ den bekannten Algorithmus zur Einschließung der Inversen einer gegebenen Punktmatrix fast unverändert übernehmen. Es sind tatsächlich nur zwei Änderungen vorzunehmen:

1. Die Näherungsinverse $\boldsymbol{R}$ wird nicht bzgl. der früheren Punktmatrix $\boldsymbol{A}$ sondern jetzt bzgl. der Mittelpunktsmatrix `M := MID(A)` berechnet, wobei $\boldsymbol{A}$ die vorgegebene Intervall–Matrix ist: $\boldsymbol{R} \approx \boldsymbol{M}^{-1}$.

2. Im Verifikationsalgorithmus ist nach Gleichung (77) bei der Berechnung der Defekteinschließung $[^l\boldsymbol{D}]$ die frühere Punktmatrix $\boldsymbol{A}$ zu ersetzen durch die jetzt gegebene Intervallmatrix $[\boldsymbol{A}]$.

Zum Beweis, daß der so abgeänderte Algorithmus eine Einschließung $\boldsymbol{X}$ von $\boldsymbol{A}^{-1}$ liefert, muß lediglich gezeigt werden, daß für jede Punktmatrix $\boldsymbol{A}_P \in \boldsymbol{A}$ auch ihre Inverse $\boldsymbol{A}_P^{-1}$ in $\boldsymbol{X}$ enthalten ist. Um dies einzusehen, formulieren wir den neuen Algorithmus für die Punktmatrix $\boldsymbol{A}_P \in \boldsymbol{A}$ in Kurzform:

1. Berechnung der Mittelpunktsmatrix: `M := MID(A)`
2. Berechnung einer Näherungsinversen: $\boldsymbol{R} \approx \boldsymbol{M}^{-1}$
3. Der Ansatz $\boldsymbol{A}_P^{-1} = \boldsymbol{R} + \boldsymbol{Y}$ liefert mit dem Defekt ${}^l\boldsymbol{D}_P := \boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}_P$ für die Korrektur $\boldsymbol{Y}$ die Fixpunktgleichung $\boldsymbol{Y} = {}^l\boldsymbol{D}_P(\boldsymbol{R}+\boldsymbol{Y})$, und mit der Defekteinschließung $[{}^l\boldsymbol{D}_P] := \Diamond(\boldsymbol{I} - \boldsymbol{R}\boldsymbol{A}_P)$ lautet die entsprechende Iterationsgleichung nach (77):

   (88) $[\boldsymbol{Y}]_\Diamond^{(\nu+1)} := [{}^l\boldsymbol{D}_P] \diamond (\boldsymbol{R} \oplus [\boldsymbol{Y}]_\Diamond^{(\nu)})$ und nach (76) gilt:

   (89) $[\boldsymbol{Y}]_\Diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]_\Diamond^{(k)} \implies \boldsymbol{Y} \in [\boldsymbol{Y}]_\Diamond^{(k+1)}$ ist eindeutig.

4. Bedeutet nun $[\boldsymbol{A}]$ die gegebene Intervall–Matrix, so lautet mit der neuen Defekteinschließung $[{}^l\boldsymbol{D}] := \Diamond(\boldsymbol{I} - \boldsymbol{R}[\boldsymbol{A}])$ die Iterationsgleichung:

   (90) $[\boldsymbol{Y}]_\Diamond^{(\nu+1)} := [{}^l\boldsymbol{D}] \diamond (\boldsymbol{R} \oplus [\boldsymbol{Y}]_\Diamond^{(\nu)})$ und analog schließen wir:

   (91) $[\boldsymbol{Y}]_\Diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]_\Diamond^{(k)} \implies \boldsymbol{Y} \in [\boldsymbol{Y}]_\Diamond^{(k+1)}$ ist eindeutig.

5. $$\boldsymbol{A}^{-1} \in \boldsymbol{X} := \boldsymbol{R} \oplus [\boldsymbol{Y}]_\Diamond^{(k+1)}$$

Der obige Punkt 3. ist natürlich nicht Bestandteil des neuen Algorithmus zur Berechnung von $\boldsymbol{X}$, sondern erläutert lediglich das Zustandekommen der Iterationsgleichungen in den Punkten 3. und 4.

Die Richtigkeit der Aussage $\boldsymbol{Y} \in [\boldsymbol{Y}]_\Diamond^{(k+1)}$ in (91) ergibt sich nun sehr einfach, wenn wir zunächst annehmen, daß für einen speziellen Index $\nu = k$ die Voraussetzung $[\boldsymbol{Y}]_\Diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]_\Diamond^{(k)}$ in (91) erfüllt ist. Berechnet man dann mit der gleichen Intervall–Matrix $[\boldsymbol{Y}]_\Diamond^{(k)}$ den anderen Wert $[\boldsymbol{Y}]_\Diamond^{(k+1)}$ jetzt nach (88), so ist dieser Wert wegen $\boldsymbol{A}_P \in [\boldsymbol{A}]$ nach der Inklusionsmonotonie der Intervallrechnung in dem nach (90) berechneten Wert $[\boldsymbol{Y}]_\Diamond^{(k+1)}$ enthalten, so daß dann damit auch die Voraussetzung $[\boldsymbol{Y}]_\Diamond^{(k+1)} \overset{\circ}{\subset} [\boldsymbol{Y}]_\Diamond^{(k)}$ in (89) erfüllt ist, woraus schließlich die in (91) angegebene Behauptung $\boldsymbol{Y} \in [\boldsymbol{Y}]_\Diamond^{(k+1)}$ in (89) unmittelbar folgt. Da diese Überlegungen für alle $\boldsymbol{A}_P \in [\boldsymbol{A}]$ gelten, haben wir mit $\boldsymbol{X}$ nach 5. tatsächlich eine garantierte Einschließung für $\boldsymbol{A}^{-1}$ berechnet.

9.3 Numerische Beispiele

Der auf Seite 97 angegebene Algorithmus zur Berechnung einer Einschließung $\boldsymbol{X}$ der Inversen $\boldsymbol{A}^{-1}$ einer vorgegebenen quadratischen Intervallmatrix $\boldsymbol{A}$ wird realisiert durch die folgende Prozedur

INV(var A : Limatrix; illcd : integer;
var X : Limatrix; var Err : integer);

exportiert durch linv

Der Eingangsparameter `illcd` steuert mit den Werten 0 bzw. 1 die Auswertung in einfacher bzw. doppeltlanger Rechnung, und der Fehlerkode Err=0 signalisiert, daß die Einschließung $\boldsymbol{X}$ erfolgreich berechnet werden konnte.

1. Beispiel:
Im ersten Beispiel soll die obige Prozedur grob getestet werden, indem zwei gleiche Matrizen $\boldsymbol{A} = \boldsymbol{B}$ zu invertieren sind, wobei nach Voraussetzung die Punktintervalle der Intervall–Matrix $\boldsymbol{B}$ genau die entsprechenden Elemente der Gleitkomma–Matrix $\boldsymbol{A}$ enthalten sollen. Wegen der **Punkt**–Intervalle von $\boldsymbol{B}$ müssen dann die Einschließungen von $\boldsymbol{A}^{-1}$ und $\boldsymbol{B}^{-1}$ identisch ausfallen. $\boldsymbol{A}$ wird als Zufallsmatrix mit der Prozedur `RAND_M(A,Long(2),Long(1e3))` berechnet, wobei die $\boldsymbol{A}[i,j]$ zwischen 2 und 1000 liegen und die Prozedur `RAND_M` vom Modul `mvran` exportiert wird. Die Anweisung `B := A` liefert dann die Intervall–Matrix $\boldsymbol{B}$ mit den Punktintervallen $\boldsymbol{B}[i,j]$.

Im nachfolgenden **PASCAL-XSC** Programm kann der Anwender die Matrixdimension n frei wählen, und die beiden Prozedur–Aufrufe `INV(...)` liefern tatsächlich identische Einschließungen $\boldsymbol{X}$ von $\boldsymbol{A}^{-1}$ und $\boldsymbol{B}^{-1}$.

```
program inv1(input,output);
use mvran, linv, lmv_ari, lmvi_ari;
var n : integer;
 procedure main(n: integer);
 var A                    : Lrmatrix[1..n,1..n];
     X,B                  : Limatrix[1..n,1..n];
     Err,k,illcd,Alefeld  : integer;
 begin { main }
  RAND_M(A,Long(2),Long(1e3));  { A: Zufallsmatrix }
```

```
    writeln('illcd = ?');    read(illcd);
    writeln('Alefeld = ?');  read(Alefeld);
    INV(A,illcd,Alefeld,X,Err);
    if Err = 0 then
    begin
     writeln('Zur X-Ausgabe k=0 eingeben:');  read(k);
     if k=0 then writeln(X);
    end        else writeln('Err = ',Err);

    B := A; { B=A; B: Matrix mit Punktintervallen }
    writeln('illcd = ?');  read(illcd);
    INV(B,illcd,X,Err);
    if Err = 0 then
    begin
     writeln('Zur X-Ausgabe k=0 eingeben:');  read(k);
     if k=0 then writeln(X);
    end        else writeln('Err = ',Err);
   end;  { main }
  begin
    writeln('Dimenison der Matrix = ?');  read(n);
    main(n);
  end.
```

2. Beispiel:
Dieses Beispiel zeigt die Aufblähungseffekte der Einschließung $\boldsymbol{X}$ von $\boldsymbol{A}^{-1}$, wenn die Intervalle $\boldsymbol{A}[i,j]$ der Intervall–Matrix $\boldsymbol{A}$ nur geringfügig vergrößert werden. Die Matrix $\boldsymbol{A}$ wird mit Hilfe des Prozeduraufrufs

`RAND_M(A,Long(2),Long(1e3),eps)`

als Zufalls–Intervallmatrix generiert, wobei der positive Eingangsparameter `eps` die Breite der Intervallelemente $\boldsymbol{A}[i,j]$ steuert. Die Eingangsparameter `Long(2)` und `Long(1e3)` liefern das Intervall $[2,1000]$, in dem alle $\boldsymbol{A}[i,j]$ enthalten sind. Die obige Prozedur `RAND_M(...)` wird vom Modul `limvran` exportiert. In der nachfolgenden Tabelle werden die willkürlich gewählten Elemente $\boldsymbol{A}[1,1]$ und $\boldsymbol{X}[10,10]$ in Abhängigkeit von den vorgegeben `eps`–Werten verglichen. Die Dimension n der zu invertierenden Zufallsmatrizen $\boldsymbol{A}$ ist gegeben durch: $n = 10$.

$\boldsymbol{A}[1,1]$ **und** $\boldsymbol{X}[10,10]$ **in Abhängigkeit von** `eps`; **Dim($\boldsymbol{A}$)=10**	
$\boldsymbol{X} \supseteq \boldsymbol{A}^{-1}$ **wird berechnet mit dem Prozeduraufruf:** `INV(A,0,X,Err)`	
Obige Prozedur wird exportiert vom Modul `linv`	
`eps` = 0	
$\boldsymbol{A}[1,1]$	$[+2.19984060399917075596\mathbf{6} \cdot 10^{+2}, +2.1998406039991707559\mathbf{8} \cdot 10^{+2}]$
$\boldsymbol{X}[10,10]$	$[+1.506032135751798712\mathbf{22} \cdot 10^{-3}, +1.506032135751798712\mathbf{36} \cdot 10^{-3}]$
`eps` = 10	
$\boldsymbol{A}[1,1]$	$[+5.414575246253318733\mathbf{59} \cdot 10^{+2}, +5.414575246253318733\mathbf{71} \cdot 10^{+2}]$
$\boldsymbol{X}[10,10]$	$[+4.3622410345935540\mathbf{1064} \cdot 10^{-3}, +4.3622410345935540\mathbf{2138} \cdot 10^{-3}]$
`eps` = 100	
$\boldsymbol{A}[1,1]$	$[+2.16484719678985289\mathbf{999} \cdot 10^{+2}, +2.16484719678985290\mathbf{101} \cdot 10^{+2}]$
$\boldsymbol{X}[10,10]$	$[-2.8275809158635185\mathbf{7875} \cdot 10^{-3}, -2.8275809158635185\mathbf{1816} \cdot 10^{-3}]$
`eps` = 1 000	
$\boldsymbol{A}[1,1]$	$[+9.2759749874547007\mathbf{3999} \cdot 10^{+2}, +9.2759749874547007\mathbf{5001} \cdot 10^{+2}]$
$\boldsymbol{X}[10,10]$	$[-1.011503793958845\mathbf{84096} \cdot 10^{-3}, -1.011503793958845\mathbf{76305} \cdot 10^{-3}]$
`eps` = 10 000	
$\boldsymbol{A}[1,1]$	$[+5.414575246253318\mathbf{69999} \cdot 10^{+2}, +5.414575246253318\mathbf{80001} \cdot 10^{+2}]$
$\boldsymbol{X}[10,10]$	$[+4.3622410345935\mathbf{4913176} \cdot 10^{-3}, +4.3622410345935\mathbf{5791959} \cdot 10^{-3}]$
`eps` = 100 000	
$\boldsymbol{A}[1,1]$	$[+2.16484719678985\mathbf{199999} \cdot 10^{+2}, +2.16484719678985\mathbf{300001} \cdot 10^{+2}]$
$\boldsymbol{X}[10,10]$	$[-2.827580915863\mathbf{54577297} \cdot 10^{-3}, -2.827580915863\mathbf{48517869} \cdot 10^{-3}]$

Mit Hilfe der fettgedruckten Ziffern, welche die Breite der jeweiligen Intervalle verdeutlichen, erkennt man, daß die Intervalle $\boldsymbol{X}[i,j]$ stets eine größere Breite aufweisen als die Intervalle $\boldsymbol{A}[i,j]$ der zu invertierenden Matrix $\boldsymbol{A}$. Beachten Sie bitte, daß unsere Matrizen $\boldsymbol{A}$ als Zufallsmatrizen fast immer gut konditioniert sind, so daß die Aufblähungen der Intervalle $\boldsymbol{X}[i,j]$ nicht durch die schlechte Konditioniertheit von $\boldsymbol{A}$, sondern hauptsächlich durch die Intervallrechnungen der notwendigen Ergebnisverifikation bedingt sind. Bei größeren Matrixdimensionen n wird dieser Effekt noch wesentlich verstärkt, wovon sich der Leser mit Hilfe des nachfolgenden Programms überzeugen möge, mit dem auch die obigen Tabellenwerte berechnet wurden.

```
program inv2(input,output);
use limvran, inv_lm, lmvi_ari, li_ari;
var n : integer;
 procedure main(n: integer);
 var A,X : Limatrix[1..n,1..n];
     Err : integer;
     eps : real;
 begin { main }
  repeat
   writeln('eps   = ?');  read(eps);
   RAND_M(A,Long(2),Long(1e3),eps); { A: Zufallsmatrix }
   writeln('A[1,1] = ', A[1,1]);
   INV(A,0,X,Err);   { 0: Rechnung in einfacher Laenge }
   if Err = 0 then writeln(X[n,n])
              else writeln('Err = ',Err);
  until Err <> 0;
 end;  { main }
begin
  writeln('Dimenison der Matrix = ?');  read(n);
  main(n);
end.
```

3. Beispiel:

Im letzten Beispiel wird gezeigt, welche gewaltige Aufblähung die Einschließung einer Inversen erfährt, wenn bei einer vorgegebenen schlechtkonditionierten Punktmatrix nur ein einziges Punktelement in ein Longreal–Intervall mit minimalem, von Null verschiedenen Intervalldurchmesser, umgewandelt wird. Um diesen Effekt zu verstehen, betrachten wir zunächst nur die beiden Punktmatrizen $\boldsymbol{A}_1$ und $\boldsymbol{A}_2$:

$$\boldsymbol{A}_1 := \begin{pmatrix} 1 & 1 & 1 & 1 & 10^{-30} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 10^{-30} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 10^{-30} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 10^{-30} \\ 10^{-30} & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 10^{-30} & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 10^{-30} & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 10^{-30} & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$A_2 := \begin{pmatrix} 1 & 1 & 1 & 1 & 10^{-30} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 10^{-30} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 10^{-30} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 10^{-30} \\ 10^{-30} & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 10^{-30} & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 10^{-30} & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 10^{-30} & 1 & 1 & 1 & 1+10^{-20} \end{pmatrix}$$

Mit Hilfe der Prozedur `INV(...)` von Seite 84 berechnen wir zunächst die Einschließungen $\boldsymbol{X}_1$ bzw. $\boldsymbol{X}_2$ der Inversen $\boldsymbol{A}_1^{-1}$ bzw. $\boldsymbol{A}_2^{-1}$. Die Ergebnisse sind in nachfolgender Tabelle zusammengestellt:

Berechnung der Einschließungen X_1 und X_2 von A_1^{-1} und A_2^{-1}
mit Hilfe der Prozedur `INV(A,illcd,Alefeld,X,Err);` `Alefeld = 0`
Obige Prozedur wird exportiert vom Modul `linv`
`INV(A1,1,0,X1,Err)` `illcd=1: Doppeltlange Rechnung`
$\boldsymbol{A}_1^{-1}[1,4] \in [6.250000\ \ldots\ 0000 \cdot 10^{-2}, 6.250000\ \ldots\ 0001 \cdot 10^{-2}] \ = \boldsymbol{X}_1[1,4]$
`INV(A2,2,0,X2,Err)` `illcd=2: Staggered arithmetic mit p=2`
$\boldsymbol{A}_2^{-1}[1,4] \in [1.874999\ \ldots\ 9999 \cdot 10^{+39}, 1.875000\ \ldots\ 0000 \cdot 10^{+39}] = \boldsymbol{X}_2[1,4]$

Die Einschließungen von $\boldsymbol{A}_1^{-1}[1,4]$ und $\boldsymbol{A}_2^{-1}[1,4]$ unterscheiden sich also um stolze 41 Zehnerpotenzen, obwohl lediglich das Punktelement $\boldsymbol{A}_2[1,4]$ um nur eine Einheit auf der letzten Stelle größer ist als $\boldsymbol{A}_1[1,4]$. Wir erkennen auch hier wieder die bekannte Tatsache, daß bei einer schlechtkonditionierten Matrix eine minimale Änderung nur eines Elements zu großen Änderungen der Inversen führen kann. Beachten Sie bitte, daß die Konditionszahlen $\varkappa$ der beiden Matrizen $\boldsymbol{A}_1$ und $\boldsymbol{A}_2$ gegeben sind durch:

$$\varkappa(\boldsymbol{A}_1) = 4.8 \cdot 10^{+31}, \qquad \varkappa(\boldsymbol{A}_2) = 3.6 \cdot 10^{+41}$$

Der große Wert $\varkappa(\boldsymbol{A}_2) = 3.6 \cdot 10^{+41}$ führt übrigens dazu, daß die Einschließung von $\boldsymbol{A}_2^{-1}$ mit Hilfe der Prozedur `INV(A2,2,0,X2,Err)` nach obiger Tabelle nur in *staggered arithmetic* (`illcd=2`) erfolgreich berechnet werden konnte.

Nach diesen Vorbereitungen ist jetzt eine Intervallmatrix zu invertieren, deren Elemente alle Punktintervalle sind, mit Ausnahme nur eines Elements, dessen Intervall sogar noch den kleinstmöglichen Durchmesser haben soll. Wir wählen:

$$\boldsymbol{B} := \begin{pmatrix} 1 & 1 & 1 & 1 & 10^{-30} & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 10^{-30} & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 10^{-30} & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 10^{-30} \\ 10^{-30} & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 10^{-30} & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 10^{-30} & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 10^{-30} & 1 & 1 & 1 & [1, 1+10^{-20}] \end{pmatrix}$$

Vergleicht man $\boldsymbol{B}$ mit $\boldsymbol{A}_1$ und $\boldsymbol{A}_2$, so verlangen die bisherigen Ergebnisse der Tabelle von Seite 102, daß mit einer berechneten Einschließung $\boldsymbol{X}$ von $\boldsymbol{B}^{-1}$ das Element $\boldsymbol{X}[1,4]$ das Intervall

$$y := [6.25000000000000000000 \cdot 10^{-2}, 1.87500000000000000000 \cdot 10^{+39}]$$

enthalten muß. Mit dem Aufruf der Prozedur `INV(B,illcd,X,Err)` von Seite 98 und dem Eingangsparameter `illcd=1` für doppeltlange Rechnung erhält man das Intervall

$$\boldsymbol{X}[1,4] := [6.24999999999999999999 \cdot 10^{-2}, 1.87500000000000000003 \cdot 10^{+39}],$$

welches das obige Intervall y tatsächlich enthält und nur ganz unwesentlich überschätzt.

Abschließend soll in diesem Zusammenhang ein Problem angesprochen werden, das entsteht, wenn z.B. die Matrix $\boldsymbol{A}_2$ von Seite 102 nicht im BCD–Format sondern im IEEE–Format zu invertieren ist. Da in diesem Format die Elemente 10^{-30} und $(1 + 10^{-20})$ nicht exakt darstellbar sind, müssen diese Dezimalwerte durch Intervalle maximalgenau eingeschlossen werden, so daß damit eine Intervallmatrix zu invertieren ist. Wegen der schlechten Kondition von $\boldsymbol{A}_2$ werden wir dann wie bei der Intervallmatrix $\boldsymbol{B}$ für die Inverse so breite Einschließungsintervalle erhalten, daß die Ergebnisse damit vom numerischen Standpunkt aus völlig wertlos sind. Im Gegensatz dazu ist $\boldsymbol{A}_2$ im Longreal–Format der BCD–Version von **PASCAL-XSC** exakt darstellbar, und die Prozedur `INV(A2,2,0,X2,Err)` liefert mit $\boldsymbol{X}_2$ sogar eine maximalgenaue Einschließung der Inversen $\boldsymbol{A}_2^{-1}$.

10 Prozeduren des Moduls `linv`

In diesem Abschnitt werden alle Funktionen, Prozeduren und Operatoren aufgelistet, die vom Modul `linv` exportiert werden. Dieses Modul stellt alle Prozeduren zur Verfügung, welche zur Inversion quadratischer, reeller Matrizen bzgl. des Typs `Longreal` benötigt und in diesem Buch beschrieben werden.

Für alle Prozeduren findet man dabei ausführliche Beschreibungen der jeweiligen Eingangs- und Ausgangsparameter. Dabei liefern die zugehörigen Seitenzahlen weitere Informationen, die in den entsprechenden Abschnitten dieses Buches nachgelesen werden können. Auch solche Anwender, die mit den Verifikationsmethoden nur wenig oder gar nicht vertraut sind, sollten mit den nachfolgenden Kurzinformationen in der Lage sein, ihre Probleme bei der Matrixinversion mit Hilfe der angebotenen Prozeduren direkt lösen zu können. Wird von einer Prozedur die Einschließung der inversen Matrix erfolgreich (`Err=0`) berechnet, so wird garantiert, daß diese Einschließung die Inverse wirklich mit mathematischer Sicherheit enthält. An dieser Stelle soll nochmals darauf hingewiesen werden, daß gerade die **garantierte** Einschließung des exakten Ergebnisses einer mathematischen Operation (z.B. Matrixinversion) das Herzstück der **PASCAL-XSC** Philosophie darstellt.

Beachten Sie bitte, daß alle Numerik–Tools wie etwa *Mathematica* [12] Rechnungen mit fast beliebig vielen Dezimalstellen zulassen, daß es jedoch keine mathematisch gesicherten Informationen darüber gibt, welche dieser Ziffern das exakte Ergebnis mit welcher Genauigkeit beschreiben. Es ist aus mathematischer Sicht auch völlig unzureichend, wenn man auf den ersten Blick hin überzeugende Genauigkeitsangaben findet, die für den Anwender dann jedoch nicht wirklich nachvollziehbar sind. Wenn beispielsweise ein Chip-Hersteller die Fehlerschranken der mitgelieferten Standardfunktionen angeben würde, so wäre das zwar schon ein wesentlicher Fortschritt. Aus mathematischer Sicht sind diese Fehlerschranken jedoch nur dann brauchbar, wenn deren Berechnungen in allen Einzelheiten zugänglich sind, wie dies in den verschiedenen **XSC**–Umgebungen gewährleistet ist.

Abschließend noch ein Hinweis an alle Mathematiker, die z.B. an den Universitäten versuchen, die Studenten in die Grundlagen der exakten Wissenschaften einzuführen. Aus der Sicht eines in der Regel nur nach harter Arbeit erfolgreichen Anfängers muß es dann jedoch ganz verblüffend sein, mit welcher Großzügigkeit die ehemaligen Lehrmeister numerische Software ohne Ergebnisverifikation fast immer bedenkenlos anwenden.

10.1 Importierte Module

Zunächst einige Hinweise zum Gebrauch von Modul–Namen:

Der Compiler unterscheidet Modul–Namen nur mit Hilfe der ersten sechs Zeichen, wobei auch mehr als sechs Zeichen benutzt werden dürfen. Da in der UNIX–Umgebung zwischen Groß- und Kleinschreibung unterschieden wird, sind `linv` und `LINV` damit unterschiedliche Namen. Um unnötige Konflikte zu vermeiden, sollte man sich jedoch darauf einigen, in Modul–Namen nur kleine Buchstaben zu verwenden!

Die folgende Tabelle enthält alle Hilfs- und System–Module [8, Seite 217], die vom Modul `linv` importiert werden:

Das Modul `linv` importiert die folgenden Module	
Modul	**exportiert u.a.**
`type_def`	Datentypen wie: `Stagg, Lstagg, LStaggrmatrix, ...`
`lstgmv`	Matrix/Vektor–Arithmetik zum Basis–Typ `Lstagg`
`lstagg`	Staggered arithmetic zum Basis–Typ `Longreal`
`lmvf0`	Matrix- und Vektor–Funktionen zum Basistyp `Longreal`
`mv_comp`	Maximalg. Berechng. von z.B. $\boldsymbol{AB} + \boldsymbol{C}$ zum Typ `Longreal`
`li_ari`*	Intervall–Arithmetik zum Basistyp `Longreal`
`lmv_ari`*	Matrix/Vektor–Arithmetik zum Basistyp `Longreal`
`lmvi_ari`*	Intervall–Matrix/Vektor–Arithmetik zum Typ `Longreal`
Das Symbol * kennzeichnet importierte System–Module, [8, Seite 217]	

Alle System–Module der BCD-Version von **PASCAL-XSC** befinden sich im Verzeichnis:

`/home/software/pxscd/sys/`

Die Hilfs–Module der BCD-Version von **PASCAL-XSC** befinden sich im Verzeichnis:

`/home/software/pxscd/user/`

10.2 Exportierte Prozeduren

10.2.1 Berechng. einer Näherungsinversen mit LU–Zerlegung durch: `INV_LU(A,R,Err)`

Prozedurvereinbarung:

```
procedure INV_LU ( var A   : Lrmatrix;
                   var R   : Lrmatrix;
                   var Err : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
Ausgangsparameter	
R	Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Prozeduraufruf `INV_LU(A,R,Err)` wird mit Hilfe der LU–Zerlegung von **A** eine brauchbare Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ berechnet, wenn **A** nicht zu schlecht konditioniert ist. Die quadratischen Matrizen **A**, **R** müssen die gleiche Dimension besitzen.

Weitere Informationen:
Seite 19 in diesem Buch, [11, Seite 18].

10.2.2 Berechng. einer Näherungsinversen mit LU–Zerlegung oder mit dem Verfahren von Rump durch: `INV_RUMP(A,illcd,factor,R,Err)`

Prozedurvereinbarung:

```
procedure INV_RUMP ( var A             : Lrmatrix;
                         illcd, factor : integer;
                     var R             : Lrmatrix;
                     var Err           : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
illcd	Steuert den internen Algorithmus
factor	Bestimmt die Versionen des Verfahrens von Rump
Ausgangsparameter	
R	Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:

Mit dem Prozeduraufruf `INV_RUMP(A,illcd,factor,R,Err)` wird eine Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ berechnet.

`illcd=0`: Es wird nur die LU–Zerlegung benutzt.

`illcd=1`: Es wird das Verfahren von Rump angewandt, wobei der integer–Parameter `factor` die Version links (0) bzw. rechts (1) bestimmt.

Die quadratischen Matrizen **A**, **R** müssen die gleiche Dimension besitzen.

Weitere Informationen:

Seite 28 in diesem Buch.

10.2.3 Berechnung einer Näherungsinversen nach Rump durch: `INV_RUMP(A,factor,R1,R2,Err)`

Prozedurvereinbarung:

```
procedure INV_RUMP ( var A      : Lrmatrix;
                         factor : integer;
                     var R1,R2  : Lrmatrix;
                     var Err    : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
factor	Bestimmt die Versionen des Verfahrens von Rump
Ausgangsparameter	
R1,R2	Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Prozeduraufruf `INV_RUMP(A,factor,R1,R2,Err)` wird nach dem Verfahren von Rump eine Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ berechnet.
`factor=0`: Es wird die Version *links* gewählt.
`factor=1`: Es wird die Version *rechts* gewählt.
Die quadratischen Matrizen **A**,**R1**,**R2** müssen gleiche Dimensionen besitzen.

Weitere Informationen:
Seite 33 in diesem Buch.

10.2.4 Berechnung einer Näherungsinversen nach Rump durch: `INV_RUMP(A,R1,R2,factor,Err)`

Prozedurvereinbarung:

procedure `INV_RUMP` (	**var**	**A**	: `Lrmatrix`;
	var	**R1,R2**	: `Lrmatrix`;
	var	**factor**	: `integer`;
	var	**Err**	: `integer`);

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
Ausgangsparameter	
R1,R2	Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$
factor	Liefert Rump–Version mit kleinster Defektnorm
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Prozeduraufruf `INV_RUMP(A,R1,R2,facor,Err)` wird nach dem Verfahren von Rump eine Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ so berechnet, daß die zugehörige Defektnorm minimal wird. Der Parameter `factor` liefert die dabei benutzte Version des Rump'schen Verfahrens:
`factor=0`: Die Defektnorm bzgl. der Version *links* ist minimal.
`factor=1`: Die Defektnorm bzgl. der Version *rechts* ist minimal.
Die quadratischen Matrizen **A**,**R1**,**R2** müssen gleiche Dimensionen besitzen.

Weitere Informationen:
Seite 33,42 in diesem Buch.

10.2.5 Verbesserung einer Näherungsinversen mit Defekt-Iteration durch: `INV_DEF(A,factor,R,Err)`

Prozedurvereinbarung:

```
procedure INV_DEF ( var A      : Lrmatrix;
                        factor : integer;
                    var R      : Lrmatrix;
                    var Err    : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
factor	Legt eine der beiden Defektdefinitionen fest.
R	Erste Näherung der Inversen: $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$
Ausgangsparameter	
R	i.a. maximalgenaue Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Aufruf `INV_DEF(A,facor,R,Err)` wird zu gegebener quadratischer Matrix **A** und zu einer ersten gegebenen Näherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer Defektiteration eine i.a. maximalgenaue Näherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ berechnet. Der integer–Parameter `factor` bestimmt dabei die benutzte Defektdefinition:
`factor=0`: Defektdefinition *links*;
`factor=1`: Defektdefinition *rechts*;
Die quadratischen Matrizen **A**,**R** müssen gleiche Dimensionen besitzen.

Weitere Informationen:
Seite 44,46 in diesem Buch.

10.2.6 Berechnung einer Näherungsinversen mit Defekt-Iteration durch: `INV(A,R,factor,Err)`

Prozedurvereinbarung:

```
procedure INV ( var  A      : Lrmatrix;
                var  R      : Lrmatrix;
                var  factor : integer;
                var  Err    : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
Ausgangsparameter	
R	i.a. maximalgenaue Näherungsinverse $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$
factor	Definiert Defektdefinition mit minimaler Defektnorm.
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:

Mit dem Aufruf `INV(A,R,facor,Err)` wird zu gegebener quadratischer Matrix **A** eine i.a. maximalgenaue Näherung $\boldsymbol{R} \approx \boldsymbol{A}^{-1}$ wie folgt berechnet: Zunächst LU–Zerlegung, dann Suche nach der Defektdefinition mit minimaler Defektnorm mit anschließender Defektiteration. Die berechnete Näherung besitzt eine minimale Defeknorm. Der integer–Parameter `factor` definiert die benutzte Defektdefinition:

`factor=0`: Defektdefinition *links*;

`factor=1`: Defektdefinition *rechts*;

Die quadratischen Matrizen **A**,**R** müssen gleiche Dimensionen besitzen.

Weitere Informationen: Seite 51 in diesem Buch.

10.2.7 Verbesserung einer Näherungsinversen mit Defekt-Iteration durch: `INV_DEF(A,factor,R1,R2,Err)`

Prozedurvereinbarung:

```
procedure INV_DEF ( var A      : Lrmatrix;
                        factor : integer;
                    var R1,R2  : Lrmatrix;
                    var Err    : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
factor	Bestimmt die vorgegebene Defektdefinition.
R1,R2	Grobe Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$
Ausgangsparameter	
R1,R2	i.a. maximalgenaue Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:

Mit dem Aufruf `INV_DEF(A,facor,R1,R2,Err)` wird zu gegebener quadratischer Matrix **A** und zu einer vorgegebenen Näherung $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ mit Hilfe einer Defekt–Iteration eine doppeltlange Näherung $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ i.a. maximalgenaue berechnet. Mit dem integer–Parameter `factor` bestimmt man die gewünschte Defektdefinition:

`factor=0`: Defektdefinition *links*;

`factor=1`: Defektdefinition *rechts*;

Die quadratischen Matrizen **A**,**R1**,**R2** müssen gleiche Dimensionen besitzen.

Weitere Informationen: Seite 62 in diesem Buch.

10.2.8 Berechnung einer Näherungsinversen in doppelter Länge durch: `INV(A,R1,R2,factor,Err)`

Prozedurvereinbarung:

```
procedure INV ( var  A      : Lrmatrix;
                var  R1,R2  : Lrmatrix;
                var  factor : integer;
                var  Err    : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Matrix
Ausgangsparameter	
R1,R2	i.a. maximalgenaue Näherungsinverse $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$
factor	Liefert die benutzte, günstigste Defektdefinition.
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Aufruf `INV(A,R1,R2,factor,Err)` wird zu gegebener quadratischer Matrix **A** mit Hilfe des Rump'schen Verfahrens und anschließender Defekt–Iteration eine doppeltlange Näherung $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$ i.a. maximalgenaue berechnet. Der integer–Parameter `factor` liefert die intern benutzte, günstigste Defektdefinition:
`factor=0`: Defektdefinition *links*;
`factor=1`: Defektdefinition *rechts*;
Die quadratischen Matrizen **A**,**R1**,**R2** müssen gleiche Dimensionen besitzen.

Weitere Informationen:
Seite 63 in diesem Buch.

10.2.9 Einschließung einer Inversen mit staggered arithmetic durch: `INV(A,S,Err)`

Prozedurvereinbarung:

```
procedure INV ( var A   : Lrmatrix;
                var S   : LStaggrmatrix;
                var Err : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Punkt–Matrix
Ausgangsparameter	
S	Einschließung der Inversen: $\boldsymbol{A}^{-1} \in \boldsymbol{S}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Aufruf `INV(A,S,Err)` wird zu gegebener quadratischer Punkt–Matrix **A** mit **S** eine Einschließung der Inversen $\boldsymbol{A}^{-1}$ in staggered arithmetic berechnet. Um den Genauigkeitsparameter z.B. auf $p = 4$ zu setzen, muß vorher die Prozedur `set_LStagg_prec(4)` aufgerufen werden, womit dann ca. $(4+1) \cdot 21 = 105$ Ziffern für die interne Rechnung zur Verfügung stehen. Die Prozedur `set_LStagg_prec(...)` wird vom Modul `type_def` exportiert. Die quadratischen Matrizen **A**, **S** müssen gleiche Dimensionen besitzen.

Weitere Informationen:
Seite 69 in diesem Buch.

Programm-Beispiel: Siehe nächste Seite!

Programmbeispiel:

```
program inv3(input,output);
use type_def, linv, lmv_ari, lmvf0,
    lstagg, lstgmv, l_block;
var n,p : integer;
procedure main(n: integer);
var U,V   : Lrmatrix[1..n,1..n];
    A     : Lrmatrix[1..2*n,1..2*n];
    S     : LStaggrmatrix[1..2*n,1..2*n,-1..LStagg_prec];
    Err,p : integer;
begin { main }
   V := 1e-30 * LID(V);
   U := LONG(1);
   block(U,V,A,Err); { Erzeugt aus U,V die Blockmatrix A }
   {   A[ub(A,1),ub(A,2)] := Long(1)+1e-20;    }
   INV(A,S,Err);     { S ist Einschlie"sung von A^(-1)   }
   if Err = 0 then
   begin
      writeln('A^(-1)[1,4] wird eingeschlossen durch:');
      writeln('S[1,4] = ');     writeln(S[1,4]);
   end        else writeln('Err = ',Err);
end;  { main }

begin
   repeat
  writeln('Der voreingestellte Genauigkeits-Parameter der');
  writeln('staggered arithmetic ist p = ',get_LStagg_prec );
  writeln('Dies liefert ca. (p+1)*21 = ',(LStagg_prec+1)*21,
          ' interne Stellen.');
  writeln('Welcher p-Wert wird gewuenscht ?');  read(p);
  set_LStagg_prec(p);
  writeln('Dimenison der Matrix = ?');  read(n);
  writeln('Die gewaehlte Dimension der Blockmatrix ist: ',2*n);
  writeln('Bitte ein wenig Geduld ...');
  main(n);
   until false;
end.
```

10.2.10 Berechnung einer Einschließung der Inversen bei gegebener Punkt–Matrix A durch: `INV(A,illcd,Alefeld,X,Err)`

Prozedurvereinbarung:

```
procedure INV ( var A       : Lrmatrix;
                    illcd   : integer;
                    Alefeld : integer;
                var X       : Limatrix;
                var Err     : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Quadratische Matrix, deren Inverse einzuschließen ist
illcd	Steuert den internen Algorithmus
Alefeld	Anzahl der Nachiterationen zur Verbesserung von **X**
Ausgangsparameter	
X	i.a. maximalgenaue Einschließung der Inversen $A^{-1} \in X$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Aufruf `INV(A,illcd,Alefeld,X,Err)` wird zu gegebener quadratischer Matrix **A** mit **X** eine Einschließung der Inversen $A^{-1} \in X$ berechnet.
`illcd=0`: LU–Zerlegung von **A**, Defektiteration und dann Verifikation, alle Rechnungen in einfacher Länge;
`illcd=1`: Verfahren von Rump, Defektiteration und Verifikation, alles in doppelter Länge;
`illcd=2`: Berechnung von **X** in *staggered arithmetic*
Die quadratischen Matrizen **A**,**X** müssen gleiche Dimensionen besitzen.

Weitere Informationen: Seite 84 in diesem Buch.

10.2.11 Berechnung einer Einschließung der Inversen bei gegebener Intervall–Matrix A durch: `INV(A,illcd,X,Err)`

Prozedurvereinbarung:

```
procedure INV ( var A     : Limatrix;
                    illcd : integer;
                var X     : Limatrix;
                var Err   : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Quadratische Intervall–Matrix, deren Inverse einzuschließen ist
illcd	Steuert den internen Algorithmus
Ausgangsparameter	
X	i.a. maximalgenaue Einschließung der Inversen $\boldsymbol{A}^{-1} \in \boldsymbol{X}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:

Mit dem Aufruf `INV(A,illcd,X,Err)` wird zu gegebener quadratischer Intervall–Matrix **A** eine Einschließung **X** der Inversen $\boldsymbol{A}^{-1} \in \boldsymbol{X}$ berechnet.

`illcd=0`: LU–Zerlegung von **A**, Defektiteration und dann Verifikation, alle Rechnungen in einfacher Länge;

`illcd=1`: Verfahren von Rump, Defektiteration und danach Verifikation, alle Rechnungen in doppelter Länge;

Die quadratischen Matrizen **A**,**X** müssen gleiche Dimensionen besitzen.

Weitere Informationen:

Siehe Seite 98; Zwei Programmbeispiele findet man auf den Seiten 98,101 dieses Buches.

10.2.12 Berechnung einer doppeltlangen Näherungsinversen mit *staggered arithmetic* durch: `INV_ST(A,R1,R2,Err)`

Prozedurvereinbarung:

```
procedure INV_ST ( var  A     : Lrmatrix;
                   var  R1,R2 : Lrmatrix;
                   var  Err   : integer );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende quadratische Punkt–Matrix
Ausgangsparameter	
R1,R2	i.a. maximalgenaue Näherungsinversen $\boldsymbol{R1} + \boldsymbol{R2} \approx \boldsymbol{A}^{-1}$
Err	Fehlerkode; Err=0: Keine Fehler aufgetreten

Arbeitsweise:
Mit dem Aufruf `INV_ST(A,R1,R2,Err)` wird zu gegebener quadratischer Punkt–Matrix **A** eine doppeltlange Näherung der Inversen $\boldsymbol{A}^{-1}$ wie folgt berechnet:
Berechnung einer ersten Näherung in *staggered arithmetic* und Rundung dieser Näherung ins doppeltlange Longreal–Format. Die Genauigkeit hängt ab vom Parameter `p`, der vorher mit Hilfe der Prozedur `set_Lstagg_prec(p)` aus dem Modul `type_def` zu wählen ist.
Die quadratischen Matrizen **A**,**R1**,**R2** müssen gleiche Dimensionen besitzen.

Weitere Informationen:

10.2.13 Verbesserung einer Einschließung der Inversen nach Alefeld/Herzberger mit: `ALEFELD_IT(A,X)`

Prozedurvereinbarung:

```
procedure ALEFELD_IT ( var A : Lrmatrix;
                       var X : Limatrix );
```

Eingangs-/Ausgangs–Parameter:

Eingangsparameter	
A	Zu invertierende reguläre, quadratische Punkt–Matrix
X	Einschließung der Inversen $\boldsymbol{A}^{-1}$
Ausgangsparameter	
X	Verbesserte Einschließung der Inversen $\boldsymbol{A}^{-1}$
Anmerkung:	
Die Eingangsgröße **X** wird durch die Ausgangsgröße **X** überschrieben	

Arbeitsweise:
Mit dem Prozeduraufruf `ALEFELD_IT(A,X)` wird eine schon existierende Einschließung **X** der Inversen $\boldsymbol{A}^{-1}$ i.a. verbessert. Die quadratischen Matrizen **A**, **X** müssen gleiche Dimensionen besitzen. $\boldsymbol{A}$ muß eine Punktmatrix sein.

Weitere Informationen:
Seite 87 in diesem Buch, [1, Abschnitt 18].

Literatur

[1] G. Alefeld and J. Herzberger. *Einführung in die Intervallrechnung.* Reihe Informatik 12. **BI** Bibliographisches Institut Mannheim Wien Zürich, 1974. ISBN 3-411-01466-0.

[2] Frithjof Blomquist. Verifizierende Numerik mit **PASCAL-XSC** BCD-Version Numerische Anwendungen Hilfsmodule Bibliothek I. Fachbereich 7, Mathematik, Bergische Universität Wuppertal, 2000. www.math.uni-wuppertal.de/org/WRST/blom.html.

[3] R. Hammer, M. Hocks, U. Kulisch, and D. Ratz. *Numerical Toolbox for Verified Computing, Basic Numerical Problems*, volume 1. Springer-Verlag Berlin Heidelberg, 1993. Springer Series in Computational Mathematics **21**
www.uni-karlsruhe.de/~iam/html/literatur/download.html

[4] W. Krämer, U. Kulisch, and R. Lohner. Numerical Toolbox for Verified Computing II. Veröffentlichung geplant im Springer Verlag.

[5] Ulrich Kulisch. *Wissenschaftliches Rechnen mit Ergebnisverifikation - Eine Einführung -*. Friedr. Vieweg & Sohn, Braunschweig/Wiesbaden, 1989. ISBN 3-528-08943-1.

[6] A. Neumaier. *Interval Methods for Systems of Equations.* Cambridge University Press, Cambridge, 1990.

[7] PASCAL-XSC. *Sprachbeschreibung mit Beispielen.* R.Klatte, U.Kulisch, M.Neaga, D.Ratz and Ch.Ullrich, Springer-Verlag Heidelberg, 1991. www.uni-karlsruhe.de/~iam/html/literatur/download.html

[8] PASCAL-XSC. *BCD-Version 1.0, Benutzerhandbuch für das dezimale Laufzeitsystem.* Frithjof Blomquist, Inst. f. Angewandte Mathematik, Universität Karlsruhe, 1997.
Forschungsschwerpunkt **C**omputerarithmetic, **I**ntervallrechnung und **N**umerische Algorithmen mit **E**rgebnisverifikation.
www.uni-karlsruhe.de/~iam/html/literatur/download.html

[9] S.M. Rump. Kleine Fehlerschranken bei Matrixproblemen. Dissertation, Universität Karlsruhe, 1980.

[10] G. Schulz. Iterative Berechnung der reziproken Matrix. *Z. Angew. Math. Mech.*, 13:57–59, 1933.

[11] H.R. Schwarz. *Numerische Mathematik.* B.G. Teubner, Stuttgart, 1997. ISBN 3-519-32960-3.

[12] Stephen Wolfram. *Mathematica, Ein System für Mathematik auf dem Computer, Zweite Auflage.* Addison Wesley Verlag (Deutschland) GmbH, 1992.

[13] Rudolf Zurmühl and Sigurd Falk. *Matrizen und ihre Anwendungen - für Angewandte Mathematiker, Physiker und Ingenieure*, volume 2 Numerische Methoden. Springer-Verlag, Berlin Heidelberg New York Tokyo, 1986. ISBN 3-540-15474-4.