Qualität im wissenschaftlichen Rechnen

Seminar Modellierung und Simulation

Betreuer: Julian Kunkel

Wintersemester 2012/2013

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Fakultät für Mathematik, Informatik und Naturwissenschaften
Universität Hamburg

Feifan Chen
5804977

Inhaltsverzeichnis

Tabellenverzeichnis

1. Einleitung

1.1 Definition des wissenschaftlichen Rechnen

Unter dem Begriff „wissenschaftlichen Rechnen“ wird das Gebiet verstanden, das sich mit der computergestützten numerischen Simulation von Prozessen beschäftigt, wie sie in den Ingenieur-, Natur- oder Wirtschafts- und Sozialwissenschaften untersucht werden.[1]

In fast allen Bereichen der Wissenschaft oder des Ingenieurwesens werden solche Verfahren benötigt, die die Lösungen liefern, welche für mathematische Modelle physikalischer Sachverhalte bereitstellen. Zum Beispiel die alltägliche Berechnung der Flugbahnen für Erdsatelliten oder planetarische Missionen, die Überprüfung der bautechnischen Sicherheit der Luftfahrzeuge, die Kartierung des Genoms sowie die Analysierung der Finanzderivate.[2]

Die mathematischen Theorien und Techniken, die als numerische Analysis bezeichnet werden, bilden den Hauptanteil am wissenschaftlichen Rechnen. Sie stellt die Sprache zur Verfügung, in der die Modelle beschrieben werden, und dient als Grundlage der Modellierung und des Entwurfs von Rechenmethoden und Algorithmen. Währenddessen sind die elektronischen Rechner auch ein wichtiger Teil des Lösungsprozesses. Sie vereinfachen die Modellierung sowie die Implementierung des Modells und erhöhen deren Effizienz, indem sie einen Teil der numerischen Methoden ersetzen und Leistungsfähigkeit und Rechengenauigkeit gewährleisten können. Dabei werden Programmiersprachen, Betriebssysteme, Datenverwaltung etc. aus dem Informatik Gebiet verwendet. Wissenschaftliches Rechnen stützt sich somit auf die Mathematik und Informatik. Es umfasst alle Hilfsmittel, Fertigkeiten und Theorien, die man dazu benötigt, Probleme der Wissenschaft und Ingenieurkunst in Form mathematischer Modelle auf einem Rechner zu lösen.[3]

1.2 Problemstellung

Wie im letzten Abschnitt beschrieben wurde, wissenschaftliches Rechnen wird in immer mehreren Bereichen benötigt. Es unterstützt direkt die Problemlösung und spielt eine

[1] Vgl. Bungartz, H./ Zenger, C. (1998).
[2] Vgl. Golub, G./ Ortega, J. (1995), S. 1; Einarsson, B. (2005), S. xxi.
[3] Vgl. Golub, G./ Ortega, J. (1995), S. 2-3, PaSCo (2004), S.6.

wesentliche unerlässliche Rolle in den wissenschaftlichen Forschungen und Anwendungen.

Daher sind die Genauigkeit und Zuverlässigkeit einer wissenschaftlicher Software bzw. Methode auch notwendig und haben eine große Bedeutung. Es ist jedoch schwierig, es zu erreichen. Neben den generellen Problemen bei der Software-Entwicklung existieren auch Schwierigkeiten von der numerischen Berechnung:

- Approximation treten auf allen Ebenen auf.
- Stetige Funktionen werden diskretisiert implementiert.
- Unbegrenzte Prozesse werden durch begrenze Prozesse ersetzt.
- Reelle Zahlen werden durch Zahlen endlicher Genauigkeit repräsentiert. Zahlen können nur bis zu einer begrenzten Genauigkeit berechnet werden.[4]

Dadurch können Fehler im mathematischen Teil der wissenschaftlichen Software nicht vermieden werden. Jedoch ist es möglich, die Fehler zu regeln und zu steuern, um Genauigkeit und Zuverlässigkeit der wissenschaftlichen Software zu garantieren. Die Eigenschaften und die Fortpflanzung der Fehler müssen untersucht, analysiert und verstanden werden.[5]

Im Rahmen des Seminars „Modellierung und Simulation" beschäftigt sich diese Ausarbeitung mit den häufig vorkommenden Fehlern in der wissenschaftlichen Software sowie mit der Definition und Bestimmung der Qualität einer wissenschaftlichen Software. Im Kapitel zwei werden die Hauptprobleme aus der mathematischen Seite vorgestellt, die bei der Numerischen Berechnung oft auftreten. Die Softwarefehler aus der Programmierseite werden nicht betrachtet. Im Kapitel drei wird erklärt, was unter Qualität im wissenschaftlichen Rechnen verstanden werden soll und wie es bewertet werden soll, mit anderen Worten wie die Genauigkeit und Zuverlässigkeit einer wissenschaftlichen Software beurteilt werden kann.

---

[4] Vgl. Einarsson, B. (2005), S. xxi.
[5] Vgl. Einarsson, B. (2005), S. xxi.

2. Fehler und Probleme in der wissenschaftlichen Software

In diesem Kapitel werden die Fehler unterschiedlicher Art in der wissenschaftlichen Software vorgestellt, die den mathematischen Hintergrund enthalten und aus der Beschränktheit der elektronischen Geräte, in dem Fall Rechner entstehen. Ferner wird gezeigt, was für Auswirkung diese Fehler auf die Ergebnisse bringen und welche Probleme in der Software sie verursachen.

2.1 Grundfehler bei der numerischen Berechnung

Zwei Hauptprobleme treten im wissenschaftlichen Rechnen auf:

- Die genauen Eingabewerte sind nicht bekannt.
- Manche Berechnungen werden nicht richtig durchgeführt.[6]

Daraus entstehen vier Grundfehler bei der numerischen Berechnung:

- **Rundungsfehler**

Wegen der Speicher- und Arbeitsweise des Computers sind nicht alle reellen Zahlen auf dem Rechner exakt darstellbar. Die exakt darstellbaren Zahlen werden Maschinenzahlen genannt. Die sonstigen Zahlen werden gerundet und als Maschinenzahlen dargestellt. Außerdem nicht immer sind die Ergebnisse der arithmetischen Operationen von Maschinenzahlen auch genau Maschinenzahlen. Solche Ergebnisse werden ebenfalls gerundet. So gibt es Unterschiede zwischen einer reellen Zahl x und ihrer näherungsweisen Darstellung x_{rd} als Maschinenzahl und zwischen den exakten arithmetischen Grundoperationen +, -, *, / für reelle Zahlen und den tatsächlichen verfügbaren näherungsweisen Operationen $+_{rd}$, $-_{rd}$, $*_{rd}$, $/_{rd}$ für Maschinenzahlen. Nach jeder Operation wird ein geringer Rundungsfehler erzeugt und wird in der nächsten Berechnung einkalkuliert, wo der neue Rundungsfehler noch zusätzlich produziert wird. So pflanzen sich die Fehler fort und das Endergebnis kann schließlich stark abweichen.[7]

Beispiel 2.1[8]: Folgender MATLAB Code iteriert von a bis b mit n Schritte.

[6] Vgl. Einarsson, B. (2005), S. 4.
[7] Vgl. Zulehner, W. (2008), S. 75-77; Einarsson, B. (2005), S. 4.
[8] Dieses Beispiel gilt in der Norm IEEE 754, doppelte Genauigkeit. Mit anderer Genauigkeit kann ein unterschiedliches Ergebnis herauskommen.

```
function step(a,b,n)
% step from a to b with n steps
h=(b-a)/n;
x=a;
disp(x)
while x < b,
  x = x + h;
  disp(x)
end
```

a, b, n sind vorgegeben. Die Schleife soll genau n mal durchgeführt werden.
Mit $a = 1$, $b = 2$, $n = 3$ läuft die Schleife jedoch viermal, also einmal zu viel. Die Ursache liegt drin: $h = (2-1)/3 = 1/3$. Als eine nicht abbrechende Dezimalzahl wird h auf 0,333… abgerundet. Nach drei Schritten bleibt $x = 1 + 0{,}333*3 = 1{,}999…$, d.h. $x < 2$. Das Kriterium $x < b$ ist immer noch erfüllt und die Schleife läuft weiter.[9]
Mit $a = 1$, $b = 1{,}1$, $n = 3$ läuft die Schleife wie erwartet dreimal. Nun ist $h = (1{,}1-1)/3 = 1/30$. x und b sind beide Dezimalzahlen und werden wie gewünscht verglichen. Das Ergebnis ist dann auch richtig.[10]
Für solche Schleifen ist empfohlen, reelle Variablen statt ganzzahligen Variablen zu verwenden, um den Rundungsfehler während der Operation zwischen ganzzahligen und reellen Variablen zu vermeiden. Für dieses Beispiel kann `while x < b` in der sechsten Zeile durch `while x < b-h/2` ersetzt werden, damit die Variable x immer mit einer nicht abbrechende Dezimalzahl verglichen wird, in dem Fall dass es selber auch eine nicht abbrechende Dezimalzahl ist.[11]

- **Kürzbarkeit**

Um diesen Fehler genauer zu beschreiben, muss zuerst der Begriff „Kürzbarkeit“ kurz erklärt werden:

[9] Vgl. Einarsson, B. (2005), S. 4.
[10] Vgl. Einarsson, B. (2005), S. 4.
[11] Vgl. Einarsson, B. (2005), S. 4.

Gegeben sei ein Magma (M, *).

Ein Element $c \in M$ heißt linkskürzbar, wenn für alle $a, b \in M$ gilt:

c*a = c*b folgt dann a = b.

Ein Element $c \in M$ heißt rechtskürzbar, wenn für alle $a, b \in M$ gilt:

a*c = b*c folgt dann a = b.

Ein Element $c \in M$ heißt kürzbar, wenn c links- und rechtskürzbar ist.

Eine Halbgruppe (S,*) heißt kürzbar, wenn jedes $c \in S$ kürzbar ist.[12]

Die Menge der reellen Zahlen mit üblicher Addition/Multiplikation/Subtraktion/Division ist kürzbar.

Fehler der Kürzbarkeit treten bei der Subtraktion von zwei Zahlen auf, die fast gleich groß sind. Es liegt dran, dass die geringe Ungenauigkeit durch so eine Subtraktion relativ große Auswirkung auf das Ergebnis hat. Zum Beispiel sei x_1 = 1,243±0,0005, x_2 = 1,234±0,0005. x_1 - x_2 = 0,009±0,001. Die kleine Abweichung spielt im Ergebnis jedoch eine große Rolle.[13]

Beispiel 2.2[14]: Die Lösung für die allgemeine quadratische Gleichung $a x^2 + bx + c = 0$ ($a \neq 0$) lautet:

$$x_{1,2}^{\alpha} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Unter der Bedingung $c \neq 0$ kann es in einen anderen Ausdruck umgewandelt werden:

$$x_{1,2}^{\beta} = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}}$$

Die beiden Ausdrücke sind mathematisch äquivalent, numerisch aber nicht.

Sei $a = 1{,}0 \cdot 10^{-5}$, $b = 1{,}0 \cdot 10^3$, $c = 1{,}0 \cdot 10^3$, durch Computer ausgerechnet werden jeweilig:

$$x_1^{\alpha} = -3{,}0518,\ x_2^{\alpha} = -1{,}0000 \cdot 10^8$$

$$x_1^{\beta} = -1{,}000,\ x_2^{\beta} = -3{,}2768 \cdot 10^7$$

[12] Vgl. Wikipedia.

[13] Vgl. Einarsson, B. (2005), S. 4.

[14] Dieses Beispiel gilt in der Norm IEEE 754, einfache Genauigkeit.

Die Wurzelpaare derselben Gleichung sind ganz unterschiedlich. Die richtigen Lösungen davon sind x_2^{α} und x_1^{β}, die jeweils aus $\frac{-b-\sqrt{b^2-4ac}}{2a}$ und $\frac{-2c}{b+\sqrt{b^2-4ac}}$, wo eine Addition von b und $\sqrt{b^2-4ac}$ durchgeführt ist, berechnet werden. Der Grund dafür ist: b^2 ist viel größer als $|4ac|$. So ist $\sqrt{b^2-4ac}$ annähernd $|b|$. In diesem Fall ist Quadratwurzel von 10^6-0,04 gleich 999,9999799999998, welche fast gleich b bzw. 1000 ist. Bei der Subtraktion der zwei nahezu gleichen Werte dominiert der Fehler bzw. Ungenauigkeit bei der Berechnung der Quadratwurzel. So ist das Gesamtergebnis auch fehlerhaft. [15]

Für den Fehler der Kürzbarkeit ist Nutzung der doppelten bzw. vierfachen Genauigkeit auch kein Ausweg. Eine Lösung wäre, Algorithmus zu wechseln oder anzupassen. Beispielsweise hier soll anhand des Vorzeichens von b entschieden werden, welche Formel benutzt werden soll.[16]

- **Rekursion**

Ein häufig benutztes mathematisch Verfahren im wissenschaftlichen Rechnen ist, eine Berechnung auf das Ergebnis der vorheriger Berechnung zu basieren und die gleiche Berechnung weiterhin iterativ oder rekursiv durchzuführen. Mit der Wiederholung können Fehler akkumulieren und schließlich die ganze Berechnung zerstören.[17]

- **Ganzzahlüberlauf**

Auf dem Computer werden alle Zahlen in Form der binären Zahlen intern gespeichert. Eine Speicherstelle (eine 1 oder 0) wird als ein Bit bezeichnet. Auf dem heutigen Rechensystem werden 32- und 64-Bit-Architekturen benutzt, d.h. die Speicherstellenanzahl ist begrenzt. Die größte darstellbare Zahl ist 2^{64}-1 = 18.446.744.073.709.551.615. Wenn das Rechenergebnis mehr Stellen erfordert, tritt ein Fehler auf, der Ganzzahlüberlauf genannt ist.[18]

---

[15] Vgl. Einarsson, B. (2005), S. 4-5.
[16] Vgl. Einarsson, B. (2005), S. 5.
[17] Vgl. Einarsson, B. (2005), S. 5.
[18] Vgl. Polifke, W./ Kopitz, J. (2009), S. 509.

Beispiel 2.3: Auf einem 32-Bit-Betriebssystem wird die Fakultätsfunktion als Arithmetik mit Integern implementiert. Bis 12! kann richtig ausgerechnet werden. Das Ergebnis für 13! ist inkorrekt. Bei 17! wird eine negative Zahl ausgegeben.[19]

Bei einfacher oder doppelter Genauigkeit ist die Grenze weiter. Der Ganzzahlüberlauf bei der Fakultätsfunktion erscheint jeweils ab 35! und 171!. Aber damit die Funktionen nicht überlaufen, soll die Berechnung der Fakultätsfunktion am besten ganz ausgeschlossen sein. Z.B. bei der Taylorformel, statt $\frac{(x-a)^n}{n!}$ zu berechnen, soll eine Multiplikation von $\frac{x-a}{n}$ und dem bereits bekannten Wert von $\frac{(x-a)^{n-1}}{(n-1)!}$ durchgeführt werden, um die Berechnung von $n!$ zu vermeiden.[20]

2.2 Probleme in den Programmen

Die numerischen Fehler entstehen aus mathematischer Struktur. Die draus erzeugten Bugs sind schwierig zu diagnostizieren und zu beheben, insbesondere aus der Sicht der Softwareentwicklungstechnik. Solche Fehler bringen negativen Einfluss auf die Zuverlässigkeit und Genauigkeit der wissenschaftlichen Software und verursachen anomales Verhalten bzw. Probleme in den Programmen:

– *Unangemessene Behandlung mit den nicht abbrechenden Dezimalzahlen.* Die nicht abbrechenden Dezimalzahlen können nur als Dezimalzahlen mit endlichen Dezimalstellen kodiert. Z. B. π als 3,14159 oder 1/9 als 0,1111. Diese Beschränkung wirkt nachteilig auf Genauigkeit und Übertragbarkeit der Software aus.

– *Überprüfung der Gleichheit.* Die Rundung und die Approximation erzeugen weitere Schwierigkeiten bzw. Fehlergebnisse bei der Überprüfung der Gleichheit zwei Variablen, wie im Beispiel 2.1. Es muss deswegen definiert werden, wann bzw. mit Erfüllung welcher Kriterien zwei Variablen als gleich betrachtet werden dürfen.

– *Inkonsistente Genauigkeit.* Die Norm IEEE enthalten Formate mit

[19] Vgl. Einarsson, B. (2005), S. 5.
[20] Vgl. Einarsson, B. (2005), S. 6.

unterschiedlichen Genauigkeiten. Bei der Berechnung von gemischten Konstanten oder Variablen mit verschiedenen Genauigkeiten wird die niedrige Genauigkeit genommen. D.h. falls das Ergebnis der Operationen eine hohe Genauigkeit benötigt, müssen auch alle betroffene Konstanten und Variablen mit mindestens dieser Genauigkeit deklariert werden.

- *Nicht offensichtlicher falscher Code*. Manche fehlerhafte Programmiercodes können trotzdem fehlerfrei laufen und ein Ergebnis liefern, das als akzeptierbar erkannt ist. Ein kleiner Fehler könnte im arithmetischen Ausdruck vorhanden sein, so dass der Code immer noch zu der richtigen Lösung führt aber mehr Zwischenschritte und Rechenzeit verbraucht. Solche Fehler sind auch schwierig zu entdecken.
- *Fehlende Erkennung der schlecht konditionierten Probleme.* Manche gestellte Probleme, die mit der Software geforscht und gelöst werden sollen, sind schlecht konditioniert. D.h. kleine Fehler wie Rundungsfehler oder kleine Änderungen von dem Problem können zu einer ganz anderen Lösung führen, die die richtige Lösung überhaupt nicht annähert. Eine robuste Software soll erkennen und die Benutzer davor warnen, wenn das Problem schlecht konditioniert ist.
- *Instabile Algorithmen.* Eine Methode ist instabil, falls die Rundungsfehler im gesamten Verfahren nicht kontrolliert sind. Manche Methoden sind nur stabil von dem Eingabewert in einem bestimmten Wertebereich. In so einer Situation sollen die Benutzer per Fehlermeldungen benachrichtigt werden, falls die Eingabe außerhalb dieses Wertebereiches liegt.
- *Falsche Abbruchkriterien.* Siehe Beispiel 2.1.[21]

[21] Vgl. Einarsson, B. (2005), S. 22-23.

3. Qualität einer wissenschaftlicher Software

„Gute Qualität" zu definieren ist sehr abstrakt. Ein Standard wurde dafür eingeführt. In diesem Kapitel werden erklärt, welche Bestandteile den Begriff „Qualität" bilden und wie die Bestimmung der Qualität einer wissenschaftlichen Software durch eine Formel festgelegt ist.

3.1 Eigenschaften

Was für Bedeutung die Qualität einer wissenschaftlichen Software genauer enthält ist schwierig nachvollzuziehen und zu beschreiben. Die folgenden vier grundlegenden Eigenschaften der Algorithmen sind dabei entscheidende Elemente:

- **Genauigkeit**

Die Genauigkeit bezieht sich auf die Anzahl der Dezimalstellen, die für das Rechnen, die Eingabe und die Ausgabe verwendet wird[22]. Es bezeichnet, wie genau das gestellte Problem untersucht wird und die gelieferte Lösung ist.

- **Zuverlässigkeit**

Mit der Zuverlässigkeit wird gezählt, wie oft (in Prozent) die Software ausfällt, in dem Sinne, dass der herausgekommene Fehler größer als der erlaubte bzw. geplante Fehler ist.[23]

- **Exaktheit**

Die Exaktheit bezieht sich auf die absoluten oder relativen Fehler von einer approximativen Größe. Absoluter Fehler ergibt Differenz zwischen dem berechneten Ergebnis x und dem eigentlichen Wert x^*:

$$|x - x^*|$$

Relativer Fehler repräsentiert das Verhältnis des absoluten Fehlers zum eigentlichen Wert:

$$\frac{|x - x^*|}{|x^*|}$$

[22] Einarsson, B. (2005), S. 21, eigene Übersetzung.
[23] Einarsson, B. (2005), S. 21, eigene Übersetzung.

Relativer Fehler gilt nicht für eine x^* annähernd 0. In diesem Fall kann nur absoluter Fehler verwendet werden. Die Exaktheit ist eine Messung der Qualität des Ergebnisses. Die Berechnung des relativen und absoluten Fehlers wird im Beispiel 3.1 mitgezeigt.[24]

Die Exaktheit und die Genauigkeit werden oft verwechselt. Sie werden unterschiedlich gerechnet und haben auch unterschiedliche Bedeutungen. Sie müssen nicht übereinstimmend sein. Hohe Genauigkeit bedeutet nicht hohe Exaktheit.[25]

- **Stabilität**

Eine Methode, die die Exaktheit der Lösung genauso garantiert wie die Daten es gewährleistet, heißt stabil. D.h. wenn der Eingabewert exakt ist, soll die Methode ebenfalls ein exaktes Ergebnis mit geringem relativem bzw. absolutem Fehler liefern. Die Stabilität befasst sich mit der Empfindlichkeit einer Methode bei den Fehlern, besonders bei dem Rundungsfehler im Laufe der Rechnung.[26] (Siehe auch Abschnitt 2.2 - Instabile Algorithmen)

Beispiel 3.1: Aus der folgenden quadratischen Gleichung

$$1{,}6\,x^2 - 100{,}1x + 1{,}251 = 0$$

soll durch die Lösungsformel

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

ergeben

$$x_1 = 62{,}53,\; x_2 = 0{,}03125.$$

Durch die Überprüfung mit der Beziehungsformel

$$x_1 x_2 = c / a$$

und mit $x_1 = 62{,}53$ ergibt

$$x_2 = 0{,}01251$$

[24] Vgl. Einarsson, B. (2005), S. 21.
[25] Vgl. Einarsson, B. (2005), S. 21.
[26] Vgl. Einarsson, B. (2005), S. 56.

welche ganz unterschiedlich als das Ergebnis der Lösungsformel ist. Wegen der Kürzbarkeit ist die Lösung inexakt. Die richtige Lösung lautet

$$x_1 = 62{,}55,\ x_2 = 0{,}0125.$$

So sind absoluter Fehler und relativer Fehler der Lösung x_2 jeweils 0,01874 und 150%.[27]

Stabilität ist die wichtigste Eigenschaft, die die determinierende Rolle bei der Berechnung der Qualität spielt. Die Stabilitätsanalyse wird im Abschnitt 3.2 weiter erklärt.

3.2 Bestimmung der Qualitätsquote und Stabilität

Messung der Qualität einer wissenschaftlichen Software kann durch eine Formel realisieren werden. Bevor die Formel vorgestellt wird, wird zuerst eine Einführung der Grundlagen für die Formel gegeben.

- **Maschinengenauigkeit**

Ein wichtiger Begriff ist die Maschinengenauigkeit eps mit folgender Definition:

$$\text{eps} = \inf\{ x > 0 : rd(1 + x) \neq 1\}$$

dabei bezeichnet die Operation $rd(x)$ die Rundung von x zur nächstgelegenen Maschinenzahl. eps ist das Infimum bzw. die größte untere Schranke der positiven reellen Zahlen, die zu 1 addiert von der Rundung wahrgenommen werden. eps wird berechnet durch die Formel:

$$\text{eps} = \frac{b^{1-m}}{2}$$

dabei ist $b \in \aleph \backslash \{1\}$ die Basis bzw. die Grundzahl des Zahlensystems und m die Mantissenlänge. eps charakterisiert das Auflösungsvermögen des Rechners. Die Rechenergebnisse sind damit enge verbunden bzw. davon abhängig.[28]

Beispiel 3.2: In der Tabelle 3.1 wird ein Beispiel von einem Gleitpunkt-Zahlensystem mit Basis $b = 10$ und Mantissenlänge $m = 6$ gezeigt. Für das Zahlensystem ergibt sich

$$\text{eps} = \frac{b^{1-m}}{2} = \frac{10^{1-6}}{2} = \frac{1}{2} \cdot 10^{-5}$$

[27] Vgl. Einarsson, B. (2005), S. 56.
[28] Vgl. Hanke-Bourgeois, M. (2009), S. 17-18; Dahmen, W./ Reusken, A. (2008), S. 36-39.

x	$rd(x)$
$\frac{1}{3} = 0{,}3333333\ldots$	$0{,}333333 \cdot 10^{0}$
$\sqrt{2} = 1{,}41421356\ldots$	$0{,}141421 \cdot 10^{1}$
$e^{-10} = 0{,}00004539927\ldots$	$0{,}453999 \cdot 10^{-4}$
$e^{10} = 22026{,}46579\ldots$	$0{,}220265 \cdot 10^{5}$
$\frac{1}{10} = 0{,}1$	$0{,}100000 \cdot 10^{0}$

Tabelle 3.1 Ein Gleitpunkt-Zahlensystem mit Basis zehn und Mantissenlänge sechs.
Quelle: Dahmen, W./ Reusken, A. (2008), S. 38, modifiziert.

- **Fehleranalyse**[29]

In meisten Fällen wird ein reales Problem numerisch dargestellt nur mit einer Approximation als Lösung von der Berechnung. Es sei ein numerischer Algorithmus $\hat{f}$ für Lösung eines Problems $x \mapsto f(x)$ mit $x \in D(f) \subset \Re$. $\hat{x}$ ist die implementierte Lösung, welche die exakte Lösung x annähert. $|\Delta x|$ ist die Differenz zwischen $\hat{x}$ und x. $\hat{f}(x) = f(\hat{x}) = f(x + \Delta x)$.[30]

Absoluter und relativer Fehler von $\hat{f}(x)$, also $\left|\hat{f}(x) - f(x)\right|$ und $\frac{\left|\hat{f}(x) - f(x)\right|}{|f(x)|}$ werden als Vorwärtsfehler bezeichnet. Die Vorwärtsfehler zeigen Differenz zwischen den berechneten und exakten Ergebnissen, wie gut die berechnete Lösung $\hat{x}$ das zu lösende Problem erfüllt. $|\Delta x|$ und $\left|\frac{\Delta x}{x}\right|$ sind jeweils als absoluter und relativer Rückwärtsfehler

[29] Diese Arbeit basiert auf Einarsson, B. (2005). Jedoch ist die Definition der Fehleranalyse in dieser Literatur anders wie in allen anderen Literaturen. Vorwärts- sowie Rückwärtsfehleranalyse sind umgekehrt definiert. In dieser Arbeit wird die generelle Definition verwendet.
[30] Vgl. Higham, N. (2002), S.6-8; Hanke-Bourgeois, M. (2009), S. 20.

definiert. Rückwärtsfehler interpretieren die berechnete Näherung als exakte Lösung mit gestörten Eingangsdaten $\hat{x}$ und untersucht die Störung $|\Delta x|$.[31]

- **Die Qualitätsquote**

Die arithmetische Qualität eines Algorithmus, einer numerischen Methode oder einer wissenschaftlichen Software hängt von dem Rückwärtsfehler ab. Je kleiner es ist, desto zuverlässiger ist der Algorithmus. Der Idealfall wäre, dass nur die Fehler im Laufe der Rechnung auftreten, die wegen der Durchführung der numerischen Berechnung auf dem Rechner unvermeidbar sind. Diese Fehler bleiben auf der Ebene der Maschinengenauigkeit. Daher ist die Qualitätsquote eines Algorithmus an der Stelle $\tilde{x}$ definiert als

$$J(\tilde{x}) = \frac{B(\tilde{x})}{eps}$$

Wegen der Eigenschaft der Maschinengenauigkeit, $J(\tilde{x}) \geq 1$. Optimaler Wert wird herausgegeben wenn der Rückwärtsfehler gleich die Maschinengenauigkeit bleibt. Also das beste Ergebnis der Qualitätsquote ist 1.[32]

Ein Algorithmus wird als optimal zuverlässig erkannt mit $J \approx 1$. Es wird auch als rückwärts stabil genannt[33]. Anderenfalls besitzt ein Algorithmus schlechte Qualität, wenn die kleinste Konstante C, die die Ungleichung $B(\tilde{x}) \leq C \cdot eps$ löst, zu groß ist.[34]

[31] Vgl. Higham, N. (2002), S.6-8; Hanke-Bourgeois, M. (2009), S. 20.
[32] Vgl. Einarsson, B. (2005), S. 98.
[33] In vielen Literatur z.B. Hanke-Bourgeois, M. (2009) steht auch, wenn die kleinste Konstante C, die die Ungleichung $B(\tilde{x}) \leq C \cdot eps$ löst, nicht zu groß ist, wird der Algorithmus rückwärts stabil genannt.
[34] Vgl. Einarsson, B. (2005), S. 98.

4. Fazit

In dieser Seminarausarbeitung wurde eine grobe Einführung in das Qualitätssystem im wissenschaftlichen Rechnen gegeben. Eine Begriffserklärung diente zuerst als theoretische Grundlagen. Dabei wurde erwähnt, wofür wissenschaftliches Rechnen eingesetzt und verwendet wird und was für eine wichtige Bedeutung es im Wissenschaften und Ingenieurwesen besitzt. Danach wurde zusammengefasst, welche Fehler bei der numerischen Berechnung häufig vorkommen, welche Probleme in der wissenschaftlichen Software draus entstehen und wie sie negativ auswirken. Dabei wurden viele Beispiele für besseres Verständnis aufgezählt. Im nächsten Kapitel wurden Eigenschaften der wissenschaftlichen Software vorgestellt, wie sie definiert sind und was sie genauer beschreiben. Zum Schluss wurde die Formel der Qualitätsquote erklärt, mit dem die Qualität eines Algorithmus bzw. einer Methode bewertet werden kann.

Als ein elektronisches Gerät bringt der Rechner einerseits hervorragende Leistungen und Ressourcen, die mit Hand- und Tischrechner nicht erreichbar werden. Es leitete ein neues Zeitalter ein. Andererseits sind Speicher begrenzt und somit sind die Operationen auch beschränkt. Fast alle Probleme von der wissenschaftlichen Software liegen grundsätzlich dran. Solche Fehler sind zurzeit nicht vermeidbar. Bei dem Entwurf und Umsetzung kann jedoch versucht werden, mithilfe der anderen Möglichkeiten die Fehler zu minimieren, und somit zuverlässige Software bzw. Verfahren zu entwickeln.

Literaturverzeichnis

Bungartz, H./ Zenger, C. (1998): *Wissenschaftliches Rechnen - eine interdisziplinäre Disziplin.* In: Aviso: Zeitschrift für Wissenschaft & Kunst in Bayern, 1/98.

Dahmen, W./ Reusken, A. (2008): *Numerik für Ingenieure und Naturwissenschaftler*. 2. Aufl., Springer-Verlag, Berlin, Heidelberg.

Einarsson, B. (2005): *Accuracy and Reliability in Scientific Computing.* Society for Industrial and Applied Mathematics, Philadelphia.

Golub, G./ Ortega, J. (1995): *Wissenschaftliches Rechnen und Differentialgleichungen: eine Einführung in die numerische Mathematik.* Heldermann Verlag, Berlin.

Hanke-Bourgeois, M. (2009): *Grundlagen der Numerischen Mathematik und des Wissenschaftlichen Rechnens*. 3. Aufl., Vieweg+Teubner, Wiesbaden.

Higham, N. (2002): *Accuracy and stability of numerical algorithms.* 2. Aufl. Society for Industrial and Applied Mathematics, Philadelphia.

PaSCo (2004): *Bericht des Paderborn Institute for Scientific Computation (PaSCo).* PaSCo, Paderborn. URL: http://www.pasco.uni-paderborn.de/report.php Stand 30.3.2013.

Polifke, W./ Kopitz, J. (2009): Wärmeübertragung: Grundlagen, analytische und numerische Methoden. 2. Aufl., Pearson Studium, München.

Wikipedia: *Kürzbarkeit.* URL: http://de.wikipedia.org/wiki/K%C3%BCrzbarkeit Stand 30.3.2013.

Zulehner, W. (2008): *Numerische Mathematik: Eine Einführung anhand von Differentialgleichungsproblemen. Band1: Stationäre Probleme.* Birkhäuser, Basel.

Eidesstattliche Erklärung

Ich, Feifan Chen, erkläre hiermit an Eides Statt, dass ich vorliegende Arbeit selbstständig und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsordnung vorgelegt und auch nicht veröffentlicht.