de Gruyter Lehrbuch

Peter Deuflhard
Andreas Hohmann

Numerische Mathematik I

Eine algorithmisch orientierte Einführung

3. Auflage

W
DE
G

Peter Deuflhard
Andreas Hohmann

Numerische Mathematik I

Eine algorithmisch orientierte Einführung

3., überarbeitete und erweiterte Auflage

Walter de Gruyter
Berlin · New York 2002

Professor Dr. Peter Deuflhard
Konrad-Zuse-Zentrum für
Informationstechnik Berlin (ZIB)
Takustr. 7
14195 Berlin
und
Freie Universität Berlin
Fachbereich Mathematik und
Informatik

Dr. Andreas Hohmann
Max-Planck-Str. 3
40670 Meerbusch

Mathematics Subject Classification 2000: Primary: 65-01
Secondary: 65 Bxx, 65 Cxx, 65 Dxx, 65 Fxx, 65 Gxx

⊗ Gedruckt auf säurefreiem Papier, das die US-ANSI-Norm über Haltbarkeit erfüllt.

Die Deutsche Bibliothek – CIP-Einheitsaufnahme

Numerische Mathematik / Peter Deuflhard ; Folkmar Bornemann. – Berlin ; New York : de Gruyter
(De-Gruyter-Lehrbuch)
Engl. Ausg. u.d.T.: Numerical analysis
1. Eine algorithmisch orientierte Einführung. – 3., überarb. und erw. Aufl. – 2002
ISBN 3-11-017182-1

Printed in Germany.
Umschlaggestaltung: Hansbernd Lindemann, Berlin.
Konvertierung von LaTeX-Dateien der Autoren: I. Zimmermann, Freiburg.
Druck und Bindung: Hubert & Co. GmbH & Co. KG, Göttingen.

de Gruyter Lehrbuch

Deuflhard / Hohmann · Numerische Mathematak I

Vorwort

Numerische Mathematik in ihrer algorithmisch orientierten Ausprägung beinhaltet die Konstruktion und das mathematische Verständnis von numerischen Algorithmen, also von Rechenmethoden zur zahlenmäßigen Lösung mathematischer Probleme. Die meisten mathematischen Probleme kommen in unseren Tagen aus vielfältigen Anwendungsgebieten außerhalb der Mathematik. In der Tat haben sich *mathematische Modelle* zur näherungsweisen Beschreibung der Wirklichkeit in den letzten Jahren derart verfeinert, dass ihre Computersimulation die Realität zunehmend genauer widerspiegelt. Treibende Kraft dieser Entwicklung ist der gleichermaßen stürmische Fortschritt bei Computern und Algorithmen. Dabei hat sich gezeigt: nicht nur die Verfügbarkeit immer besserer Computer, sondern mehr noch die Entwicklung immer besserer Algorithmen macht heute immer komplexere Probleme lösbar. Bisher verschlossene Bereiche der Natur- und Ingenieurwissenschaften öffnen sich mehr und mehr einer mathematischen Modellierung und damit der Simulation auf dem Rechner.

Angesichts dieser Entwicklung versteht sich Numerische Mathematik heute als Teil des übergeordneten Gebietes *Scientific Computing*, zu deutsch oft auch als *Wissenschaftliches Rechnen* übersetzt. Dieses Gebiet im interdisziplinären Spannungsfeld von Mathematik, Informatik, Natur- und Ingenieurwissenschaften ist erst in jüngerer Zeit zusammengewachsen. Es wirkt in zahlreiche Zweige der Industrie (Chemie, Elektronik, Robotik, Fahrzeugbau, Luft- und Raumfahrt etc.) hinein und leistet bei wichtigen gesellschaftlichen Fragen (sparsamer und zugleich umweltverträglicher Umgang mit Primärenergie, globale Klimamodelle, Verbreitung von Epidemien etc.) einen unverzichtbaren Beitrag. Als Konsequenz davon haben sich tiefgreifende Änderungen der Stoffauswahl und der Darstellungsweise in Vorlesungen und Seminaren der Numerischen Mathematik zwingend ergeben, und dies bereits in einführenden Veranstaltungen: manches früher für wichtig Gehaltene fällt ersatzlos weg, anderes kommt neu hinzu. Die hier getroffene Auswahl ist natürlich vom fachlichen Geschmack der Autoren geprägt, hat sich allerdings nun bereits in der dritten Auflage dieses erfreulich verbreiteten Lehrbuches bewährt.

Das vorliegende Buch richtet sich in erster Linie an Studierende der Mathematik, Informatik, Natur- und Ingenieurwissenschaften. In zweiter Linie wollen wir aber auch bereits im Beruf stehende Kollegen (und Kolleginnen – hier ein für alle Mal) oder Quereinsteiger erreichen, die sich mit den etablierten modernen Konzepten der Numerischen Mathematik auf elementarer Ebene im Selbststudium vertraut machen wollen. Der Stoff setzt lediglich Grundkenntnisse der Mathematik voraus, wie sie an deutschsprachigen Universitäten in den Grundvorlesungen „Lineare Algebra I/II“ und „Analysis I/II“ üblicherweise vermittelt werden. Weitergehende Kenntnisse werden in diesem einführenden Lehrbuch nicht verlangt. In einer Reihe von Einzelthemen (wie Interpolation oder Integration) haben wir uns bewusst auf den *eindimensionalen* Fall beschränkt. Durchgängiges Muster dieser Einführung ist, wesentliche Konzepte der modernen Numerik, die später auch bei gewöhnlichen und partiellen Differentialgleichungen eine Rolle spielen, bereits hier am einfachst möglichen Problemtyp zu behandeln.

Oberstes Ziel des Buches ist die Förderung des *algorithmischen Denkens*, das ja historisch eine der Wurzeln unserer heutigen Mathematik ist. Es ist kein Zufall, dass neben heutigen Namen auch historische Namen wie Gauß, Newton und Tschebyscheff an zahlreichen Stellen des Textes auftauchen. Die Orientierung auf Algorithmen sollte jedoch nicht missverstanden werden: gerade effektive Algorithmen erfordern ein gerüttelt Maß an mathematischer Theorie, die innerhalb des Textes auch aufgebaut wird. Die Argumentation ist in der Regel mathematisch elementar; wo immer sinnvoll, wird *geometrische Anschauung* herangezogen – was auch die hohe Anzahl an Abbildungen erklärt. Begriffe wie Skalarprodukt und Orthogonalität finden durchgängig Verwendung, bei endlicher Dimension ebenso wie in Funktionenräumen. Trotz der elementaren Darstellung enthält das Buch zahlreiche Resultate, die ansonsten unpubliziert sind. Darüber hinaus unterscheidet sich auch bei eher klassischen Themen unsere Herleitung von der in herkömmlichen Lehrbüchern.

Der Erstautor hat seit 1978 Vorlesungen zur Numerischen Mathematik gehalten – u.a. an der Technischen Universität München, der Universiät Heidelberg und der Freien Universität Berlin. Er hat die Entwicklung des Gebietes Scientific Computing durch seine wissenschaftliche Tätigkeit über Jahre weltweit mit beeinflusst. Der Zweitautor hatte zunächst seine Ausbildung mit Schwerpunkt Reine Mathematik an der Universität Bonn und ist erst anschließend in das Gebiet der Numerischen Mathematik übergewechselt. Diese Kombination hat dem vorliegenden Buch sicher gutgetan, wie die unveränderte Aktualität der Themen auch in dieser dritten Auflage zeigt.

Gegenüber der zweiten Auflage ist im wesentlichen nur das Kapitel 5.5 über stochastische Eigenwertprobleme hinzugekommen, da diese Problemklasse in den letzten Jahren zunehmend an Bedeutung gewonnen hat und sich für eine elementare Darstellung im Rahmen unseres Gesamtkonzeptes eignet. Darüber hinaus liegt der auf diesem ersten Band aufbauende zweite Band über die Numerik von gewöhnlichen Differentialgleichungen inzwischen bereits in einer überarbeiteten und erweiterten zweiten Auflage [23] vor.

An dieser Stelle nehmen wir gerne die Gelegenheit wahr, eine Reihe von Kollegen zu bedenken, die uns bei diesem Buch auf die eine oder andere Weise besonders unterstützt haben. Der Erstautor blickt dankbar zurück auf seine Zeit als Assistent von R. Bulirsch (TU München, emeritiert seit 2001), in dessen Tradition sich sein heutiger Begriff von Scientific Computing geformt hat. Intensive Diskussionen und vielfältige Anregungen zahlreicher Kollegen sind in unsere Darstellung mit eingeflossen. Einigen Kollegen wollen wir hier zu folgenden Einzelthemen besonders danken: Ernst Hairer und Gerhard Wanner (Universität Genf) zur Diskussion des Gesamtkonzepts des Buches; Wolfgang Dahmen (RWTH Aachen) zu Kapitel 7; Folkmar Bornemann (TU München) zur Darstellung der Fehlertheorie, der verschiedenen Konditionsbegriffe sowie zur Definition des Stabilitätsindikators in Kapitel 2; Dietrich Braess (Ruhruniversität Bochum) zur rekursiven Darstellung der schnellen Fourier-Transformation in Kapitel 7.2. Für die vorliegende dritte Auflage geht unser besonders herzlicher Dank an Rainer Roitzsch (ZIB), ohne dessen in die Tiefe gehende TEX-Kenntnisse dieses Buch nicht hätte erscheinen können, sowie an Erlinda Körnig und Sigrid Wacker für vielfältige Hilfe.

Berlin und Düsseldorf, Januar 2002

Peter Deuflhard
Andreas Hohmann

Inhaltsverzeichnis

Überblick

Dieses einführende Lehrbuch richtet sich in erster Linie an Studierende der Mathematik, der Informatik, der Natur- und Ingenieurwissenschaften. In zweiter Linie wendet es sich jedoch ausdrücklich auch an bereits im Beruf stehende Kollegen oder Quereinsteiger aus anderen Disziplinen, die sich mit den moderneren Konzepten der Numerischen Mathematik auf elementarer Ebene im Selbststudium vertraut machen wollen.

Das Buch gliedert sich in neun Kapitel mit dazugehörigen Übungsaufgaben, ein Softwareverzeichnis, ein Literaturverzeichnis und einen Index. Die ersten fünf und die letzten vier Kapitel sind inhaltlich eng verknüpft.

In **Kapitel 1** beginnen wir mit der *Gauß-Elimination* für lineare Gleichungssysteme als dem klassischen Musterfall eines Algorithmus. Über die elementare Elimination hinaus diskutieren wir Pivotstrategien und Nachiteration als Zusatzelemente. **Kapitel 2** enthält die unverzichtbare *Fehleranalyse*, fußend auf den Grundgedanken von Wilkinson. Kondition eines Problems und Stabilität eines Algorithmus werden einheitlich dargestellt, sauber auseinandergehalten und zunächst an einfachen Beispielen illustriert. Die leider noch allzu oft übliche ε-Schlacht im Rahmen der linearisierten Fehlertheorie können wir vermeiden – was zu einer drastischen Vereinfachung in der Darstellung und im Verständnis führt. Als Besonderheit ergibt sich ein Stabilitätsindikator, der eine kompakte Klassifikation der numerischen Stabilität gestattet. Mit diesem Rüstzeug können wir schließlich die Frage, wann eine gegebene Näherungslösung eines linearen Gleichungssystems akzeptabel ist, algorithmisch beantworten. In **Kapitel 3** behandeln wir Orthogonalisierungsverfahren im Zusammenhang mit der Gauß'schen *linearen Ausgleichsrechnung* und führen den äußerst nützlichen Kalkül der Pseudoinversen ein. Er findet im anschließenden **Kapitel 4** unmittelbar Anwendung. Dort stellen wir Interationsverfahren für nichtlineare Gleichungssysteme (*Newton-Verfahren*), nichtlineare Ausgleichsprobleme (*Gauß-Newton-Verfahren*) und parameterabhängige Probleme (*Fortsetzungsmethoden*) in engem inneren Zusammenhang dar. Besondere Aufmerksamkeit widmen wir der modernen affin-invarianten Form von Konvergenztheorie und iterativen Algorithmen. **Kapitel 5** beginnt mit der Konditionsanalyse des *linearen Eigenwertproblems* für allgemeine Matrizen. Dies richtet das Augenmerk zunächst in natürlicher weise auf den reell-symmetrischen Fall: für diesen Fall stellen wir Vektoriteration und *QR*-Algorithmus im Detail vor. In den gleichen Zusammenhang passt auch die in den Anwendungen so wichtige Singulärwertzerlegung für allgemeine Matrizen. Abschließend betrachten wir stochastische Eigenwertprobleme, die in der dritten Auflage neu hinzugekommen sind. Sie spielen insbesondere in der sogenannten Clusteranalyse in letzter Zeit eine zunehmend wichtige Rolle.

Die zweite inhaltlich gechlossene Sequenz beginnt in **Kapitel 6** mit einer ausführlichen Behandlung der Theorie von *Drei-Term-Rekursionen*, die bei der Realisierung von Orthogonalprojektionen in Funktionenräumen eine Schlüsselrolle spielen. Die Kondition von Drei-Term-Rekursionen stellen wir anhand der diskreten Greenschen Funktionen dar und bereiten so eine mathematische Struktur vor, die in Anfangs- und Randwertproblemen bei Differentialgleichungen wiederkehrt. Mit dem verstärkten Aufkommen des symbolischen Rechnens hat sich gerade in letzter Zeit das Interesse an *Speziellen Funktionen* auch in der Numerik wiederbelebt. Numerische Algorithmen für ihre effiziente Summation über die zugehörigen Drei-Term-Rekursionen illustrieren wir am Beispiel der Kugel- und Besselfunktionen. In **Kapitel 7** behandeln wir zunächst die klassische polynomielle *Interpolation* und *Approximation* im eindimensionalen Fall. Wir führen sie sodann weiter über Bézier-Technik und Splines bis hin zu Methoden, die heute im CAD (Computer Aided Design) oder CAGD (Computer Aided Geometric Design), also in Disziplinen der Computergraphik, von zentraler Bedeutung sind. Unsere Darstellung in **Kapitel 8** über *iterative* Methoden zur Lösung von *großen* symmetrischen Gleichungssystemen stützt sich in bequemer Weise auf Kapitel 6 (Drei-Term-Rekursion) und Kapitel 7 (Minimaxeigenschaft von Tschebyscheff-Polynomen). Das gleiche gilt für den Lanczos-Algorithmus für große symmetrische Eigenwertprobleme.

Das abschließende **Kapitel 9** ist mit voller Absicht etwas länger geraten: Es trägt den Hauptteil der Last, Prinzipien der numerischen Lösung von gewöhnlichen und partiellen Differentialgleichungen vorab ohne technischen Ballast am einfachst möglichen Problemtyp vorzustellen, hier also an der numerischen Quadratur. Nach den historischen Newton-Cotes-Formeln und der Gauß-Quadratur stellen wir die klassische Romberg-Quadratur als einen ersten adaptiven Algorithmus dar, bei dem jedoch nur die Approximationsordnung variabel ist. Die Formulierung des Quadraturproblems als *Anfangswertproblem* bietet uns sodann die Möglichkeit, eine *adaptive Romberg-Quadratur* mit Ordnungs- *und* Schrittweitensteuerung auszuarbeiten; dies liefert uns zugleich den didaktischen Einstieg in adaptive *Extrapolationsverfahren*, die bei der Lösung gewöhnlicher Differentialgleichungen eine tragende Rolle spielen. Die alternative Formulierung des Quadraturproblems als *Randwertproblem* nutzen wir zur Herleitung einer *adaptiven Mehrgitter-Quadratur*; auf diese Weise stellen wir das adaptive Prinzip bei Mehrgittermethoden für partielle Differentialgleichungen separiert vom Prinzip der schnellen Lösung am einfachsten Modellfall dar.

1 Lineare Gleichungssysteme

In diesem Kapitel behandeln wir die numerische Lösung eines Systems von n linearen Gleichungen

$$\begin{array}{ccccccccc} a_{11}x_1 & + & a_{12}x_2 & + & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & a_{22}x_2 & + & \cdots & + & a_{2n}x_n & = & b_2 \\ \vdots & & & & & & \vdots & & \vdots \\ a_{n1}x_1 & + & a_{n2}x_2 & + & \cdots & + & a_{nn}x_n & = & b_n \end{array}$$

oder kurz

$$Ax = b,$$

wobei $A \in \mathrm{Mat}_n(\mathbb{R})$ eine reelle (n, n)-Matrix ist und $b, x \in \mathbb{R}^n$ reelle n-Vektoren sind. Bevor wir mit der Berechnung der Lösung x beginnen, stellen wir uns die Frage:

Wann ist ein lineares Gleichungssystem überhaupt lösbar?

Aus der linearen Algebra kennen wir das folgende Resultat, das die Lösbarkeit mit Hilfe der Determinante der Matrix A charakterisiert.

Satz 1.1. *Sei* $A \in \mathrm{Mat}_n(\mathbb{R})$ *eine reelle quadratische Matrix mit* $\det A \neq 0$ *und* $b \in \mathbb{R}^n$. *Dann existiert genau ein* $x \in \mathbb{R}^n$, *so dass* $Ax = b$.

Falls $\det A \neq 0$, so lässt sich die Lösung $x = A^{-1}b$ mit der Cramerschen Regel berechnen, zumindest im Prinzip. Offenbar gibt es hier eine direkte Verbindung von Existenz- und Eindeutigkeitsaussagen zum Rechenverfahren. Allgemein werden wir fordern: falls das Problem keine Lösung hat, darf ein verlässlicher Algorithmus auch keine „ausrechnen". Das ist nicht selbstverständlich, es gibt Gegenbeispiele. *Verlässlichkeit* ist also ein erste wichtige Eigenschaft eines „guten" Algorithmus.

Allerdings kann die Cramersche Regel noch nicht das endgültige Ziel unserer Überlegungen zu sein: Wenn wir nämlich die Determinante in der Leibnizschen Darstellung

$$\det A = \sum_{\sigma \in S_n} \operatorname{sgn} \sigma \cdot a_{1,\sigma(1)} \cdots a_{n,\sigma(n)}$$

als Summe über alle Permutationen $\sigma \in S_n$ der Menge $\{1, \ldots, n\}$ berechnen, beträgt der *Aufwand* zur Berechnung von $\det A$ immerhin $n \cdot n!$ Operationen. Selbst mit der rekursiven Bestimmung über Unterdeterminanten nach dem Laplaceschen Entwicklungssatz

$$\det A = \sum_{i=1}^{n} (-1)^{i+1} a_{1i} \det A_{1i}$$

sind 2^n Operationen auszuführen, wobei $A_{1i} \in \mathrm{Mat}_{n-1}(\mathbb{R})$ die Matrix bezeichnet, die aus A durch Streichen der ersten Zeile und der i-ten Spalte entsteht. Wie wir sehen werden, sind alle im folgenden beschriebenen Verfahren bereits für $n \geq 3$ effektiver als die Cramersche Regel. *Schnelligkeit* bei der Lösung des gewünschten Problems ist also sicher eine zweite wichtige Eigenschaft eines „guten" Algorithmus.

Bemerkung 1.2. Von einem guten Rechenverfahren erwarten wir sicherlich, dass es die gestellte Aufgabe mit möglichst geringem Aufwand (an Rechenoperationen) löst. Intuitiv gibt es zu jedem Problem einen minimal nötigen Aufwand, den wir als *Komplexität* des Problems bezeichnen. Ein Algorithmus ist umso effektiver, je näher der benötigte Rechenaufwand an der Komplexität des Problems liegt. Der Rechenaufwand eines konkreten Algorithmus ist also immer eine *obere* Schranke für die Komplexität eines Problems. Die Berechnung *unterer* Schranken ist im allgemeinen wesentlich schwieriger – für Details sei das Buch von J. Traub und H. Wozniakowski [83] genannt.

Die Schreibweise $x = A^{-1}b$ könnte den Gedanken aufkommen lassen, zur Berechnung der Lösung von $Ax = b$ zunächst die inverse Matrix A^{-1} zu berechnen und diese dann auf b anzuwenden. Die Berechnung von A^{-1} beinhaltet jedoch die Schwierigkeiten der Lösung von $Ax = b$ für *sämtliche* rechten Seiten b. Wir werden im zweiten Kapitel sehen, dass die Berechnung von A^{-1} „bösartig" sein kann, auch wenn sie für spezielle b „gutartig" ist. Bei der Notation $x = A^{-1}b$ handelt es sich daher nur um eine formale Schreibweise, die nichts mit der tatsächlichen Berechnung der Lösung x zu tun hat. Man sollte deshalb tunlichst vermeiden, von der „Invertierung von Matrizen" zu reden, wenn eigentlich die „Lösung linearer Gleichungssysteme" gemeint ist.

Bemerkung 1.3. Es gab über lange Zeit die offene Wette eines wissenschaftlich hochkarätigen Kollegen, der eine hohe Summe darauf setzte, dass in praktischen Fragestellungen niemals das Problem der „Invertierung einer Matrix" unvermeidbar auftritt. Soweit bekannt, hat er seine Wette in allen Einzelfällen gewonnen.

Auf der Suche nach einem effektiven Lösungsverfahren für beliebige lineare Gleichungssysteme werden wir im folgenden zunächst besonders einfach zu lösenden Spezialfälle studieren. Der einfachste ist sicherlich der Fall einer *diagonalen* Matrix A: dabei zerfällt das Gleichungssystem in n voneinander unabhängige skalare Gleichungen. Die Idee, ein allgemeines System in ein diagonales umzuformen, liegt dem *Gauß-Jordan-Verfahren* zugrunde. Da es jedoch weniger effektiv als das in

Kapitel 1.2 beschriebene Verfahren ist, lassen wir es hier weg. Der nächst schwierigere Fall ist der eines *gestaffelten Gleichungssystems*, den wir unmittelbar anschließend behandeln wollen.

1.1 Auflösung gestaffelter Systeme

Hier betrachten wir den Fall eines *gestaffelten Gleichungssystems*

$$\begin{array}{ccccccccc} r_{11}x_1 & + & r_{12}x_2 & + & \cdots & + & r_{1n}x_n & = & z_1 \\ & & r_{22}x_2 & + & \cdots & + & r_{2n}x_n & = & z_2 \\ & & & & \ddots & & \vdots & & \vdots \\ & & & & & & r_{nn}x_n & = & z_n, \end{array}$$

in Matrix-Vektor-Notation kurz

$$Rx = z, \tag{1.1}$$

wobei R eine *obere Dreiecksmatrix* ist, d. h. $r_{ij} = 0$ für alle $i > j$. Offenbar erhalten wir x durch rekursive Auflösung, beginnend mit der Zeile n:

$$\begin{array}{lll} x_n & := z_n/r_{nn}, & \text{falls } r_{nn} \neq 0, \\ x_{n-1} & := (z_{n-1} - r_{n-1,n}x_n)/r_{n-1,n-1}, & \text{falls } r_{n-1,n-1} \neq 0, \\ \vdots & & \\ x_1 & := (z_1 - r_{12}x_2 - \cdots - r_{1n}x_n)/r_{11}, & \text{falls } r_{11} \neq 0, \end{array}$$

Nun gilt für die obere Dreiecksmatrix R, dass $\det R = r_{11} \cdots r_{nn}$ und daher

$$\det R \neq 0 \iff r_{ii} \neq 0 \quad \text{für alle } i = 1, \ldots, n.$$

Der angegebene Algorithmus ist also wiederum (wie bei der Cramerschen Regel) genau dann anwendbar, wenn $\det R \neq 0$, also unter der Bedingung des Existenz- und Eindeutigkeitssatzes. Für den Rechenaufwand ergibt sich:

a) für die i-te Zeile: je $n - i$ Additionen und Multiplikationen, eine Division

b) insgesamt für die Zeilen n bis 1:

$$\sum_{i=1}^{n}(i-1) = \frac{n(n-1)}{2} \doteq \frac{n^2}{2}$$

Multiplikationen und ebenso viele Additionen.

Dabei steht das Zeichen „$\doteq$“ für „gleich bis auf Terme niedrigerer Ordnung“, d. h. wir betrachten nur den für große n entscheidenden Term mit dem größten Exponenten.

Vollkommen analog lässt sich ein gestaffeltes Gleichungssystem der Form

$$Lx = z \tag{1.2}$$

mit einer *unteren Dreiecksmatrix* L lösen, indem man jetzt in der ersten Zeile beginnt und sich bis zur letzten Zeile durcharbeitet.

Diese Auflösung gestaffelter Systeme heißt im Fall von (1.1) *Rückwärtssubstitution* und im Fall von (1.2) *Vorwärtssubstitution*. Der Name Substitution, d. h. Ersetzung, rührt daher, dass der Vektor der rechten Seite jeweils komponentenweise durch die Lösung ersetzt werden kann, wie wir es in dem folgenden Speicherschema für die Rückwärtssubstitution andeuten wollen:

$$\begin{gathered}
(z_1, z_2, \ldots, z_{n-1}, z_n) \\
(z_1, z_2, \ldots, z_{n-1}, x_n) \\
\vdots \\
(z_1, x_2, \ldots, x_{n-1}, x_n) \\
(x_1, x_2, \ldots, x_{n-1}, x_n).
\end{gathered}$$

Der Fall der Vorwärtssubstitution ist gerade umgekehrt.

1.2 Gaußsche Eliminationsmethode

Wir wenden uns nun dem effizientesten der klassischen Verfahren zur Auflösung eines linearen Gleichungssystems zu, der *Gaußschen Eliminationsmethode*. Carl Friedrich Gauß (1777 – 1855) beschreibt sie 1809 in seiner Arbeit über Himmelsmechanik „Theoria Motus Corporum Coelestium“ [36] mit den Worten „die Werte können mit der üblichen Methode der Elimination erhalten werden“. Er benutzte dort das Verfahren im Zusammenhang mit der Methode der kleinsten Fehlerquadrate (siehe Kapitel 3). Allerdings hatte Lagrange die Eliminationsmethode bereits 1759 vorweggenommen und in China war sie bereits im ersten Jahrhundert vor Christus bekannt.

Kehren wir also wieder zurück zum allgemeinen Fall eines linearen Gleichungssystems

$$\begin{array}{ccccccc}
a_{11}x_1 & + & a_{12}x_2 & + \cdots + & a_{1n}x_n & = & b_1 \\
a_{21}x_1 & + & a_{22}x_2 & + \cdots + & a_{2n}x_n & = & b_2 \\
\vdots & & \vdots & & \vdots & & \vdots \\
a_{n1}x_1 & + & a_{n2}x_2 & + \cdots + & a_{nn}x_n & = & b_n
\end{array} \tag{1.3}$$

und versuchen, es in ein gestaffeltes umzuformen. Zielen wir auf eine obere Dreiecksmatrix ab, so muss die erste Zeile nicht verändert werden. Die restlichen Zeilen wollen wir so behandeln, dass die Koeffizienten vor x_1 verschwinden, d. h. die Variable x_1 aus den Zeilen 2 bis n *eliminiert* wird. So entsteht ein System der Art

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a'_{22}x_2 + \cdots + a'_{2n}x_n &= b'_2 \\ \vdots \qquad\qquad & \quad \vdots \\ a'_{n2}x_2 + \cdots + a'_{nn}x_n &= b'_n . \end{aligned} \tag{1.4}$$

Haben wir das erreicht, so können wir dasselbe Verfahren auf die letzten $n-1$ Zeilen anwenden und so rekursiv ein gestaffeltes System erhalten. Es genügt daher, den ersten Eliminationsschritt von (1.3) nach (1.4) zu untersuchen. Wir setzen voraus, dass $a_{11} \neq 0$. Um den Term $a_{i1}x_1$ in Zeile i $(i = 2, \ldots, n)$ zu eliminieren, subtrahieren wir von der Zeile i ein Vielfaches der unveränderten Zeile 1, d. h.

$$\text{Zeile } i \text{ neu} := \text{Zeile } i - l_{i1} \cdot \text{Zeile } 1$$

oder explizit

$$\underbrace{(a_{i1} - l_{i1}a_{11})}_{=0} x_1 + \underbrace{(a_{i2} - l_{i1}a_{12})}_{=a'_{i2}} x_2 + \cdots + \underbrace{(a_{in} - l_{i1}a_{1n})}_{=a'_{in}} x_n = \underbrace{b_i - l_{i1}b_1}_{=b'_i} .$$

Aus $a_{i1} - l_{i1}a_{11} = 0$ folgt sofort $l_{i1} = a_{i1}/a_{11}$. Damit ist der erste Eliminationsschritt unter der Annahme $a_{11} \neq 0$ ausführbar. Das Element a_{11} heißt *Pivotelement*, die erste Zeile *Pivotzeile*. (Der aus dem Englischen stammende Begriff „pivot“ lässt sich dabei am besten mit „Dreh- und Angelpunkt“ übersetzen.) In den Zeilen 2 bis n bleibt nach diesem ersten Eliminationsschritt eine $(n-1, n-1)$-Restmatrix stehen. Wir sind damit in einer Situation wie zu Beginn, allerdings um eine Dimension kleiner.

Wenden wir auf diese Restmatrix die Eliminationsvorschrift erneut an, so erhalten wir eine Folge

$$A = A^{(1)} \to A^{(2)} \to \cdots \to A^{(n)} =: R$$