

Xpert.press

Die Reihe **Xpert.press** vermittelt Professionals
in den Bereichen Softwareentwicklung,
Internettechnologie und IT-Management aktuell
und kompetent relevantes Fachwissen über
Technologien und Produkte zur Entwicklung
und Anwendung moderner Informationstechnologien.

Gisela Engeln-Müllges · Klaus Niederdrenk
Reinhard Wodicka

Numerik-Algorithmen

Verfahren, Beispiele, Anwendungen

Neunte, vollständig überarbeitete und erweiterte Auflage
mit zahlreichen Abbildungen und Beispielen
sowie 2 CD-ROMs



Prof. Dr. rer. nat. Gisela Engeln-Müllges
Fachbereich Maschinenbau und Mechatronik
Fachhochschule Aachen
Goethestraße 1
52064 Aachen
engeln-muellges@fh-aachen.de

Prof. Dr. rer. nat. Klaus Niederdrenk
Fachbereich Chemieingenieurwesen
Fachhochschule Münster
Stegerwaldstraße 39
48565 Steinfurt
niederdrenk@fh-muenster.de

Stud. Prof. Dr. rer. nat. Reinhard Wodicka
Am Kupferofen 34
52066 Aachen

Bibliografische Information der Deutschen Bibliothek
Die Deutsche Bibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie;
detaillierte bibliografische Daten sind im Internet über
<http://dnb.ddb.de> abrufbar.

Die Neuauflage fasst die 5. Auflage von „Numerische Mathematik für Ingenieure“ von Gisela Engeln-Müllges und Fritz Reutter (ursprünglich Bibliographisches Institut/Brockhaus AG, Mannheim) und die 8. Auflage von „Numerik-Algorithmen“ von Gisela Engeln-Müllges und Fritz Reutter (ursprünglich VDI-Verlag, Düsseldorf) zusammen.

ISSN 1439-5428

ISBN 3-540-62669-7 Springer Berlin Heidelberg New York

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

Springer ist nicht Urheber der Daten und Programme. Weder Springer noch die Autoren übernehmen die Haftung für die CD-ROMs und das Buch, einschließlich ihrer Qualität, Handels- und Anwendungseignung. In keinem Fall übernehmen Springer oder die Autoren Haftung für direkte, indirekte, zufällige oder Folgeschäden, die sich aus der Nutzung der CD-ROMs oder des Buches ergeben.

Springer ist ein Unternehmen von Springer Science+Business Media
springer.de

© Springer-Verlag Berlin Heidelberg 2005
Printed in Germany

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, dass solche Namen im Sinne der Warenzeichen- und Markenschutzgesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften. Text und Abbildungen wurden mit größter Sorgfalt erarbeitet. Verlag und Autor können jedoch für eventuell verbliebene fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Satz: Druckfertige Daten der Autoren
Herstellung: LE-TeX Jelonek, Schmidt & Vöckler GbR, Leipzig
Umschlaggestaltung: KunkelLopka Werbeagentur, Heidelberg
Gedruckt auf säurefreiem Papier 33/3142/YL - 5 4 3 2 1 0

*Im Gedenken an unseren akademischen Lehrer
Prof. Dr. rer. techn. Fritz Reutter (1911–1990)
Rheinisch-Westfälische Technische Hochschule Aachen*

Vorwort zur 9. Auflage

Das vorliegende Werk steht in einer langen Tradition. Die erste Auflage erschien 1974 mit den Autoren Gisela Engeln-Müllges und Fritz Reutter unter dem Titel „Formelsammlung zur Numerischen Mathematik mit FORTRAN-Programmen“. Der Inhalt des Buches orientierte sich an Vorlesungen über Numerische Mathematik für Studierende der Ingenieurwissenschaften.

Bei den nachfolgenden Auflagen, an denen Fritz Reutter bis zu seiner schweren Erkrankung 1983 mitarbeitete, nahm der Umfang des Buches stark zu, und die Programm-Anhänge wurden um die Sprachen Turbo Pascal, C, Modula 2 und Quick-Basic erweitert. Dies hing zusammen mit der raschen Entwicklung der Computertechnologie und ihren Impulsen auf die Numerische Mathematik, die zu einer enormen qualitativen wie auch quantitativen Zunahme numerischer Verfahren führte.

Grundlage des vorliegenden Buches ist die achte Auflage des Buches von Gisela Engeln-Müllges und Fritz Reutter mit dem Titel „Numerik-Algorithmen“ (VDI-Verlag, 1996), die erstmals eine CD-ROM mit Fortran 77/90-, Turbo Pascal- und ANSI C-Programmen enthielt.

Wie bisher wird auch in dieser neunten Auflage von den unten genannten Verfassern besonderer Wert gelegt auf die Erläuterung der Prinzipien der behandelten Verfahren der Numerischen Mathematik, auf die exakte Beschreibung leistungsfähiger Algorithmen und die Entwicklung und Dokumentation zugehöriger Programme. Die ausführliche Darstellung der mathematischen Grundlagen, ergänzt durch viele Abbildungen und durchgerechnete Beispiele, soll den Zugang zur Numerischen Mathematik erleichtern und das Verständnis des algorithmischen Vorgehens fördern. Zahlreiche Beispiele aus ingenieurwissenschaftlichen Anwendungen belegen die Notwendigkeit der behandelten numerischen Verfahren.

Das Buch wendet sich in erster Linie an Ingenieure sowie an Naturwissenschaftler und Informatiker in Studium und Beruf, ferner an Lehrkräfte für mathematisch-naturwissenschaftlichen Unterricht. Deshalb ist besonders auf die Verwendbarkeit der behandelten Themen in der Praxis geachtet worden.

Beigefügt ist dem Buch eine CD-ROM, die umfassend getestete C-Programme zu nahezu allen angegebenen Algorithmen enthält, sowie weitere Software, zu der Informationen auf den folgenden Seiten zu finden sind.

Die inhaltliche Erweiterung des Buches erfordert es, die bisher in ihm enthaltenen numerischen Verfahren für Anfangs- und Randwertprobleme bei gewöhnlichen Differentialgleichungen in einen weiteren Band zu verlagern. Dieser wird, einem häufig geäußerten Wunsch folgend, zusätzlich Numerik partieller Differentialgleichungen sowie wichtige stochastische Methoden, insbesondere statistische Schätz- und Prüfverfahren enthalten und ebenfalls bei Springer erscheinen. Verfasser dieses Buches sind Gisela Engeln-Müllges, Klaus Niederdrenk und Wieland Richter.

Ganz besonders herzlich danken wir den Autoren der Programme und ebenso Doris und Uli Eggermann, die mit Hilfe des Satzprogramms $\text{\LaTeX}$ das reproduktionsreife Manuskript sowie eine Vielzahl der Abbildungen mit äußerstem Engagement und sehr großem Geschick fertig gestellt haben. Darüber hinaus hat uns Uli Eggermann tatkräftig beim Korrekturlesen und beim Rechnen der Beispiele zur Kubatur unterstützt.

Ein herzlicher Dank gilt nicht zuletzt Herrn Hermann Engesser vom Springer-Verlag für die hervorragende und effektive Zusammenarbeit.

Aachen, Münster, Juli 2004

Gisela Engeln-Müllges
Klaus Niederdrenk
Reinhard Wodicka

Informationen zur beigefügten Software (CD-1, CD-2)

Dem Buch sind zwei CDs beigefügt, welche unterschiedliche Demonstrationsprogramme enthalten sowie ANSI-C-Quellen zur Verwendung in selbstgeschriebenen Programmen.

Informationen zur CD-ROM mit C-Programmen u.a. (CD-1)

Auf dieser CD befinden sich Ansi-C-Quellen von Unterprogrammen zu den meisten der im Buch angegebenen Algorithmen. Diese Quellen können compilerunabhängig in eigenen C-Programmen verwendet werden. Mittels beigefügter Makefile-Datei ist man in der Lage,

- gezielt einzelne Module zu übersetzen
- eine eigene Bibliothek zusammenzustellen, die zu selbstgeschriebenen Programmen hinzugelinkt werden kann
- spezielle Testprogramme zu den Unterprogrammen zu erstellen und mit geeigneten Testdatensätzen ablaufen zu lassen.

Weitergehende Informationen zur Verwendung der Ansi-C-Quellen und Antworten zu Compiler- und Makefile-Fragen sind auf der CD in der Datei [ReadMe.htm](#) und in der DMAKE-Datei [Makefile.mk](#) angegeben.

Systemvoraussetzungen

Die Unterprogramme wurden unter verschiedenen Betriebssystemen getestet, u.a. MS-DOS, Windows (95, 98, 2000, NT), OS/2, TOS 4.04, UNIX, Linux. Dabei wurden diverse C-Compiler verwendet.

C++ -Programme und Campuslizenzen

Die C-Programme der CD-ROM sind bei der FH-Aachen auch in C++ erhältlich. Informationen über Lizenzen zu den C++-Programmen sowie über Campuslizenzen zu den C- und C++-Programmen können per e-Mail angefordert werden:

engeln-muellges@fh-aachen.de

Weitere Software auf der CD-1

Auf der CD-1 sind noch folgende Programme zu finden:

- CurveTrac
- CurveView
- Interaktive Lehrunterstützung

Zu allen drei Programmen finden Sie Informationen in den folgenden Abschnitten. In der HTML-Datei *ReadMe* auf der CD sind sie ebenfalls kurz beschrieben und können (unter Windows) direkt aus dem Browser (z.B. IE, Netscape, Mozilla, Opera) heraus gestartet werden.

Systemvoraussetzungen

Betriebssystem:	Windows (98, 2000 oder XP)
Arbeitsspeicher:	mindestens 256 MB RAM
Browser:	Netscape ab 6.2, Internet Explorer ab 5.5, Mozilla ab 1.4
Java:	aktivieren (ab Version 1.2); wenn von CD gestartet (aus ReadMe.htm), wird mitgeliefertes Java verwendet.

Informationen zum Expertensystem „CurveTrac“ (CD-1)

CurveTrac ist ein Baukastensystem von numerischen Anwendungen, die auf den C-Programmen aus diesem Buch basieren. Die Benutzer-Oberfläche ist in Java programmiert. Dadurch ist gewährleistet, dass die Anwendung auf den gängigsten Betriebssystemen genutzt werden kann. Getestet wurde CurveTrac unter Windows und Linux. In der hier vorliegenden Ausbaustufe ist das Modul „Höhenlinien“ eingebunden. Informationen über weitere Module können Sie bei engeln-muellges@fh-aachen.de anfordern. Dies sind im Einzelnen:

- Expertensystem zur Numerischen Mathematik
- Berechnung der Schnittkurve zweier Flächen im Raum
- Splinemodul (Flächensplines und Kurvensplines)
- u. a. m. . . .

Beschreibung des Moduls „Höhenlinien“

Mit dem Modul „Höhenlinien“ können beliebig angeordnete Wertetripel $(x_i, y_i, z_i = f(x_i, y_i))$, z. B. Messdaten, eingegeben werden. Die Funktion $f(x, y)$ wird näherungsweise mittels eines zweidimensionalen Oberflächenspline berechnet und als dreidimensionale Grafik dargestellt. Neben dieser Funktionalität ist es zusätzlich möglich, beliebige Schnitte parallel zur x, y -Ebene zu berechnen und das daraus erzeugte Höhenliniendiagramm in einer zweidimensionalen Grafik darzustellen. Bei Bedarf kann das Höhenliniendiagramm mit frei definierbaren Farben ausgefüllt werden.

CurveTrac mit dem Modul Höhenlinien ist auf der CD-ROM mit den C-Programmen zu finden.

Programmiert wurde CurveTrac von Dominikus Bartusch, Frank Hähling und Thomas Layh.

Informationen zu dem Modul „CurveView“ (CD-1)

CurveView ist wie CurveTrac ein in Java programmiertes Baukastensystem ohne Benutzeroberfläche. Mittels in XML-Dateien abgelegten numerischen Berechnungsvorschriften (basierend auf den C-Programmen aus diesem Buch) können die Ergebnisse einer Berechnung als zwei- bzw. dreidimensionale Grafik bereitgestellt werden. In der hier vorliegenden Ausbaustufe ist das Berechnungsmodul „Kurvensplines“ mit vielen Beispielen eingebunden. Wenn Interesse an weiteren Modulen besteht, z. B.

- Flächensplines
- Nullstellenverfahren
- Differentialgleichungen
- Matrizen
- Statistik
- u. a. m. . . .

senden Sie eine Nachricht an engeln-muellges@fh-aachen.de.
 Programmiert wurde CurveView von Stefan Kleemann.

Informationen zum Demo-Framework „Interaktive Lehrunterstützung“ auf der CD-ROM (CD-1)

Mit dem Framework können die Lernenden Themengebiete aus Lehrveranstaltungen selbstständig mit interaktiven Elementen vertiefen. Es gibt eine Auswahl von über 100 Interaktionen und Animationen, die gezielt zu speziellen Themengebieten zusammengestellt und dem Lernenden zur Verfügung gestellt werden können. In dieser Demo-Version sind zu jedem der folgenden Themengebiete beispielhaft Interaktionen und Animationen vorhanden:

- Quadratur
- Nullstellen
- Splines
- Differentialgleichungen

Wenn Interesse an weiteren Interaktionen und Animationen besteht, können Informationen unter der Mail-Adresse engeln-muellges@fh-aachen.de angefordert werden.
 Programmiert wurde das Framework von Michael Neßlinger.

Informationen zur Demo-CD-ROM „NUMAS“ (CD-2)

Die beiliegende Demo-CD-ROM enthält das Lernfeld „Kurvensplines“ aus dem multimedialen Lehr- und Lernsystem NUMAS zur Numerischen Mathematik und Statistik. Der

Inhalt der Demo-CD-ROM entspricht den Kapiteln über Kurvensplines in diesem Buch und informiert auch über den Gesamthalt des Lernsystems.

NUMAS bietet didaktisch aufbereitetes Wissen zur Numerischen Mathematik und Statistik. Es werden Inhalte angeboten, die für die Hochschulausbildung vieler Fachrichtungen grundlegend sind und die sich gleichzeitig hervorragend dazu eignen, in Formen des angeleiteten Selbststudiums aufgenommen zu werden. Lernerinnen und Lerner werden zeitgemäß in ihren Selbstlern- und Selbstorganisationskompetenzen gefördert.

Für die Vollversion NUMAS wurden in großen Teilen die Inhalte dieses Buches und des im Vorwort angekündigten weiteren Bandes mit Differentialgleichungen und Statistik verwendet.

NUMAS ist aus einem vom BMBF und vom Land NRW geförderten Projekt innerhalb der Ausschreibungen „Neue Medien in der Bildung“ und „Neue Medien in der Hochschullehre“ hervorgegangen. An dem Projekt waren die Fachhochschule Aachen (Prof. Dr. Engeln-Müllges), die Freie Universität Berlin (Prof. Dr. Martus), die Fachhochschule Münster (Prof. Dr. Niederdrenk) und die Fachhochschule Südwestfalen (Prof. Dr. Richter) beteiligt.

NUMAS ist auch im Internet unter <http://www.numas.de/> zu erreichen.

Der Umgang mit der CD-ROM selbst ist denkbar einfach: Nach dem Einlegen der CD-ROM in das Laufwerk erscheint selbstständig ein Auswahlbildschirm mit den im Gesamtsystem zur Verfügung stehenden Lernfeldern. Von hier aus kann man sich den Inhalt ansehen und bearbeiten. Bei der Demo-CD-ROM ist nur das Lernfeld Kurvensplines freigeschaltet.

Die CD-ROM stellt nur einen Teil der Funktionalität des Lernsystems zur Verfügung. Zum Beispiel sind die Kommunikationswerkzeuge, die an das Internet gebunden sind, nur im Online-System verfügbar.

Systemvoraussetzungen

Ihr Computersystem sollte folgende Bedingungen erfüllen, um einen reibungslosen Betrieb von NUMAS erreichen zu können:

Betriebssystem:	Windows (98, 2000 oder XP)
Browser:	Netscape ab 6.2, Internet Explorer ab 5.5, Mozilla ab 1.4
Javascript:	aktivieren (ab Version 1.2)
Cookies:	aktivieren

Um auch die interaktiven und multimedialen Elemente von NUMAS nutzen zu können, müssen zusätzlich folgende Plug-Ins installiert sein:

Macromedia Flash Player und SUN Java (JRE) ab Version 1.4 (mit Ausnahme der Version 1.4.1_03)

Weitere Informationen zu NUMAS können Sie unter der Mail-Adresse info@numas.de anfordern.

Bezeichnungen

$\Rightarrow$	wenn – dann bzw. hat zur Folge
$\Leftrightarrow$	dann und nur dann
$a := b$	a wird definiert durch b
$\stackrel{!}{=}$	geforderte Gleichheit
$< \leq$	kleiner, kleiner oder gleich
$> \geq$	größer, größer oder gleich
$a << b$	a ist wesentlich kleiner als b
$\approx$	ungefähr gleich
$\equiv$	identisch
$\sim$	proportional bzw. gleichmäßig zu
$\{a_1, a_2, \dots\}$	Menge aus den Elementen $a_1, a_2, \dots$
$\{x \dots\}$	Menge aller x für die gilt $\dots$
$\in$	Element von
$\notin$	nicht Element von
$\subseteq$	enthalten in oder Teilmenge von
$\subset$	echt enthalten in oder echte Teilmenge von
$\not\subset$	nicht Teilmenge von
$\mathbb{N}$	Menge der natürlichen Zahlen
$\mathbb{N}_0$	Menge der natürlichen Zahlen mit Null
$\mathbb{Z}$	Menge der ganzen Zahlen
$\mathbb{Q}$	Menge der rationalen Zahlen
$\mathbb{R}$	Menge der reellen Zahlen
$\mathbb{C}$	Menge der komplexen Zahlen
$\mathbb{R}^+, \mathbb{R}^-$	Menge der positiven bzw. negativen reellen Zahlen
(a, b)	offenes Intervall von a bis b , $a < b$
$[a, b]$	abgeschlossenes Intervall von a bis b , $a < b$
$[a, b)$	halboffenes Intervall von a bis b (rechts offen), $a < b$
$(a, b]$	halboffenes Intervall von a bis b (links offen), $a < b$
$n!$	n Fakultät mit $n! = 1 \cdot 2 \cdot 3 \cdots n$, $n \in \mathbb{N}$, $0! := 1$
$\binom{n}{k}$	n über k mit $\binom{n}{k} := \frac{n!}{k!(n-k)!}$, $k \in \mathbb{N}_0$, $k \leq n$, $n \in \mathbb{N}$
$\prod_{i=1}^n a_i$	$a_1 \cdot a_2 \cdot a_3 \cdots a_n$
$\sum_{i=1}^n a_i$	$a_1 + a_2 + \cdots + a_n$
$\int_a^b$	Integral in den Grenzen a und b

XIV Bezeichnungen

i	imaginäre Einheit $i := \sqrt{-1}$
e	Eulersche Zahl $= 2.718\,281\,828\,459\dots$
$ a $	Betrag von a mit $ a := \begin{cases} a & \text{für } a \geq 0 \\ -a & \text{für } a < 0 \end{cases}$
$\ \cdot\ $	Norm von $\cdot$
$\{a_k\}$	Folge von a_k
$\lim_{k \rightarrow \infty} a_k$	Limes von a_k für $k \rightarrow \infty$
$\max \{f(x) x \in [a, b]\}$	Maximum aller Funktionswerte $f(x)$ für $x \in [a, b]$
$\min \{ M_i \mid i = 1, 2, \dots, m\}$	Minimum aller $ M_i $ für $i = 1, 2, \dots, m$
(x, y)	geordnetes Paar
$(x_1, x_2, \dots, x_n)$	geordnetes n -Tupel
$f: I \rightarrow \mathbf{R}$	Abbildung f von I nach $\mathbf{R}$
$x \mapsto f(x), x \in D$	x wird $f(x)$ zugeordnet für $x \in D$
$f', f'', f''', f^{(4)}, \dots, f^{(n)}$	erste, zweite, dritte, vierte, $\dots$, n -te Ableitung von f
$C[a, b]$	die Menge der auf $[a, b]$ stetigen Funktionen
$C^n[a, b]$	die Menge der auf $[a, b]$ n -mal stetig differenzierbaren Funktionen
$\mathbf{R}^n$	n -dimensionaler euklidischer Raum
$A = O(h^q)$	$ A/h^q \leq C$ für $h \rightarrow 0$, $C = \text{const.}$
$i = m(1)n$	$m, n \in \mathbf{Z}, \quad m \leq n, \quad i = m, m+1, \dots, n$
$\text{sign}(a), \text{sgn}(a)$	$\begin{cases} 1 & \text{für } a > 0 \\ 0 & \text{für } a = 0 \\ -1 & \text{für } a < 0 \end{cases}$
$\mathbf{x}, \mathbf{y}, \mathbf{z}, \dots$	Vektoren
$\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$	Matrizen
$\mathbf{0}$	Nullmatrix bzw. Nullvektor
$\mathbf{x} \times \mathbf{y}$	Vektorprodukt bzw. Kreuzprodukt
$\mathbf{E}$	Einheitsmatrix
$\mathbf{A}^\top$	transponierte Matrix von $\mathbf{A}$
$\mathbf{x}^\top = (x_1, x_2, \dots, x_n)$	transponierter Vektor zu $\mathbf{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$
$\mathbf{A}^{-1}$	inverse Matrix von $\mathbf{A}$
$ \mathbf{A} , \det(\mathbf{A})$	Determinante von $\mathbf{A}$
[Verweis]	s. im Literaturverzeichnis unter [Verweis]
o. B. d. A.	ohne Beschränkung der Allgemeinheit
$\square$	Ende eines Beispiels oder eines Beweises

Inhaltsverzeichnis

Vorwort zur 9. Auflage	VII
Informationen zur beigefügten Software (CD-1, CD-2)	IX
1 Darstellung von Zahlen und Fehleranalyse	1
1.1 Definition von Fehlergrößen	1
1.2 Zahlensysteme	3
1.2.1 Darstellung ganzer Zahlen	3
1.2.2 Darstellung reeller Zahlen	6
1.3 Rechnung mit endlicher Stellenzahl	11
1.4 Fehlerquellen	17
1.4.1 Eingabefehler	17
1.4.2 Verfahrensfehler	18
1.4.3 Fehlerfortpflanzung und die Kondition eines Problems	19
1.4.4 Rechnungsfehler und numerische Stabilität	24
2 Lösung nichtlinearer Gleichungen	27
2.1 Aufgabenstellung und Motivation	27
2.2 Definitionen und Sätze über Nullstellen	29
2.3 Allgemeines Iterationsverfahren	31
2.3.1 Konstruktionsmethode und Definition	31
2.3.2 Existenz einer Lösung und Eindeutigkeit der Lösung	34
2.3.3 Konvergenz eines Iterationsverfahrens	37
2.3.3.1 Heuristische Betrachtungen	37
2.3.3.2 Analytische Betrachtung	39
2.3.4 Fehlerabschätzungen und Rechnungsfehler	40
2.3.5 Praktische Durchführung	46
2.4 Konvergenzordnung eines Iterationsverfahrens	49
2.5 Newtonsche Verfahren	51
2.5.1 Das Newtonsche Verfahren für einfache Nullstellen	51
2.5.2 Gedämpftes Newton-Verfahren	57
2.5.3 Das Newtonsche Verfahren für mehrfache Nullstellen – Das modifizierte Newtonsche Verfahren	57
2.6 Das Sekantenverfahren	63
2.6.1 Das Sekantenverfahren für einfache Nullstellen	63
2.6.2 Das modifizierte Sekantenverfahren für mehrfache Nullstellen	66
2.7 Einschlussverfahren	66

2.7.1	Das Prinzip der Einschlussverfahren	67
2.7.2	Das Bisektionsverfahren	69
2.7.3	Die Regula falsi	71
2.7.4	Das Pegasus-Verfahren	74
2.7.5	Das Verfahren von Anderson-Björck	77
2.7.6	Die Verfahren von King und Anderson-Björck-King – Das Illinois-Verfahren	80
2.7.7	Ein kombiniertes Einschlussverfahren	81
2.7.8	Das Zeroin-Verfahren	83
2.8	Anwendungsbeispiele	85
2.9	Effizienz der Verfahren und Entscheidungshilfen	89
3	Verfahren zur Lösung algebraischer Gleichungen	91
3.1	Vorbemerkungen	91
3.2	Das Horner-Schema	92
3.2.1	Das einfache Horner-Schema für reelle Argumentwerte	93
3.2.2	Das einfache Horner-Schema für komplexe Argumentwerte	95
3.2.3	Das vollständige Horner-Schema für reelle Argumentwerte	97
3.2.4	Anwendungen	100
3.3	Bestimmung von Lösungen algebraischer Gleichungen	101
3.3.1	Vorbemerkungen und Überblick	101
3.3.2	Das Verfahren von Muller	102
3.3.3	Das Verfahren von Bauhuber	109
3.3.4	Das Verfahren von Jenkins und Traub	111
3.4	Anwendungsbeispiel	112
3.5	Entscheidungshilfen	113
4	Lösung linearer Gleichungssysteme	115
4.1	Aufgabenstellung und Motivation	115
4.2	Definitionen und Sätze	120
4.3	Lösbarkeitsbedingungen für ein lineares Gleichungssystem	132
4.4	Prinzip der direkten Methoden zur Lösung linearer Gleichungssysteme	133
4.5	Der Gauß-Algorithmus	136
4.5.1	Gauß-Algorithmus mit Spaltenpivotsuche als Rechenschema	136
4.5.2	Spaltenpivotsuche	141
4.5.3	Gauß-Algorithmus als Dreieckszerlegung	145
4.5.4	Gauß-Algorithmus für Systeme mit mehreren rechten Seiten	149
4.6	Matrizeninversion mit dem Gauß-Algorithmus	151
4.7	Verfahren für Systeme mit symmetrischen Matrizen	153
4.7.1	Systeme mit symmetrischer, streng regulärer Matrix	154
4.7.2	Systeme mit symmetrischer, positiv definiter Matrix – Cholesky-Verfahren	155
4.7.3	Systeme mit symmetrischer, positiv definiter Matrix – Verfahren der konjugierten Gradienten (CG-Verfahren)	160
4.8	Das Gauß-Jordan-Verfahren	164
4.9	Gleichungssysteme mit tridiagonaler Matrix	165
4.9.1	Systeme mit tridiagonaler Matrix	165
4.9.2	Systeme mit symmetrischer, tridiagonaler, positiv definiter Matrix	169

4.10	Gleichungssysteme mit zyklisch tridiagonaler Matrix	172
4.10.1	Systeme mit zyklisch tridiagonaler Matrix	172
4.10.2	Systeme mit symmetrischer, zyklisch tridiagonaler Matrix	175
4.11	Gleichungssysteme mit fünfdiagonaler Matrix	177
4.11.1	Systeme mit fünfdiagonaler Matrix	177
4.11.2	Systeme mit symmetrischer, fünfdiagonaler, positiv definiter Matrix	180
4.12	Gleichungssysteme mit Bandmatrix	183
4.13	Householdertransformation	194
4.14	Fehler, Kondition und Nachiteration	199
4.14.1	Fehler und Kondition	199
4.14.2	Konditionsschätzung	203
4.14.3	Möglichkeiten zur Konditionsverbesserung	208
4.14.4	Nachiteration	208
4.15	Gleichungssysteme mit Blockmatrix	210
4.15.1	Vorbemerkungen	210
4.15.2	Gauß-Algorithmus für Blocksysteme	211
4.15.3	Gauß-Algorithmus für tridiagonale Blocksysteme	213
4.15.4	Weitere Block-Verfahren	214
4.16	Algorithmus von Cuthill-McKee	215
4.17	Entscheidungshilfen	219
5	Iterationsverfahren zur Lösung linearer Gleichungssysteme	223
5.1	Vorbemerkungen	223
5.2	Vektor- und Matrizennormen	223
5.3	Das Iterationsverfahren in Gesamtschritten	225
5.4	Das Gauß-Seidelsche Iterationsverfahren	234
5.5	Relaxation beim Gesamtschrittverfahren	236
5.6	Relaxation beim Einzelschrittverfahren – SOR-Verfahren	236
5.6.1	Schätzung des Relaxationskoeffizienten – Adaptives SOR-Verfahren	237
6	Systeme nichtlinearer Gleichungen	241
6.1	Aufgabenstellung und Motivation	241
6.2	Allgemeines Iterationsverfahren für Systeme	244
6.3	Spezielle Iterationsverfahren	250
6.3.1	Newtonsche Verfahren für nichtlineare Systeme	250
6.3.1.1	Das quadratisch konvergente Newton-Verfahren	250
6.3.1.2	Gedämpftes Newton-Verfahren für Systeme	253
6.3.2	Sekantenverfahren für nichtlineare Systeme	254
6.3.3	Das Verfahren des stärksten Abstiegs (Gradientenverfahren) für nichtlineare Systeme	255
6.3.4	Das Verfahren von Brown für Systeme	257
6.4	Entscheidungshilfen	258

7	Eigenwerte und Eigenvektoren von Matrizen	259
7.1	Definitionen und Aufgabenstellungen	259
7.2	Diagonalähnliche Matrizen	260
7.3	Das Iterationsverfahren nach v. Mises	262
7.3.1	Bestimmung des betragsgrößten Eigenwertes und des zugehörigen Eigenvektors	262
7.3.2	Bestimmung des betragskleinsten Eigenwertes	269
7.3.3	Bestimmung weiterer Eigenwerte und Eigenvektoren	269
7.4	Konvergenzverbesserung	271
7.5	Das Verfahren von Krylov	272
7.5.1	Bestimmung der Eigenwerte	272
7.5.2	Bestimmung der Eigenvektoren	274
7.6	QD-Algorithmus	275
7.7	Transformationen auf Hessenbergform	276
7.7.1	Transformation einer Matrix auf obere Hessenbergform	276
7.7.2	LR-Verfahren	280
7.7.3	QR-Verfahren	282
7.8	Verfahren von Martin, Parlett, Peters, Reinsch und Wilkinson	283
7.9	Entscheidungshilfen	284
7.10	Anwendungsbeispiel	285
8	Lineare und nichtlineare Approximation	291
8.1	Aufgabenstellung und Motivation	291
8.2	Lineare Approximation	294
8.2.1	Approximationsaufgabe und beste Approximation	294
8.2.2	Kontinuierliche lineare Approximation im quadratischen Mittel	296
8.2.3	Diskrete lineare Approximation im quadratischen Mittel	302
8.2.3.1	Normalgleichungen für den diskreten linearen Ausgleich	302
8.2.3.2	Diskreter Ausgleich durch algebraische Polynome unter Verwendung orthogonaler Polynome	308
8.2.3.3	Lineare Regression – Ausgleich durch lineare algebraische Polynome	310
8.2.3.4	Householder-Transformation zur Lösung des linearen Ausgleichsproblems	313
8.2.4	Approximation von Polynomen durch Tschebyscheff-Polynome	316
8.2.4.1	Beste gleichmäßige Approximation, Definition	316
8.2.4.2	Approximation durch Tschebyscheff-Polynome	317
8.2.5	Approximation periodischer Funktionen	323
8.2.5.1	Kontinuierliche Approximation periodischer Funktionen im quadratischen Mittel	324
8.2.5.2	Diskrete Approximation periodischer Funktionen im quadratischen Mittel	326
8.2.5.3	Fourier-Transformation und FFT	329
8.2.6	Fehlerabschätzungen für lineare Approximationen	336
8.2.6.1	Gleichmäßige Approximation durch algebraische Polynome	337

8.2.6.2	Gleichmäßige Approximation durch trigonometrische Polynome	340
8.3	Diskrete nichtlineare Approximation	342
8.3.1	Transformationsmethode beim nichtlinearen Ausgleich	342
8.3.2	Nichtlinearer Ausgleich im quadratischen Mittel	348
8.4	Entscheidungshilfen	348
9	Polynomiale Interpolation sowie Shepard-Interpolation	351
9.1	Aufgabenstellung	351
9.2	Interpolationsformeln von Lagrange	353
9.2.1	Lagrangesche Formel für beliebige Stützstellen	353
9.2.2	Lagrangesche Formel für äquidistante Stützstellen	355
9.3	Aitken-Interpolationsschema für beliebige Stützstellen	356
9.4	Inverse Interpolation nach Aitken	360
9.5	Interpolationsformeln von Newton	362
9.5.1	Newtonsche Formel für beliebige Stützstellen	362
9.5.2	Newtonsche Formel für äquidistante Stützstellen	365
9.6	Abschätzung und Schätzung des Interpolationsfehlers	368
9.7	Zweidimensionale Interpolation	373
9.7.1	Zweidimensionale Interpolationsformel von Lagrange	374
9.7.2	Shepard-Interpolation	376
9.8	Entscheidungshilfen	385
10	Interpolierende Polynom-Splines zur Konstruktion glatter Kurven	387
10.1	Polynom-Splines dritten Grades	387
10.1.1	Aufgabenstellung	390
10.1.2	Woher kommen Splines? Mathematische Analyse	395
10.1.3	Anwendungsbeispiele	397
10.1.4	Definition verschiedener Arten nichtparametrischer kubischer Splinefunktionen	402
10.1.5	Berechnung der nichtparametrischen kubischen Splines	408
10.1.6	Berechnung der parametrischen kubischen Splines	425
10.1.7	Kombinierte interpolierende Polynom-Splines	433
10.1.8	Näherungsweise Ermittlung von Randableitungen durch Interpolation	438
10.1.9	Konvergenz und Fehlerabschätzungen interpolierender kubischer Splines	440
10.2	Hermite-Splines fünften Grades	442
10.2.1	Definition der nichtparametrischen und parametrischen Hermite-Splines	442
10.2.2	Berechnung der nichtparametrischen Hermite-Splines	443
10.2.3	Berechnung der parametrischen Hermite-Splines	447
10.3	Polynomiale kubische Ausgleichssplines	452
10.3.1	Aufgabenstellung und Motivation	452
10.3.2	Konstruktion der nichtparametrischen Ausgleichssplines	456
10.3.3	Berechnung der parametrischen kubischen Ausgleichssplines	464
10.4	Entscheidungshilfen für die Auswahl einer geeigneten Splinemethode	465

11 Akima- und Renner-Subsplines	471
11.1 Akima-Subsplines	471
11.2 Renner-Subsplines	478
11.3 Abrundung von Ecken bei Akima- und Renner-Kurven	488
11.4 Berechnung der Länge einer Kurve	492
11.5 Flächeninhalt einer geschlossenen ebenen Kurve	495
11.6 Entscheidungshilfen	498
12 Spezielle Splines	499
12.1 Interpolierende zweidimensionale Polynom-Splines	499
12.2 Zweidimensionale interpolierende Oberflächensplines	513
12.3 Bézier Splines	516
12.3.1 Bézier-Spline-Kurven	517
12.3.2 Bézier-Spline-Flächen	521
12.3.3 Modifizierte (interpolierende) kubische Bézier-Splines	529
12.4 B-Splines	530
12.4.1 B-Spline-Kurven	530
12.4.2 B-Spline-Flächen	536
12.5 Anwendungsbeispiel	541
12.6 Entscheidungshilfen	546
13 Numerische Differentiation	549
13.1 Aufgabenstellung und Motivation	549
13.2 Differentiation mit Hilfe eines Interpolationspolynoms	550
13.3 Differentiation mit Hilfe interpolierender kubischer Polynom-Splines	553
13.4 Differentiation mit dem Romberg-Verfahren	555
13.5 Entscheidungshilfen	559
14 Numerische Quadratur	561
14.1 Vorbemerkungen	561
14.2 Konstruktion von Interpolationsquadraturformeln	564
14.3 Newton-Cotes-Formeln	567
14.3.1 Die Sehnentrapezformel	569
14.3.2 Die Simpsonsche Formel	574
14.3.3 Die 3/8-Formel	579
14.3.4 Weitere Newton-Cotes-Formeln	582
14.3.5 Zusammenfassung zur Fehlerordnung von Newton-Cotes-Formeln	586
14.4 Quadraturformeln von Maclaurin	586
14.4.1 Die Tangententrapezformel	587
14.4.2 Weitere Maclaurin-Formeln	589
14.5 Die Euler-Maclaurin-Formeln	591
14.6 Tschebyscheffsche Quadraturformeln	593
14.7 Quadraturformeln von Gauß	595
14.8 Verallgemeinerte Gauß-Quadraturformeln	599
14.9 Quadraturformeln von Clenshaw-Curtis	602
14.10 Das Verfahren von Romberg	603
14.11 Fehlerschätzung und Rechnungsfehler	608
14.12 Adaptive Quadraturverfahren	610

14.13	Konvergenz der Quadraturformeln	612
14.14	Anwendungsbeispiel	613
14.15	Entscheidungshilfen	614

15 Numerische Kubatur 617

15.1	Problemstellung	617
15.2	Konstruktion von Interpolationskubaturformeln	619
15.3	Newton-Cotes-Kubaturformeln für Rechteckbereiche	622
15.4	Das Romberg-Kubaturverfahren für Rechteckbereiche	630
15.5	Gauß-Kubaturformeln für Rechteckbereiche	633
15.6	Riemannsche Flächenintegrale	636
15.7	Vergleich der Verfahren anhand von Beispielen	636
15.8	Kubaturformeln für Dreieckbereiche	641
15.8.1	Kubaturformeln für Dreieckbereiche mit achsenparallelen Katheten	641
15.8.1.1	Newton-Cotes-Kubaturformeln für Dreieckbereiche	641
15.8.1.2	Gauß-Kubaturformeln für Dreieckbereiche mit achsenparallelen Katheten	644
15.8.2	Kubaturformeln für Dreieckbereiche allgemeiner Lage	648
15.8.2.1	Newton-Cotes-Kubaturformeln für Dreieckbereiche allgemeiner Lage	649
15.8.2.2	Gauß-Kubaturformeln für Dreieckbereiche allgemeiner Lage	652
15.9	Entscheidungshilfen	655

Sachwortverzeichnis 669

Kapitel 1

Darstellung von Zahlen und Fehleranalyse, Kondition und Stabilität

1.1 Definition von Fehlergrößen

Ein numerisches Verfahren liefert im Allgemeinen anstelle einer gesuchten Zahl a nur einen Näherungswert A für diese Zahl a . Zur Beschreibung dieser Abweichung werden Fehlergrößen eingeführt.

Definition 1.1. (*Wahrer und absoluter Fehler*)

Ist A ein Näherungswert für die Zahl a , so heißt die Differenz

$$\Delta_a = a - A$$

der *wahren Fehler* von A und deren Betrag

$$|\Delta_a| = |a - A|$$

der *absoluten Fehler* von A .

Sehr oft wird in der mathematischen Literatur Δ_a bereits als absoluter Fehler und $|\Delta_a|$ als Absolutbetrag des Fehlers bezeichnet. In ingenieurwissenschaftlichen Anwendungen ist allerdings die Schreibweise in Definition 1.1 häufiger anzutreffen.

In den meisten Fällen ist die Zahl a nicht bekannt, so dass weder der wahre noch der absolute Fehler eines Näherungswertes A angegeben werden können. Daher versucht man, für den absoluten Fehler $|\Delta_a|$ von A eine möglichst kleine obere Schranke $\varepsilon_a > 0$ anzugeben, so dass $|\Delta_a| \leq \varepsilon_a$ gilt.

Definition 1.2. (*Fehlerschranke für den absoluten Fehler, absoluter Höchstfehler*)
Ist $|\Delta_a|$ der absolute Fehler eines Näherungswertes A und ist $\varepsilon_a > 0$ eine obere Schranke für $|\Delta_a|$, so dass

$$|\Delta_a| \leq \varepsilon_a$$

gilt, dann heißt ε_a eine *Fehlerschranke für den absoluten Fehler* von A oder *absoluter Höchstfehler* von A .

Bei bekanntem ε_a ist wegen $|\Delta_a| = |a - A| \leq \varepsilon_a$

$$A - \varepsilon_a \leq a \leq A + \varepsilon_a, \quad \text{also} \quad a \in [A - \varepsilon_a, A + \varepsilon_a]. \quad (1.1)$$

Um einen Näherungswert A unabhängig von der Größenordnung von a beurteilen zu können, wird der relative Fehler eingeführt.

Definition 1.3. (*Relativer Fehler*)

Ist $|\Delta_a|$ der absolute Fehler eines Näherungswertes A für die Zahl a , so heißt der Quotient

$$|\delta_a| = \frac{|\Delta_a|}{|a|} \quad \text{für } a \neq 0$$

der *relative Fehler* von A .

Streng genommen müsste man $\delta_a = \Delta_a/a$ als relativen Fehler von A bezeichnen. Das Vorzeichen von δ_a gibt dann eine zusätzliche Information über die Richtung des Fehlers, d. h. für eine positive Zahl a hat $\delta_a = (a - A)/a < 0$ demnach $a < A$, also einen zu großen Näherungswert A für a zur Folge.

Da in der Regel a unbekannt ist, wird häufig auch

$$|\delta_a| = \frac{|\Delta_a|}{|A|} \quad \text{für } A \neq 0$$

relativer Fehler von A genannt. Da dann auch Δ_a nicht exakt angebbar ist, wird man sich wieder mit der Angabe einer möglichst guten oberen Schranke für den relativen Fehler behelfen müssen.

Definition 1.4. (*Fehlerschranke für den relativen Fehler, relativer Höchstfehler*)

Ist $|\delta_a|$ der relative Fehler eines Näherungswertes A und gilt mit einem $\varrho_a > 0$

$$|\delta_a| \leq \varrho_a,$$

dann heißt ϱ_a eine *Fehlerschranke für den relativen Fehler* $|\delta_a|$ oder *relativer Höchstfehler* von $|\delta_a|$.

Ist ε_a ein absoluter Höchstfehler von A , so ist

$$\varrho_a = \varepsilon_a/|a| \quad \text{bzw.} \quad \varrho_a = \varepsilon_a/|A|$$

ein relativer Höchstfehler von A .

Definition 1.5. (*Prozentualer Fehler, Fehlerschranke für den prozentualen Fehler*)

Ist $|\delta_a|$ der relative Fehler des Näherungswertes A , so heißt

$$|\delta_a| \cdot 100$$

der *prozentuale Fehler* von A (relativer Fehler in Prozent), und σ_a mit

$$|\delta_a| \cdot 100 \leq 100 \cdot \varrho_a = \sigma_a$$

heißt eine *Fehlerschranke für den prozentualen Fehler*.

Beispiel 1.6.

Für die Zahl $x = \pi$ sind $X = 3.14$ ein Näherungswert und $\varepsilon_x = 0.0016$ eine Schranke für den absoluten Fehler $|\Delta_x|$. Also gilt nach (1.1)

$$3.1384 = 3.14 - 0.0016 \leq \pi \leq 3.14 + 0.0016 = 3.1416,$$

und der relative Höchstfehler ergibt sich zu

$$|\delta_x| \leq \varrho_x = \frac{0.0016}{3.14} \approx 0.00051.$$

Für den prozentualen Fehler folgt nach Definition 1.5

$$100 \cdot |\delta_x| = 0.051\%.$$

□

1.2 Zahlensysteme

1.2.1 Darstellung ganzer Zahlen

Für jede ganze Zahl a gibt es genau eine Potenzentwicklung zur Basis 10 (*Zehnerpotenzen*) der Gestalt

$$a = v \cdot (a_n 10^n + a_{n-1} 10^{n-1} + \dots + a_1 10^1 + a_0 10^0) = v \cdot \sum_{k=0}^n a_k 10^k$$

mit dem Vorzeichen $v \in \{-1, 1\}$, den Koeffizienten $a_k \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ und einer nicht negativen ganzen Zahl n ($n \in \mathbb{N}_0$). Diese *Dezimaldarstellung* von a erhält man, indem man die Ziffern, die zur Bezeichnung der Zahlen a_k dienen, in absteigender Reihenfolge aufschreibt:

$$a = v \cdot a_n a_{n-1} \dots a_1 a_0 \tag{1.2}$$

Die Ziffern a_k in der Dezimaldarstellung (1.2) werden auch *Stellen* von a genannt; die Stellung einer Ziffer in der Zahl gibt ihren Wert (Einer – Zehner – Hunderter usw.) wieder.

Es gibt keinen triftigen Grund, nur das uns gewohnte und vertraute *Dezimalsystem* zu benutzen. Wegen der einfachen technischen Realisierbarkeit benutzen digitale Rechenanlagen durchweg das *Dualsystem*, das auf den zwei verschiedenen Ziffern 0 und 1 beruht. Jede ganze Zahl a kann damit in eindeutiger Weise als Potenzentwicklung zur Basis 2 (Zweierpotenzen)

$$a = v \cdot \sum_{k=0}^n a_k \cdot 2^k \quad \text{mit } a_k \in \{0, 1\}$$

und einem Vorzeichen $v \in \{-1, 1\}$ geschrieben werden. Und wiederum gibt nur die Stellung einer Ziffer a_k Auskunft über ihren Stellenwert, so dass die Angabe der dualen Ziffern in der richtigen Reihenfolge zur Charakterisierung ausreicht:

$$a = v \cdot (a_n a_{n-1} \dots a_1 a_0)_2$$

Der Index 2 soll hierbei auf die Darstellung als Dualzahl verweisen; weggelassen wird nur der Index 10 für die üblichen Dezimalzahlen. So hat man beispielsweise

$$\begin{aligned} 2004 &= (+1) \cdot (2004)_{10} \\ &= (+1) \cdot (1 \cdot 2^{10} + 1 \cdot 2^9 + 1 \cdot 2^8 + 1 \cdot 2^7 + 1 \cdot 2^6 + 1 \cdot 2^4 + 1 \cdot 2^2) \\ &= (+1) \cdot (11111010100)_2. \end{aligned}$$

Definition 1.7. (*Stellenwertsystem zur Basis β*)

Sei $\beta \in \mathbb{N}$, $\beta \geq 2$. Das System der β verschiedenen Ziffern $0, 1, \dots, \beta-1$ bildet ein *Stellenwertsystem zur Basis β* , und jede ganze Zahl a lässt sich darin in der Form

$$a = v \cdot \sum_{k=0}^n a_k \cdot \beta^k = v \cdot (a_n a_{n-1} \dots a_1 a_0)_\beta$$

mit eindeutig bestimmten Ziffern $a_k \in \{0, 1, \dots, \beta-1\}$ und einem Vorzeichen $v \in \{-1, 1\}$ darstellen.

Die Wahl $\beta = 10$ führt auf das geläufige Dezimalsystem und $\beta = 2$ auf das Dualsystem. Anwendungen finden immer wieder auch die Basiswahlen $\beta = 8$ (*Oktalsystem*) und $\beta = 16$ (*Hexadezimalsystem* mit den Ziffern $0, 1, \dots, 9, A, B, C, D, E, F$)

Die gegenseitige Konvertierung von Zahlen in unterschiedlichen Stellenwertsystemen lässt sich einfach bewerkstelligen, wobei es reicht, als ein Bezugssystem das Dezimalsystem anzunehmen. Die Umwandlung einer β -Zahl in die zugehörige Dezimalzahl erfolgt ökonomisch mit dem *Horner-Schema* (s. Abschnitt 3.2) für Polynome:

$$\begin{aligned} a &= v \cdot (a_n \beta^n + a_{n-1} \beta^{n-1} + a_{n-2} \beta^{n-2} + \dots + a_1 \beta + a_0) \\ &= v \cdot \left(\underbrace{\left\{ \dots \left[\underbrace{(a_n \beta + a_{n-1})}_{s_{n-1}} \beta + a_{n-2} \right] \beta + \dots + a_1 \right\}}_{s_{n-2}} \beta + a_0 \right) \\ &\quad \underbrace{\hspace{10em}}_{s_1} \\ &\quad \underbrace{\hspace{15em}}_{s_0} \end{aligned}$$

Algorithmus 1.8. (*Umwandlung einer β -Zahl in eine Dezimalzahl*)

Berechnet man für eine im Stellenwertsystem zur Basis β gegebene ganze Zahl $a = v \cdot (a_n a_{n-1} \dots a_1 a_0)_\beta$, $v \in \{+1, -1\}$, ausgehend von $s_n = a_n$, nacheinander die Größen

$$s_k = \beta \cdot s_{k+1} + a_k, \quad k = n-1, n-2, \dots, 1, 0,$$

dann ist $a = v \cdot s_0$ im Dezimalsystem.

Umgekehrt folgt aus dem letzten Zwischenergebnis

$$a = v \cdot s_0 = v \cdot (s_1 \cdot \beta + a_0),$$

dass a_0 als der Rest der Division der ganzen Zahl s_0 durch β angesehen werden kann:

$$\frac{s_0}{\beta} = s_1 + \frac{a_0}{\beta} \quad \text{oder} \quad \frac{s_0}{\beta} = s_1 \quad \text{Rest } a_0$$

Geht man schrittweise weiter zurück, so folgen die nächsten β -Ziffern a_1, a_2 und so weiter.

Algorithmus 1.9. (*Umwandlung einer Dezimalzahl in eine β -Zahl*)

Berechnet man für eine im Dezimalsystem gegebene ganze Zahl a , ausgehend von $s_0 = |a|$, aus der Gleichung

$$\frac{s_k}{\beta} = s_{k+1} \quad \text{Rest } a_k$$

nacheinander für $k = 0, 1, 2, \dots$ die Größen s_1 und a_0 , s_2 und a_1 usw., bis für ein $k = n$ der Wert $s_{n+1} = 0$ ist, dann ist mit den auf diese Weise gewonnenen Ziffern $a_0, a_1, \dots, a_n \in \{0, 1, \dots, \beta-1\}$ die β -Darstellung von a gegeben durch

$$a = v \cdot (a_n a_{n-1} \dots a_1 a_0)_\beta,$$

wobei $v \in \{-1, 1\}$ das Vorzeichen von a bezeichnet.

Die Zahl 2004 besitzt demnach wegen

$$\begin{array}{rclcl} \frac{2004}{2} & = & 1002 & \text{Rest } 0 & (= a_0) \\ \frac{1002}{2} & = & 501 & \text{Rest } 0 & (= a_1) \\ \frac{501}{2} & = & 250 & \text{Rest } 1 & (= a_2) \\ \frac{250}{2} & = & 125 & \text{Rest } 0 & (= a_3) \\ \frac{125}{2} & = & 62 & \text{Rest } 1 & (= a_4) \\ \frac{62}{2} & = & 31 & \text{Rest } 0 & (= a_5) \\ \frac{31}{2} & = & 15 & \text{Rest } 1 & (= a_6) \\ \frac{15}{2} & = & 7 & \text{Rest } 1 & (= a_7) \\ \frac{7}{2} & = & 3 & \text{Rest } 1 & (= a_8) \\ \frac{3}{2} & = & 1 & \text{Rest } 1 & (= a_9) \\ \frac{1}{2} & = & 0 & \text{Rest } 1 & (= a_{10}) \end{array} \quad \uparrow$$

die Dualdarstellung

$$2004 = (+1) \cdot (11111010100)_2.$$

Rechnerintern wird eine ganze Zahl $a = v \cdot |a|$ als Dualzahl durch eine Reihe von Bits (“binary digits”) abgespeichert, gewöhnlich in zwei oder vier Bytes (1 Byte = 8 Bits). Bei einer 2-Byte-Hinterlegung bedeutet das:

1. Byte								2. Byte							
μ	α_{14}	α_{13}	α_{12}	α_{11}	α_{10}	α_9	α_8	α_7	α_6	α_5	α_4	α_3	α_2	α_1	α_0

Das Vorzeichen wird dann über

$$\mu = \begin{cases} 0, & \text{falls } v = +1 \\ 1, & \text{falls } v = -1 \end{cases}$$

gewählt, und im Fall einer positiven Zahl entsprechen α_{14} bis α_0 den Dualziffern von a :

$$a = (+1) \cdot (\alpha_{14}\alpha_{13} \dots \alpha_1\alpha_0)_2$$

Als größte darstellbare positive Zahl ergibt sich dann

$$a = (+1) \cdot (11111111111111)_2 = \sum_{k=0}^{14} 1 \cdot 2^k = \frac{2^{15} - 1}{2 - 1} = 32\,767.$$

Damit eine Subtraktion auf eine Addition über $a - b = a + (-b)$ zurückgeführt werden kann, werden negative Zahlen rechnerintern durch ein Komplement dargestellt, d. h. im Fall $\mu = 1$ bzw. $v = -1$ stimmen die Hinterlegungen α_k nicht mehr mit den Dualziffern a_k der Zahl $a = (-1) \cdot (a_{14}a_{13} \dots a_1a_0)_2$ überein.

Vorsicht ist geboten, wenn bei einer Addition oder Subtraktion das Ergebnis nicht mehr im darstellbaren Bereich liegt: Addiert man beispielsweise bei einer 1-Byte-Zahl [größte darstellbare positive Zahl: $2^7 - 1 = 127$] $97 = (+1) \cdot (1100001)_2$ und $43 = (+1) \cdot (0101011)_2$, so folgt

$$\begin{array}{r} 0 : 1\,1\,0\,0\,0\,0\,1 \\ + \quad 0 : 0\,1\,0\,1\,0\,1\,1 \\ \hline 1 : 0\,0\,0\,1\,1\,0\,0 \end{array}$$

d. h. ein Übertrag beeinflusst letztendlich das Vorzeichenbit und führt auf ein völlig inplausibles negatives Resultat! Solche Effekte treten leider häufiger auf, ohne dass vom Rechnersystem eine Fehlermeldung oder zumindest eine Warnung ausgesprochen wird. Deshalb ist **bei Rechnerergebnissen stets eine Prüfung auf Plausibilität** erforderlich!

1.2.2 Darstellung reeller Zahlen

Jede nicht ganze, reelle Zahl a besitzt eine Entwicklung der Form

$$a = v \cdot \left(\sum_{k=0}^n a_k 10^k + \sum_{k=1}^{\infty} b_k 10^{-k} \right)$$

mit einem Vorzeichen $v \in \{-1, 1\}$ und Ziffern $a_k, b_k \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$, wobei mindestens ein Term

$$b_k 10^{-k} \neq 0$$

auftritt. Die Dezimaldarstellung einer nicht ganzen Zahl heißt *Dezimalbruch*. Im Dezimalbruch einer Zahl a wird zwischen a_0 und b_1 ein Punkt, der sog. *Dezimalpunkt*, gesetzt:

$$a = v \cdot a_n a_{n-1} a_{n-2} \dots a_1 a_0 \cdot b_1 b_2 \dots b_t \dots \quad (1.3)$$

Die rechts vom Punkt notierten Stellen heißen *Dezimalstellen* oder *Dezimalen*. Gibt es in einem Dezimalbruch eine Dezimale $b_j \neq 0$, so dass alle folgenden Dezimalen $b_{j+1} = b_{j+2} = \dots = 0$ sind, dann heißt der Dezimalbruch *endlich*, andernfalls *unendlich*. Eine Darstellung (1.3) mit endlich vielen Dezimalstellen heißt *Festpunktdarstellung*. Demnach vergegenwärtigt $2\frac{1}{8} = 2.125$ einen endlichen Dezimalbruch und $\frac{1}{3} = 0.3333\dots$ nicht. Dezimalbruchdarstellungen sind, abgesehen vom Sonderfall der Neunerperiode (z. B. ist $-1.29999\dots = -1.2\bar{9} = -1.3$), eindeutig.

In einem Stellenwertsystem zur Basis β gilt analog die allgemeine Darstellung

$$\begin{aligned} a &= v \cdot \left(\sum_{k=0}^n a_k \beta^k + \sum_{k=1}^{\infty} b_k \beta^{-k} \right) \\ &= v \cdot (a_n a_{n-1} \dots a_1 a_0 \cdot b_1 b_2 \dots)_\beta \end{aligned}$$

mit $v \in \{+1, -1\}$ und $a_k, b_k \in \{0, 1, \dots, \beta-1\}$; der Index β verweist wieder auf das zugrunde liegende Stellenwertsystem und entfällt nur im vertrauten Fall $\beta = 10$. Der zwischen den Ziffern a_0 und b_1 gesetzte Punkt heißt nun *β -Punkt*, und man nennt $(a_n a_{n-1} \dots a_1 a_0)_\beta$ – manchmal auch unter Berücksichtigung des Vorzeichens v – den *ganzzahligen Anteil* und $(.b_1 b_2 b_3 \dots)_\beta$ den *gebrochenen* oder *fraktionierten Anteil* von a . Der Sonderfall $\beta = 2$ führt nun zur *Dualbruchdarstellung* einer reellen Zahl. Gibt es nur endliche viele „Nachkomma“-Stellen, so spricht man von einem *endlichen β -Bruch*, andernfalls von einem *unendlichen β -Bruch*.

Es gilt der

Hilfssatz 1.10.

Jede rationale Zahl p/q mit teilerfremden $p \in \mathbb{Z}$ und $q \in \mathbb{N}$ wird durch einen endlichen oder durch einen unendlichen periodischen Dezimal- oder Dualbruch dargestellt, jede irrationale Zahl durch einen unendlichen nicht periodischen Dezimal- oder Dualbruch.

Zur Umwandlung reeller Zahlen in ein β -System reicht es, den ganzzahligen Anteil – siehe Abschnitt 1.2.1 – und den gebrochenen Anteil unabhängig voneinander zu konvertieren. Beispielsweise folgt mit $\beta = 2$ aus

$$\begin{aligned} 0.625 &= (+1) \cdot (.b_1 b_2 b_3 \dots)_2 && \text{mit } b_k \in \{0, 1\} \\ &= b_1 2^{-1} + b_2 2^{-2} + b_3 2^{-3} + \dots \end{aligned}$$

durch Multiplikation mit $\beta = 2$

$$\begin{aligned} 1.25 &= b_1 + b_2 2^{-1} + b_3 2^{-2} + \dots \\ &= (b_1 . b_2 b_3 \dots)_2 . \end{aligned}$$

Somit entspricht b_1 dem ganzzahligen Anteil 1 von 1.25 und weiter

$$0.25 = (.b_2b_3\dots)_2 = b_2 2^{-1} + b_3 2^{-2} + \dots$$

Führt man nun mit einer erneuten Multiplikation mit $\beta = 2$ fort

$$0.5 = (b_2.b_3\dots)_2 = b_2 + b_3 2^{-1} + \dots,$$

so führt der Vergleich der ganzzahligen und der gebrochenen Anteile in dieser Gleichung auf $b_2 = 0$ und $0.5 = (.b_3b_4\dots)_2$. Der nächste gleichartige Schritt ergibt über

$$1.0 = (b_3.b_4\dots)_2 = b_3 + b_4 2^{-1} + \dots$$

schließlich $b_3 = 1$ und $b_k = 0$ für $k \geq 4$. Mithin gilt

$$0.625 = (+1) \cdot (.101)_2,$$

was auch plausibel ist, denn 0.625 ist die Summe von $\frac{1}{2} = 2^{-1}$ und $\frac{1}{8} = 2^{-3}$.

Allgemein formuliert heißt das:

Algorithmus 1.11. (*Umwandlung einer echt gebrochenen Zahl in einen β -Bruch*)

Gegeben sei eine echt gebrochene Zahl a , $-1 < a < 1$.

Berechnet man, ausgehend von $c_0 = |a|$, für $k = 1, 2, 3, \dots$ nacheinander die Größen

$$\begin{aligned} b_k &= \text{int}(\beta \cdot c_{k-1}) && \text{„ganzzahliger Anteil von } \beta \cdot c_{k-1}\text{“,} \\ c_k &= \beta \cdot c_{k-1} - b_k && \text{„echt gebrochener Anteil von } \beta \cdot c_{k-1}\text{“,} \end{aligned}$$

dann ist

$$a = v \cdot (.b_1b_2b_3\dots)_\beta,$$

wobei $v \in \{-1, 1\}$ das Vorzeichen von a wiedergibt.

Die Berechnung wird abgebrochen, wenn hinreichend viele „Nachkomma“-Stellen ermittelt wurden.

Beispiel 1.12.

Algorithmus 1.11 führt mit $a = 0.1$ auf die folgenden Größen für das Dualsystem:

$$\begin{aligned} b_1 &= \text{int}(0.2) = 0, & c_1 &= 0.2 \\ b_2 &= \text{int}(0.4) = 0, & c_2 &= 0.4 \\ b_3 &= \text{int}(0.8) = 0, & c_3 &= 0.8 \\ b_4 &= \text{int}(1.6) = 1, & c_4 &= 0.6 \\ b_5 &= \text{int}(1.2) = 1, & c_5 &= 0.2 \\ b_6 &= \text{int}(0.4) = 0, & c_6 &= 0.4 \\ &\dots & &\dots \end{aligned}$$

Man erkennt, dass sich die Dualziffernfolge periodisch wiederholt:

$$\begin{aligned} 0.1 &= (+1) \cdot (.00011001100110011\dots)_2 \\ &= (+1) \cdot (.000\overline{11})_2. \end{aligned}$$

Mithin entspricht 0.1 einem unendlichen periodischen Dualbruch. □

Definition 1.13. (*Tragende Ziffern*)

Alle Ziffern in der β -Darstellung einer Zahl $a = v \cdot (a_n a_{n-1} \dots a_0 . b_1 b_2 \dots)_\beta$ mit $v \in \{+1, -1\}$ und $a_k, b_k \in \{0, 1, \dots, \beta-1\}$, beginnend mit der ersten von Null verschiedenen Ziffer ($a_n \neq 0$), heißen *tragende Ziffern*.

Beispiel 1.14.

$a = 0.0024060$	besitzt 7 Dezimalen	und	5 tragende Ziffern,
$a = 1573800$	besitzt keine Dezimalen	und	7 tragende Ziffern.
$a = 47.110$	besitzt 3 Dezimalen	und	5 tragende Ziffern.
$a = 0.1$	besitzt 1 Dezimale	und	1 tragende Ziffer.

Für die letzte Zahl ergibt sich im Dualsystem $a = (0.00011)_2$, eine Zahl mit unendlich vielen Dualstellen und somit unendlich vielen tragenden Ziffern. $\square$

Definition 1.15. (*Normalisierte Gleitpunktdarstellung*)

Jede reelle Zahl $a \neq 0$ kann als β -Zahl in der Form

$$a = v \cdot (.d_1 d_2 d_3 \dots d_s d_{s+1} \dots)_\beta \cdot \beta^k \quad (1.4)$$

für ein $k \in \mathbb{Z}$ dargestellt werden, wobei $v \in \{+1, -1\}$ das Vorzeichen von a angibt und $d_1 \neq 0$ gilt. (1.4) heißt *normalisierte β -Gleitpunktdarstellung* von a , $m = (.d_1 d_2 d_3 \dots)_\beta$ ihre β -Mantisse, und k ihr β -Exponent. Besitzt die Mantisse s tragende Ziffern, $s \in \mathbb{N}$, so heißt sie *s-stellig*.

So besitzen $a = 346.5201$ und $b = -0.005386$ im Dezimalsystem die normalisierten Gleitpunktdarstellungen

$$\begin{aligned} a &= 0.3465201 \cdot 10^3 && \text{(7-stellige Mantisse),} \\ b &= -0.5386 \cdot 10^{-2} && \text{(4-stellige Mantisse).} \end{aligned}$$

Einem Computer stehen für Berechnungen nur endlich viele in ihm darstellbare Zahlen, die *Maschinenzahlen*, zur Verfügung. Die Mantissen m dieser Maschinenzahlen haben gewöhnlich eine feste Anzahl von Ziffern. Ferner ist der Exponent $k \in \mathbb{Z}$ durch $-k_1 \leq k \leq k_2$ mit $k_1, k_2 \in \mathbb{N}$ begrenzt. Eine weltweit gültige Norm nach ANSI (American National Standards Institute) und IEEE (Institute of Electrical and Electronics Engineers) schreibt für reelle Zahlen, als Dualbruch mit 4 Bytes (= 32 Bits) im Rechner hinterlegt, das Format

$$\mu : e_1 \dots e_8 : d_2 \ d_3 \dots \dots \dots \dots \dots \dots d_{24}$$

vor, wobei

- μ das Vorzeichen der Zahl wiedergibt,
- $e_1 \dots e_8$ zur Darstellung des Exponenten k in (1.4) gehört; als nicht negative ganze Zahl, auch *Charakteristik* genannt, wird hiervon stets die Konstante 127 abgezogen, d. h. $k = (e_1 \dots e_8)_2 - 127$,
- $d_2 \dots d_{24}$ die Mantisse (bei negativen Zahlen wieder komplementär genommen) der Zahl darstellt, wobei $d_1 = 1$ zu ergänzen ist (als normalisierte Gleitpunktdarstellung bleibt für $d_1 \neq 0$ im Dualsystem nur $d_1 = 1$ übrig!).

Hinzu kommen zusätzliche Kennungen für Fehlerfälle wie "Overflow" (etwa bei Division durch 0) und anderes mehr.

Für die Darstellung von $0.1 = (+1) \cdot (.00011)_2 = (+1) \cdot (.1100)_2 \cdot 2^{-3}$ bedeutet das

$$\boxed{0 : \underbrace{01111100}_{=124} : 100 \overset{d_2}{1100} 1100 1100 1100 1100} \overset{d_{23}}{1100} 11 \dots$$

Als 2-Exponent ergibt sich $124 - 127 = -3$, und $d_1 = 1$ ist unterdrückt worden. Mithin ist klar, dass 0.1 im Rechner nicht exakt wiedergegeben werden kann! Rundet die Rechnerarithmetik auf, so wird d_{24} zu 1 gewählt, und die interne Maschinenzahl ist größer als 0.1 (genauer: 0.1000000015...); wird abgerundet, also $d_{24} = 0$ gesetzt, dann hat die interne Maschinenzahl einen Wert unterhalb von 0.1 (genauer: 0.0999999940...). Deshalb wird ein Rechner – übrigens unabhängig vom Speicherformat – für das Produkt $10 \cdot 0.1$ nie genau den Wert 1 ermitteln.

Dass mit Gleitpunktzahlen auf Rechenanlagen überhaupt gearbeitet wird, liegt daran, dass man damit bei gleicher Speichergröße einen enorm viel größeren Zahlenbereich überstreicht.

Beispiel 1.16.

Nimmt man – abgesehen vom Vorzeichen – an, dass 8 dezimale Speichereinheiten zur Verfügung stehen, so ließen sich damit in den einzelnen Systemen die folgenden positiven Zahlen darstellen:

1) Ganzzahlsystem

- kleinste darstellbare positive Zahl 0000 0001
- größte darstellbare positive Zahl 9999 9999

2) Festpunktsystem mit 4 Dezimalen

- kleinste darstellbare positive Zahl 0000.0001
- größte darstellbare positive Zahl 9999.9999

3) Normalisiertes Gleitpunktsystem mit 6 Mantissen- und 2 Exponentenziffern (Exponent = Charakteristik – 50)

- kleinste darstellbare positive Zahl $.100\,000 \cdot 10^{-50}$
- größte darstellbare positive Zahl $.999\,999 \cdot 10^{49}$