Mathematische Werkzeuge für Computergrafik 2016/17

Gleitkommzahlen

1 Grundlagen[1]

Da im Computer nur endliche Ressourcen zur Verfügung stehen, können reelle Zahlen in vielen Fällen nicht exakt dargestellt werden. Dezimaldarstellungen in der Form

$$x_1 \cdots x_m, x_{m+1} \cdots x_n = x_1, x_2 \cdots x_n \times 10^{m-1}, \tag{1a}$$

wie zum Beispiel

$$124,26749 = 1,2426749 \times 10^2, \tag{1b}$$

sind grundsätzlich für eine Verwendung im Computer geeignet, da es ausreicht, die Zahlenfolge $x_1 \cdots x_n$ und die Position des Kommas zu speichern. Durch die endliche Menge von Speicher, können irrationale Zahlen, wie zum Beispiel π, und rationale Zahlen mit einer "sehr langen" Dezimaldarstellung allerdings nicht exakt im Computer abgebildet werden. Für beliebige reelle Zahlen ist also numerisch, d.h. im Computer, nur eine approximative Darstellung möglich. Verschiedene Ansätze für eine solche Darstellung existieren. Bei jedem dieser Ansätze wird versucht, für einen bestimmten Anwendungsfall den Approximationsfehler sowohl für die Darstellung von Zahlen als auch die Durchführung von Berechnungen möglichst gering zu halten, bei möglichst geringem Aufwand.

Festkommazahlen Bei Festkommazahlen wird die Position des Kommas nicht verändert. Damit hat eine Zahl die Darstellung:

$$x_1, x_2 \cdots x_n, \tag{2}$$

wobei im Computer die Folge $x_1 \cdots x_n$ gespeichert wird und x_1 per Konvention vor dem Komma steht. Die feste Position des Kommas hat bei Operationen mit „sehr“ großem oder „sehr“ kleinem Ergebnis, relativ zur ursprünglichen Größenordnung, allerdings Überläufe oder den Verlust von Genauigkeit zur Folge, das heißt, man erhält Ergebnisse, welche mit der gewählten Kommaposition nicht mehr oder nur sehr ungenau darstellbar sind.

[1] Dieses Material wurde zum Teil für die Veranstaltung "Wissenschaftliches Rechnen" an der TU Berlin (2014-2016) erarbeitet.

Gleitkommazahlen Im Gegensatz zu Festkommazahlen wird bei Gleitkommazahlen, basierend auf der Größenordnung der darzustellenden Zahl, die Position des Kommas verschoben. Dies hat den Vorteil, dass man einen größeren Wertebereich abdecken kann, die Genauigkeit der Darstellung hängt dann allerdings von der Größenordnung der Zahl ab.

Eine Gleitkommazahl besteht aus drei Teilen:

Basis b: Die Basis bestimmt bezüglich welcher Basis die Zahlen dargestellt werden. Zum menschlichen Verständnis ist $b = 10$ am besten geeignet. Der Computer verwendet $b = 2$, d.h. eine binäre Darstellung.

Mantisse m: Die Mantisse $m = (m_1, \cdots, m_n)$ enthält die Ziffern der darzustellenden Zahl mit $m_i \in \{0, \cdots, b-1\}$ und $m_1 > 0$. Die Bedingung $m_1 > 0$ stellt eine sinnvolle Wahl des Exponenten e sicher, da dieser dann exakt der Größenordnung der Zahl entspricht.

Exponent e: Der Exponent $e = (e_1, \cdots, e_k)$ mit $e_i \in \{0, \cdots, b-1\}$, speichert die Position des Kommas und damit die Größenordnung der Zahl.

Mit diesen Teilen stellt man eine Gleitkommazahl wie folgt dar:

$$\pm \underbrace{m_1, \cdots m_n}_{\text{Mantisse}} \times \underbrace{b^{\pm e}}_{\text{Basis}} \in \mathbb{G}(b, n, k) \tag{3}$$

wobei wir mit $\mathbb{G}(b, n, k)$ die Menge aller Gleitkommazahlen meinen, welche mit gewählten Parametern b,n,k darstellbar sind (wenn b,n, k sich nicht ändern, dann werden wir oft $\mathbb{G}$ schreiben ohne die Abhängigkeit von den Parameters des Formats explizit aufzuführen).

Bei der Darstellung einer Gleitkommazahl im Computer wird immer die Basis $b = 2$ verwendet und muss deshalb nicht explizit gespeichert werden.[2] Eine IEEE 64-Bit Gleitkommazahl hat zum Beispiel im Speicher die folgende Repräsentation:

$$\underbrace{\pm}_{\text{1-Bit Vorzeichen}} \quad \underbrace{e_1\ e_2 \cdots e_{11}}_{\text{11-Bit Exponent}} \quad \underbrace{1, m_1\ m_2 \cdots m_{52}}_{\text{52-Bit Mantisse}}.$$

Der Fehler, welcher bei der Abbildung einer beliebigen reellen Zahl $x \in \mathbb{R}$ in eine Gleitkommazahl $\hat{x} \in \mathbb{G} \equiv \mathbb{G}(b, n, k)$ entsteht, hängt von den gewählten Parametern b,n,k ab. Wir nehmen im Folgenden an, dass eine beliebige reelle Zahl immer auf die, entsprechend den arithmetischen Regeln (was wohl-definiert ist, da $\mathbb{G} \subset \mathbb{R}$), nächste Zahl $\hat{x} \in \mathbb{G}$ abgebildet wird, d.h.

$$G \ : \ \mathbb{R} \mapsto \mathbb{G} \ : \ x \mapsto \mathrm{rd}_{\mathbb{G}}(x) = \hat{x}; \tag{4}$$

[2] Diese Aussage bezieht sich auf die Darstellung, welche im Prozessor verwendet wird. In Software können beliebige Gleitkommaformate emuliert bzw. verwendet werden. In Python gibt es zum Beispiel das `Decimal` Modul, welches Gleitkommazahlen mit einer beliebigen Basis darstellen kann, siehe https://docs.python.org/2/library/decimal.html#module-decimal.

dies wird auch in der Praxis im Computer meist verwendet. Wie bei der Rundung üblich tritt der maximale Fehler durch $G : \mathbb{R} \to \mathbb{R}$ genau dann auf, wenn man in der Mitte zwischen zwei benachbarten Gleitkommazahlen ist, siehe Abbildung 1. Um den maximalen Rundungsfehler zu verstehen, müssen wir

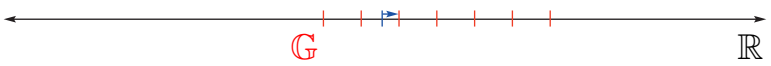

Abbildung 1: Beliebige reelle Zahlen (blau) werden auf die nächste Gleitkommazahl (rot) gerundet.

also den Abstand zweier Gleitkommazahlen bestimmen. Für zwei beliebige, benachbarte Gleitkommazahlen ist der Abstand, wie üblich, durch ihre Differenz gegeben. Damit habe wir (wir nehmen hier an, dass kein Überlauf im letzten Bit auftritt; dass diese Annahme hier sinnvoll ist, wird in Kürze klar werden):

$$\begin{array}{rccccccc} & m_1 & , & m_2 & \cdots & (m_n+1) & \times & b^e \\ - & m_1 & , & m_2 & \cdots & m_n & \times & b^e \\ \hline & 0 & , & 0 & \cdots & 1 & \times & b^e \end{array} \tag{5}$$

wobei die Rechnung analog zur schriftlichen Subtraktion erfolgt (wenn der Exponent nicht gleich wäre, so müsste dieser zunächst angepasst werden, in dem beide Zahlen bezüglich des größeren dargestellt würden). Um erneut eine gültige Gleitkommazahl zu erhalten, für welche gilt $m_1 > 0$, muss das Komma um $n-1$ Stellen verschoben werden. Damit erhalten wir für die Differenz

$$1 \quad , \quad 0 \quad \cdots \quad 0 \quad \times \quad b^{e-(n-1)}. \tag{6}$$

Um eine konkrete Vorstellung über den Abstand benachbarter Gleitkommazahlen zu erhalten, wählen wir $b = 10$ und $n = 1$, d.h. ein in der Praxis wenig geeignetes aber intuitives Gleikommazahl-Format mit einer Ziffer und Basis 10. Für den Abstand erhalten wir damit:[3]

e	Abstand
-2	0.01
-1	0.1
0	1.0
1	10.0
2	100.0

Ein wenig überraschend hängt der Abstand also vom Exponenten ab. Zum Beispiel haben Gleitkommazahlen in $[1, 10)$ einen Abstand von 1, in $[10, 100)$ aber einen Abstand von 10 usw. Wir sehen also, dass es nicht ***einen*** Abstand

[3]Es sei daran erinnert, dass, auf Grund der Bedingung $m_1 > 0$, der Exponent nicht frei wählbar ist, sondern eine inhärente Eigenschaft der Zahl ist.

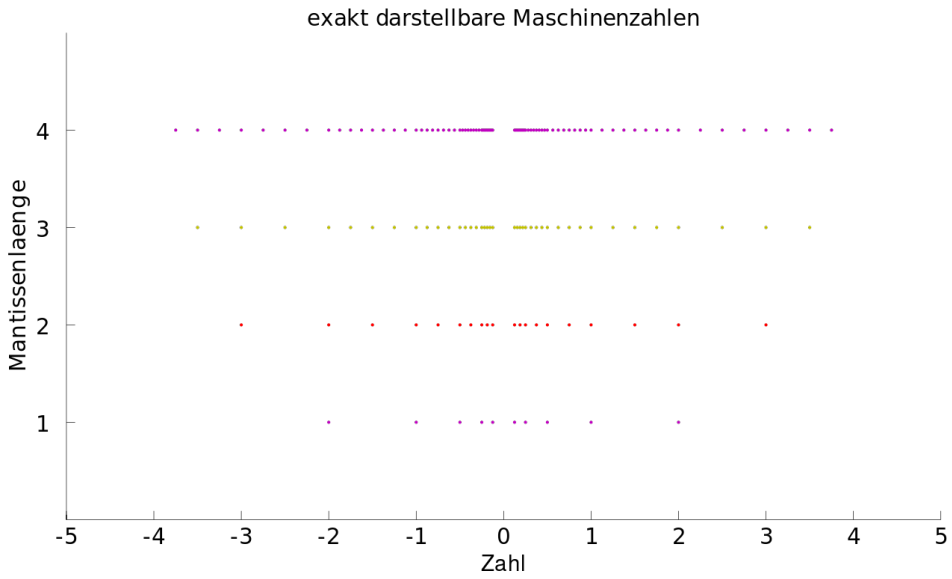

Abbildung 2: Abstände von Gleitkommazahlen mit Basis $b = 2$ (Quelle: http://upload.wikimedia.org/wikipedia/de/0/0b/Gleitkommazahlen.svg).

zwischen benachbarten Gleitkommazahlen gibt, sondern dass dieser von der Magnitude der Zahlen abhängt. Dies ist in Abbildung 2 graphisch dargestellt. Daraus folgt, dass der maximale Fehler für das Runden $G : \mathbb{R} \to \mathbb{G}$ in die Gleitkommazahlen auch von der Größenordnung abhängt. Zum Beispiel haben wir für den maximalen Rundungsfehler:

$$\begin{aligned} G(2.5) &= 3 \qquad |2.5 - G(2.5)| = 0.5 \\ G(25) &= 30 \qquad |25 - G(25)| = 5 \end{aligned}$$

Die Abhängig des Abstandes von Gleitkommazahlen und des Rundungsfehlers von der Magnitude der darzustellenden Zahl ist nicht zufällig, sondern stellt einen Kompromiss dar zwischen der Abdeckung eines großen Wertebereiches in $\mathbb{R}$ (für 64-bit Gleitkommazahlen hat man $\hat{x}_{min} \approx 2.2251 \times 10^{-308}$ und $\hat{x}_{max} = 1.7977 \times 10^{308}$) und einer nicht von der Größe der Zahl abhängigen Anzahl von Bits zu dessen Darstellung. Der Kompromiss basiert dabei auf der Beobachtung, dass in sehr vielen Anwendungen der tolerierbare Fehler von der Magnitude der Ergebnisse abhängt. Zum Beispiel, wenn die darzustellende Zahl $6.0 \in \mathbb{R}$ ist und man stellt diese als $10 \in \mathbb{G}$ dar, dann ist der Fehler für die allermeisten Anwendungen vollkommen inakzeptabel. Will man jedoch $1000006.0 \in \mathbb{R}$ darstellen und dies wird als $1000000 \in \mathbb{G}$ abgebildet, so ist dies in vielen Fällen vertretbar. Eine solche Abhängigkeit des Fehlers von der Größenordnung einer Zahl wird im relativen Fehler formalisiert, welchen wir im

Folgenden betrachten.

2 Relativer vs. absoluter Fehler

Man unterscheidet bei Gleitkommazahlen zwischen *relativem Fehler* und *absolutem Fehler*. Der absolute Fehler ist der numerische Wert des tatsächlichen Fehlers, wohingegen der relative Fehler die Größe des Fehlers in Relation zum Ergebnis beschreibt. Somit misst der relative Fehler die Wichtigkeit des Fehlers im Bezug zum Ergebnis. Mit ihm kann die Frage beantwortet werden, ob eine Abweichung, egal ob sie sehr klein oder sehr groß ist, für das Ergebnis akzeptabel ist. Die Fehler sind wie folgt definiert:

$$\text{Absoluter Fehler: } E_a = |x_a - x|$$

$$\text{Relativer Fehler: } E_r = \frac{|x_a - x|}{|x|}$$

wobei x der korrekte (oder ursprüngliche) Wert ist und x_a die Approximation.

Um sowohl den absoluten als auch den relativen Fehler besser zu verstehen, betrachten wir zwei Beispiele in denen leicht falsche Lösung gegeben sind. Diese Abweichungen sollen als durch Rundung verursachte Fehler verstanden werden:

$$\begin{aligned}
\text{Rechnung: } & 1 + 1? \\
\text{„Ergebnis”: } & 3. \\
\text{Absoluter Fehler: } & E_a = |3 - 2| = 1 \\
\text{Relativer Fehler: } & E_r = \frac{|3 - 2|}{|2|} = 0.5 = 5 \cdot 10^{-1}
\end{aligned}$$

$$\begin{aligned}
\text{Rechnung: } & 10000 + 10000? \\
\text{„Ergebnis”: } & 20001. \\
\text{Absoluter Fehler: } & E_a = |20001 - 20000| = 1 \\
\text{Relativer Fehler: } & E_r = \frac{|20001 - 20000|}{|20000|} = 0.00005 = 5 \cdot 10^{-5}
\end{aligned}$$

Wir sehen, dass der absolute Fehler in beiden Rechnungen identisch ist. Der relative Fehler ist in der zweiten Rechnung jedoch sehr viel kleiner als in der ersten, was der intuitiven Idee entspricht, dass der absolute Fehler in der zweiten Rechnung für die meisten Anwendungen vernachlässigbar klein ist.

3 Genauigkeit von Gleitkommazahlen

Die maximale Genauigkeit einer Gleitkommazahl-Darstellung wird Maschinengenauigkeit ϵ genannt (obwohl sie heutzutage nicht mehr von der Maschine, d.h. der Hardware, abhängig ist, sondern lediglich von der Anzahl der verwendeten Bits). Zwei äquivalente Definition der Maschinengenauigkeit existieren:

1. Die Maschinengenauigkeit ist (bis auf eine Konstante) die kleinste Zahl δ, so dass $(1+\delta)$ nicht wieder auf 1 gerundet wird, d.h.

$$\epsilon = C \operatorname{argmin}_{\delta \in \mathbb{G}} \big\{ G(1+\delta) > 1 \big\}. \tag{7}$$

2. Die Maschinengenauigkeit ist der maximale *relative* Fehler, welcher sich durch die Rundung einer reellen Zahl in das Gleitkommaformat ergibt,

$$\epsilon = \max_{x \in \mathbb{R}_{\mathbb{G}}} \frac{|x - G(x)|}{|x|} \tag{8}$$

wobei $\mathbb{R}_{\mathbb{G}}$ den Teilbereich der reellen Zahlen $\mathbb{R}$ bezeichnet, welcher durch die kleinste und größte Gleitkommazahl $\mathbb{G}_{\text{min}}$ und $\mathbb{G}_{\text{max}}$ im Format $\mathbb{G}(b,n,k)$ beschränkt ist, d.h. $\mathbb{R}_{\mathbb{G}} = \big\{ x \in \mathbb{R} \mid \mathbb{G}_{\text{min}} < x < \mathbb{G}_{\text{max}} \big\}$.

Die Konstante C in der ersten Definition hängt vom Rundungsmodus ab, welcher für $G : \mathbb{R} \mapsto \mathbb{G}$ verwendet wird:

Rundungsmodus	C
round-to-nearest	2
floor	1

Für den zweite Teil der Definition stellt sich die Frage, ob ϵ von der Magnitude von x abhängt. Verwenden wir floor als Rundungsmodus, so haben wir, dass das Maximum von $|x - G(x)|$ der Abstand zweier benachbarter Gleitkommazahlen ist, welchen wir bereits in Gleichung 6 berechnet haben, d.h. $|x - G(x)| = 1.0 \times b^{e-(n-1)}$. Der Gesamtausdruck wird dann maximiert, wenn der Nenner minimal ist und also den Wert $1.0 \times b^e$ hat. Die Maschinengenauigkeit ist damit also:

$$\epsilon = \max_{x \in \mathbb{R}} \frac{|x - G(x)|}{|x|} = \frac{1.0 \times b^{e-(n-1)}}{1.0 \times b^e} = 1.0 \times b^{-n+1}. \tag{9}$$

Wir sehen, dass der *relative* Fehler von $G : \mathbb{R} \mapsto \mathbb{G}$ *unabhängig* von der Magnitude von x ist und nur von der Anzahl der Ziffern n in der Mantisse abhängt. Diese Unabhängigkeit zeigt erneut, dass Gleitkommazahlen eine "natürliche" approximative Darstellung von reellen Zahlen sind, wenn man den relativen Fehler als relevantes Fehlermaß betrachtet. Es ist auch einfach zu sehen, dass die beiden Definitionen der Maschinengenauigkeit in Gleichung 7 und Gleichung 8 in der Tat äquivalent sind. Verwenden wir weiterhin floor als Rundungsmodus, dann gilt $G(1+\delta) > 1$ genau dann, wenn δ die nächste Gleitkommazahl nach 1 ist. Mit Gleichung 6 hat δ den Wert

$$\delta = 1.0 \times b^{-n+1}, \tag{10}$$

was in der Tat mit Gleichung 9 übereinstimmt.

Neben der notwendigen Rundung, um eine beliebige reelle Zahl im Gleitkommaformat darstellen zu können, treten weiter Fehler bei der Durchführung von Berechnungen mit Gleitkommazahlen auf. Zum Beispiel gilt selbst für die

elementaren arithmetischen Operationen, dass das Ergebnis im Allgemeinen keine Gleitkommazahl ist, d.h.

$$G(G(x) * G(y)) \neq G(x) * G(y) \neq G(x * y), \tag{11}$$

wobei "$*$" hier für eine beliebige elementare arithmetische Operation, d.h. Addition, Subtraktion, Multiplikation oder Division, steht. Ausserdem gelten im Allgemeinen weder das Assoziativ- noch das Distributivgesetz. Ein Beispiel hierfür ist in Abbildung 3 gegeben. Für Addition, Multiplikation und Division

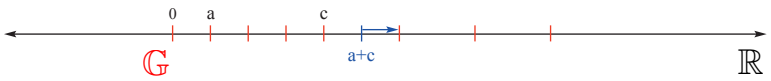

Abbildung 3: Notwendige Rundung bei der Addition, um für $a + c$ wieder eine gültige Gleitkommazahl zu erhalten.

ist die Größenordnung des Fehlers durch die Maschinengenauigkeit gegeben. Wie wir im Folgenden sehen, kann bei der Subtraktion die sogenannte Auslöschung auftreten und ein beliebig großer Fehler entstehen.

3.1 Auslöschung

Beginnen wir mit einem Beispiel. Gegeben sei $\mathbb{G}(10, 3, 3)$ und drei Zahlen a, b, c, die wie folgt dargestellt werden:

	$\mathbb{R}$	$\mathbb{G}$	rel. Fehler
a	1.22	1.22×10^0	0.0
b	3.34	3.34×10^0	0.0
c	2.28	2.28×10^0	0.0

In der 3-Ziffer Gleitkomma-Darstellung wollen wir $b^2 - 4\,a\,c$ berechnen, was zur Lösung einer quadratischen Gleichung bestimmt werden muss. Wir nehmen an, dass zunächst b^2 und $4\,a\,c$ berechnet werden, und dann die Differenz gebildet wird. Für die Zwischenergebnisse erhalten wir:

	$\mathbb{R}$	$\mathbb{G}$	rel. Fehler
b^2	11.1556	1.12×10^1	0.00398
$4\,a\,c$	11.1264	1.11×10^1	0.00237

Für die Gleitkomma-Darstellung des obigen Ergebnisses muss eine Rundung erfolgen, da nur drei Ziffern für die Mantisse zur Verfügung stehen. Der relative Fehler, welcher durch die Rundung entsteht, ist jedoch klein, so dass er für die meisten Anwendungen als vernachlässigbar gelten kann. Um das Ergebnis zu erhalten, muss noch eine Subtraktion ausgeführt werden. Hierfür erhalten wir:

	$\mathbb{R}$	$\mathbb{G}$	rel. Fehler
$b^2 - 4\,a\,c$	0.0292	1.00×10^{-1}	2.42466

Trotz des sehr kleinen relativen Fehlers für b^2 und $4\,a\,c$ im Zwischenergebnis, erhalten wir durch die Subtraktion ein Ergebnis, welches für die meisten Anwendungen unbrauchbar ist. Diese Potenzierung des relativen Fehlers bei der Subtraktion wird als ***Auslöschung*** (oder "catastrophic cancellation") bezeichnet.

Die Ursache für die Auslöschung liegt in den verschiedenen Größenordnungen der Argumente der Subtraktion und von dessen Ergebnis. Die Subtraktion löscht dabei alle Ziffern am Anfang der Mantisse aus, welche für Minuend und Subtrahend gleich sind. Die Genauigkeit des Ergebnisses wird damit nur durch die Ziffern bestimmt, in welchen sich die Terme unterscheiden. Durch den Unterschied in der Größenordnung von Eingabe und Ausgabe der Subtraktion können sich jedoch nur wenige Ziffern am Ende unterscheiden (ansonsten wäre das Ergebnis nicht wesentlich kleiner als Minuend und Subtrahend). Im obigen Beispiel gehen uns durch die Subtraktion zwei Ziffern verloren, da Minuend und Subtrahend in den ersten beiden übereinstimmen. Das Ergebnis hat dadurch effektiv nur noch eine Ziffer, welche Informationen enthält (die verbliebenen, nicht-Null Ziffern werden auch als "significant digits" bezeichnet).

Auslöschung tritt also immer dann auf, wenn zwei Zahlen subtrahiert werden, welche sich nur geringfügig unterscheiden, d.h. wenn die Differenz wesentlich kleiner ist, als die Argumente für die Subtraktion. Interessanter Weise tritt bei keiner anderen der verbleibenden grundlegenden arithmetischen Operationen, d.h. Addition, Multiplikation und Division, eine vergleichbare Potenzierung des relativen Fehlers auf.