Kapitel 1

Einführung

Die Numerische Mathematik ist eine etwas andere Art von Mathematik, als sie bisher in der Vorlesung geboten wurde. Sie ist vor allem für die Lösung von Problemen in den Anwendungen von entscheidender Bedeutung.

1.1 Aufgabenstellungen und Ziele der Numerischen Mathematik

Viele Probleme, die in der Mathematik auftreten, kann man mit ,,rein mathematischen Methoden" nur sehr schwer oder gar nicht lösen. Beispiele sind:

- die Berechnung vieler bestimmter Integrale, da man die Stammfunktion nicht findet,
- die Berechnung von Nullstellen von Funktionen, da man nicht nach der unbekannten Variablen auflösen kann,
- die Lösung großer linearer Gleichungssystem per Hand ist sehr zeitaufwendig,
- die Lösung vieler Differentialgleichung geht nicht analytisch.

Innerhalb der Numerischen Mathematik werden Verfahren (Algorithmen) zur approximativen Lösung dieser Probleme entwickelt, welche mit Hilfe von Computern abgearbeitet werden können. Wichtige Fragestellungen, die im Rahmen der Numerischen Mathematik untersucht werden, sind:

- Wie teuer sind diese Verfahren? Wie lange dauert ihre Abarbeitung?
- Wie genau ist das berechnete Ergebnis?
- Unter welchen Bedingungen konvergieren Interationsverfahren? Wie schnell ist die Konvergenz?
- Wie robust sind die Verfahren gegenüber Datenfehlern?

In dieser Vorlesung kann aus Zeitgründen auf Details zur Beantwortung dieser Fragen nicht eingegangen werden. Es werden lediglich wichtige Antworten präsentiert.

1.2 Computerzahlen und numerische Verfahren

Auf einem Computer kann nur eine endliche Menge von reellen Zahlen dargestellt werden. Zahlen, die man nicht darstellen kann, müssen gerundet werden. Deswegen ist jede Rechenoperation mit Computerzahlen potentiell mit Rundungsfehlern behaftet.

Alle modernen Computer nutzen zur Darstellung von Computerzahlen die Ba-

sis 2, das heißt das Binärsystem. In dieser Basis wird eine Zahl $x \in \mathbb{R}$ durch

$$x = \sum_{k=-\infty}^{\infty} x_k 2^k \tag{1.1}$$

dargestellt. Die Koeffizienten x_k werden Ziffern genannt. Im Binärsystem hat man nur die Ziffern $x_k \in \{0, 1\}$. Zum Vergleich, im gewöhnlichen Dezimalsystem hat man

$$x = \sum_{k=-\infty}^{\infty} y_k 10^k$$

mit den Ziffern $y_k \in \{0, 1, \ldots, 9\}$. Auf einem Computer kann man nur Zahlen mit endlich vielen Ziffern darstellen, das bedeutet, anstelle von (1.1) hat man

$$x = \sum_{k=-m}^{M} x_k 2^k, \quad 0 < m, M < \infty.$$

Beispiel 1.1 *Binärdarstellungen einiger Zahlen.*

$$\begin{aligned} 5 &= 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0 = 101, \\ 0.5 &= 1 \cdot 2^{-1} = 0.1, \\ 123.75 &= 1 \cdot 2^6 + 1 \cdot 2^5 + 1 \cdot 2^4 + 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 1 \cdot 2^{-2} \\ &= 1111011.11. \end{aligned}$$

□

Moderne Programmiersprachen (C, C++, FORTRAN90) unterscheiden zwischen ganzen Zahlen (integer) und Fließkommazahlen (double). Zur Speicherung von Integerzahlen werden im allgemeinen 4 Byte = 32 Bit genutzt. Ein Bit braucht man für das Vorzeichen, so dass man noch 31 Bits für Ziffern hat. Für double–Zahlen werden im allgemeinen 8 Byte = 64 Bit genutzt. Dabei verwendet man die Darstellung

$$x = \pm m 2^e,$$

wobei man ein Bit für das Vorzeichen braucht, im allgemeinen 52 Bit für die Ziffern m (Mantisse) und 11 Bit für den Exponenten e verwendet.

Infolge der endlich vielen Computerzahlen gelten nicht mehr alle bekannten Rechengesetze. Das trifft zum Beispiel auf das Assoziativitätsgesetz der Addition zu. Man erhält auf dem Computer

$$1 + 10^{20} - 10^{20} = 0, \quad 1 + \left(10^{20} - 10^{20}\right) = 1.$$

(*mit MATLAB demonstrieren.*)

Falls man numerische Verfahren im Detail untersuchen will, muss man wissen, wie sich die Rundungsfehler auf das Ergebnis auswirken. Solche Untersuchungen können hier aus Zeitgründen nicht präsentiert werden. Es stellt sich heraus, dass die meisten Operationen gutartig sind, das heißt, kleine Rundungsfehler in den Eingangsdaten führen nur zu kleinen Fehlern in den berechneten Ergebnissen. Es gibt jedoch eine wichtige Ausnahme: Die Subtraktion zweier fast gleich großer Zahlen kann zu einem großen relativen Fehler im Ergebnis führen, siehe Übungsaufgabe. Dieses Phänomen nennt man Auslöschung.

Die Kosten der Verfahren werden oft in der Anzahl der für die Berechnung benötigten Operationen (Addition, Subtraktion, Multiplikation, Division) gemessen. Diese Operationen nennt man Flops (floating point operations).

1.3 Klassifizierung von Problemen

Wir werden in dieser Vorlesung nur korrekt gestellte Problem betrachten. Ein Problem heißt korrekt gestellt, falls

- es eine eindeutige Lösung besitzt,
- die Lösung stetig von den Daten abhängt.

Anderenfalls heißt das Problem schlecht gestellt. Schlecht gestellte Probleme, zum Beispiel sogenannte inverse Probleme, spielen in der Praxis eine große Rolle. Ihre numerische Behandlung ist jedoch ziemlich kompliziert und jenseits dem Anliegen dieser Vorlesung.

Stetige Abhängigkeit von den Daten bedeutet, dass kleine Störungen in den Daten nur kleine Änderungen in der Lösung bewirken. Diese Eigenschaft ist sehr wichtig für die numerische Lösung von Problemen, da man durch das Runden eigentlich immer kleine Störungen der Daten hat. Man kann ein abstraktes Maß definieren, welches die Störungen der Lösung in Abhängigkeit von den Datenstörungen misst – die sogenannte Konditionszahl. Wir werden dieses Maß im Spezialfall bei der Lösung linearer Gleichungssysteme kennenlernen. Grob gesprochen ist die Konditionszahl der Quotient des maximalen Lösungsfehlers und eines vorgegebenen Datenfehlers. Das heißt, eine kleine Konditionszahl ist günstig. Korrekt gestellte Probleme mit großer Konditionszahl werden schlecht konditioniert genannt.