Staatliche Studienakademie Leipzig

Hausarbeit

Numerische Differentiation

ausgeführt am Fachbereich Informatik der
Staatlichen Studienakadmie Leipzig

durch

Thomas Fiedler, MA
Mat.Nr.: 5000345

Inhaltsverzeichnis

1 Einleitung

In der vorliegenden Hausarbeit soll die numerische Differentiation anhand eines Beispiels näher betrachtet werden. Konkret soll die erste Ableitung der Funktion 1 an der Stelle $x_0 = 1$ numerisch bestimmt werden.

$$f(x) = y(x) = \sqrt{\sin x^2 + ln(2 + x^2)} \tag{1}$$

Der Vollständigkeit halber ist der Graph der Funktion in Abbildung 1 dargestellt:

Mathematisch ist die erste Ableitung an der Stelle x_0 der Anstieg, der an diesen Punkt angelegten Tangente. Es liegt nun nahe, die Tangente durch eine Sekante des zu betrachtenden Graphen zu approximieren. Dazu wählt man zwei Punkte auf der Kurve, die in der Nachbarschaft von x_0 liegen, bzw. $(x_0, f(x_0))$ selbst und einen weiteren Punkt und berechnet daraus den Anstieg der Sekante mittels:

$$\frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x} \tag{2}$$

Aus diesem sog. Diffenrenzenquotienten lässt sich nun die Definition der ersten Ableitung herleiten:

$$\lim_{h \to 0} \frac{f(x_0 + h) - f(x_0)}{h} h = x - x_0 \tag{3}$$

Mit Hilfe dieser Gleichung ist es bereits möglich, den Wert der ersten Ableitung zu approximieren. Die Wahl eines geeigneten Wertes für h stellt allerdings ein Problem dar. Da Stetigkeit Voraussetzung für die Differenzierbarkeit ist, ist davon auszugehen, dass sich $f(x_0 + h)$ und $f(x_0)$ für genügend kleine h nur unwesentlich unterscheiden. Das führt allerdings dazu, dass sich bei einer Subtraktion die meisten signifikanten Stellen auslöschen und es teilweise zu inakzeptabel großen Rundungsfehlern kommt. Daher ist es angebrachter, den sog. symmetrischen Differenzenquotienten zu verwenden. Dazu werden Punkte links und rechts der Stelle x_0 gewählt und der Anstieg der durch diese Punkte definierte Sekante bestimmt. Es ergibt sich:

$$\frac{f(x_0 - h) - f(x_0 + h)}{2h} \tag{4}$$

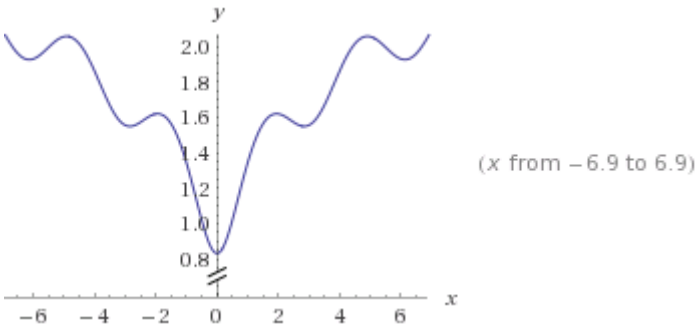

Abbildung 1: Funktionsgraph der gegebenen Funktion

Durch den größeren Abstand der beiden Punkte voneinander sind die auftretenden Rundungsfehler nicht so verheerend, wie bei Anwendung von Funktion 2. Der Approximationsfehler kann mit

$$R = \frac{-f^{(3)}(\xi)}{6} h^2 \tag{5}$$

angegeben werden, ist also abhängig von h^2. Dabei ist ξ eine reele Zahl zwischen $x_0 - h$ und $x_0 + h$. Damit stellt der symmetrische Differenzenquotient die verlangte Approximation in Fehlerordnung h^2 dar. Die ausführliche Betrachtung der Fehler finden in Abschnitt 3 statt.

Um die Genauigkeit zu erhöhen, können auch andere Verfahren zum Einsatz kommen. Statt zwei kann man fünf Punkte der Kurve verwenden. Für eindimensionale Betrachtungen sind die x-Koordinaten der Punkte dann entsprechend: $\{x_0 - 2h, x_0 - h, x, x_0 + h, x_0 + 2h\}$. Wird für $f(x)$ an diesen Stellen die Taylor-Entwicklung durchgeführt, erhält man ein Gleichungssystem mit vier Gleichungen. Die Lösung des Gleichungssystems führt zu:

$$f'(x) \approx \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + f(x-2h)}{12h} \tag{6}$$

Der Approximationsfehler ergibt sich ebenfalls aus der Taylor-Entwicklung und ist mit

$$R = \frac{-f^{(5)}(\xi)}{30} h^4 \tag{7}$$

angegeben.

Welche Ergebnisse die Anwendung der beiden Verfahren für die gegebene Funktion erbringt, wird im nächsten Abschnitt 2 vorgestellt.

2 Lösung

Die in der Aufgabenstellung gegebene Funktion 1 lässt sich umformen und unter Einsatz der bekannten Differenzierungsregeln analytisch ableiten und es ergibt sich Funktion 8.

$$\frac{dy}{dx} = \frac{\frac{2x}{x^2+2} + sin(2x)}{2\sqrt{ln(x^2+2) + sin^2(x)}} \tag{8}$$

Mit Hilfe dieser Gleichung ist es möglich, die erste Ableitung an der Stelle $x_0 = 1$ in Maschinengenauigkeit zu berechnen. Das entsprechende Ergebnis ist somit lediglich im Rahmen der üblichen, der Maschine inhärenten Rundungsfehler fehlerbehaftet.
Die beiliegende Java-Umsetzung des in Abschnitt 1 dargelegten numerischen Lösungsverfahrens gibt zum Vergleich ebenfalls das „exakte“ Ergebnis aus. Im Rahmen der Maschinengenauigkeit beträgt dies 0.5862394204568043 und mit der geforderten künstlichen Begrenzung der Mantissenlänge auf 12 Stellen ergibt sich $f'(x_0) = 0.586239420457$.

Das Programm führt anschließend die numerische Ermittlung von $f'(x_0 = 1)$ sowohl für die Fehlerordnung 2 als auch für Fehlerordnung 4 durch und gibt die Ergebnisse für
$h = 10^{-1}, 10^{-2}, \ldots, 10^{-13}$ mit Mantissenlänge 12 aus. Tabelle 1 fasst die entsprechenden Ausgaben in einer Tabelle zusammen.

Der notwendige Java-Quellcode für die Durchführung der eigentlichen numerischen Berechnung gestaltet sich relativ simpel und ist in Listing 1 dargestellt.

Listing 1: Java-Implementierung der numerischen Differentiation in Fehlerordung h^2 und h^4

```
1  public double h2(Function f, double x, double h) {
2      return (f.f(x + h) - f.f(x-h))/(2*h);
3  }
4
5  public double h4(Function f, double x, double h) {
6      return (f.f(x - 2*h) - 8*f.f(x-h) + 8*f.f(x + h) - f.f(x + 2*h))/(12*h);
7  }
```

Tabelle 1: Numerische Ergebnisse in h^2 und h^4

h	h^2	h^4
0.1	0.584574093556	0.586253751862
0.01	0.586222802638	0.586239421889
0.001	0.586239254282	0.586239420457
1.0E-4	0.586239418795	0.586239420457
1.0E-5	0.586239420441	0.586239420466
1.0E-6	0.586239420564	0.586239420582
1.0E-7	0.586239420342	0.586239420342
1.0E-8	0.58623941257	0.586239405168
1.0E-9	0.586239390366	0.586239316351
1.0E-10	0.586239945477	0.586240500589
1.0E-11	0.586242165923	0.586244016295
1.0E-12	0.586197757002	0.586142245851
1.0E-13	0.586197757002	0.58564264549

Um die Stabilität des Verfahrens beurteilen zu können, eignet sich diese Darstellung jedoch nur bedingt. Zu diesem Zweck ist es möglich, die Ergebnisse grafisch darzustellen.

In den Abbildungen 5 und 6 lässt sich deutlich erkennen, dass das Verfahren nur für bestimmte Intervalle stabile Ergebnisse liefert[1]. So sind besonders große und besonders kleine Werte von h problematisch. Die scharfen Knicke in den Kurvenverläufen machen dies deutlich. Es ist daher anzunehmen, dass Ergebnisse in diesen Bereichen hohe Fehler aufweisen.

Durch Extrapolation können die Ergebnisse nun noch verfeinert werden und eine höhere Genauigkeit erreicht werden. Dies war laut Aufgabenstellung auch gefordert. Dazu werden je zwei aufeinanderfolgende Ergebnisse verwendet. Das beiliegende Programm gibt die Resultate dann alle zusammen in einer Art Tabelle aus.

Abbildung 2 und 3 stellen die entsprechenden Auszüge aus den Ausgaben des Programms dar[2].

[1] Bei den dargestellten Grafiken skaliert die x-Achse jeweils logarithmisch, da mit einer linearen Skalierung der Verlauf nicht mehr dargestellt werden konnte.

[2] Die Daten zu den Screenshots können durch Programmausführung gewonnen werden, sind aber auch unter https://gist.github.com/lyio/f42ba4ef24a4875e6556 (h^2) bzw. https://gist.github.com/lyio/573f9a278d3afcb1bfe7 (h^4) verfügbar.

```
h =     0.1: h = 0.584574093556, error2:  0.0016653269006149385, extrapolated:  0.59047888238
h =    0.01: h = 0.586222802638, error2:  1.6617818623276825E-5, extrapolated: 0.586239456265
h =   0.001: h = 0.586239254282, error2:  1.6617477605063158E-7, extrapolated:  0.58623942046
h =  1.0E-4: h = 0.586239418795, error2:  1.6618172393734199E-9, extrapolated: 0.586239420457
h =  1.0E-5: h = 0.586239420441, error2: 1.5356493854312703E-11, extrapolated: 0.586239420458
h =  1.0E-6: h = 0.586239420564, error2: 1.0676803885445452E-10, extrapolated: 0.586239420565
h =  1.0E-7: h = 0.586239420342, error2: 1.1527656607057679E-10, extrapolated: 0.586239420339
h =  1.0E-8: h =  0.58623941257, error2:   7.886837738446673E-9, extrapolated: 0.586239412491
h =  1.0E-9: h = 0.586239390366, error2:  3.0091298230949803E-8, extrapolated: 0.586239390141
h = 1.0E-10: h = 0.586239945477, error2:   5.250202140816285E-7, extrapolated: 0.586239951084
h = 1.0E-11: h = 0.586242165923, error2:  2.7454662633319415E-6, extrapolated: 0.586242188352
h = 1.0E-12: h = 0.586197757002, error2:   4.166345472167432E-5, extrapolated: 0.586197308427
h = 1.0E-13: h = 0.586197757002, error2:   4.166345472167432E-5, extrapolated: 0.586197757002
```

Abbildung 2: Ergebnisse für h^2

```
h =     0.1: h = 0.586253751862, error2:  1.4331405260192831E-5, extrapolated:   0.5863123831
h =    0.01: h = 0.586239421889, error2:  1.4320264973832764E-9, extrapolated: 0.586239420456
h =   0.001: h = 0.586239420457, error2: 1.5543122344752192E-15, extrapolated: 0.586239420457
h =  1.0E-4: h = 0.586239420457, error2: 1.8351986597053838E-13, extrapolated: 0.586239420457
h =  1.0E-5: h = 0.586239420466, error2:  8.698264331030714E-12, extrapolated: 0.586239420466
h =  1.0E-6: h = 0.586239420582, error2: 1.2527179293897461E-10, extrapolated: 0.586239420582
h =  1.0E-7: h = 0.586239420342, error2: 1.1527656607057679E-10, extrapolated: 0.586239420342
h =  1.0E-8: h = 0.586239405168, error2:  1.5288324606288484E-8, extrapolated: 0.586239405167
h =  1.0E-9: h = 0.586239316351, error2:  1.0410616657630101E-7, extrapolated: 0.586239316342
h = 1.0E-10: h = 0.586240500589, error2:  1.0801317263942067E-6, extrapolated: 0.586240500707
h = 1.0E-11: h = 0.586244016295, error2:   4.595837971077543E-6, extrapolated: 0.586244016646
h = 1.0E-12: h = 0.586142245851, error2:   9.717460595293215E-5, extrapolated: 0.586142235673
h = 1.0E-13: h =  0.58564264549, error2:   5.967749670343636E-4, extrapolated: 0.585642595525
```

Abbildung 3: Ergebnisse für h^4

3 Fehlerdiskussion

Grundlegend sind bei dieser Art numerischer Rechnung zwei Fehlerquellen zu betrachten. Dabei handelt es sich zum einen um Approximationsfehler, die dem numerischen Verfahren selbst entstammen und Rundungsfehler, die sich daraus ergeben, dass eine beliebige Rechenmaschine Zahlen nur mit endlicher Genauigkeit darstellen kann.

In Abschnitt 1 wurden bereits die zu erwartenden Fehlerterme vorgestellt (s. Gleichungen 5 und 7). Diese dienen der oberen Abschätzung des zu erwartenden Approximationsfehlers.

Wie bereits in Abschnitt 1 angedeutet, ist insbesondere die Wahl von h von Bedeutung für den Fehler. Bei Wahl eines zu kleinen h rücken die Punkte so nah aneinander, dass die Funktionswerte an den entsprechenden Stellen nahezu identisch sind. Dies führt bei der Subtraktion zur Auslöschung der meisten signifikanten Stellen und führt damit zu unzuverlässigen Ergebnissen. Erhöht man den Abstand zwischen den Punkten, umgeht man dieses Problem und erhält eine genauere Annäherung an den Anstieg der Sekante. Allerdings verschlechtert sich dadurch die Annäherung der Sekante an die Tangente und das Ergebnis wird dennoch ungenau. Damit gibt es einen optimalen Wert für h. Abbildung 4 veranschaulicht diese Abhängigkeit.

Im vorliegenden Fall ist es wie in Abschnitt 2 möglich den absoluten Fehler – im Rahmen der Genauigkeit der ausführenden Maschine – zu ermitteln. Dabei ist die Maschine die *Java Virtual Machine*, die intern für die entsprechenden Gleitkommaoperationen eine 64-Bit-Gleitkommazahl

verwendet. Daraus ergibt sich ein Rundungsfehler ε von:

$$\begin{aligned} \varepsilon &= 0.5 \cdot B^{-t+1} \\ \varepsilon &= 2^{-53} \approx 1{,}1 \cdot 10^{-16} \end{aligned} \tag{9}$$

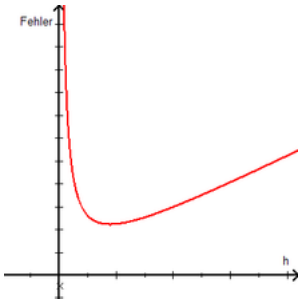

Abbildung 4: Gesamtfehler in Abhängigkeit von h; Quelle: Wikipedia

Dies ist die maximal zu erwartende Genauigkeit, d. h. der Fehler muss mindestens diese Größenordnung haben. Da Bestandteil der Aufgabenstellung ist, die Ergebnisse mit einer künstlich verkürzten Mantisse auszugeben, erhöht sich der Fehler entsprechend weiter. Laut Gleichung 9 beträgt dieser Fehler dann mindestens $\varepsilon = 10^{-11}$. Zusammen mit dem Approximationsfehler aus Abschnitt 1 ergeben sich die Fehler in der Spalte „error2“ in Abbildungen 2 und 3. Damit lässt sich der Wert mit der geringsten Abweichung vom errechneten exakten Ergebnis ermitteln. Für Fehlerordnung h^2 liegt dies bei $h = 1.0E-5$ mit $Ergebnis = 0.586239420441$. Für Fehlerordnung h^4 liegt das h mit dem kleinsten Fehler bei $h = 0.001$ und beträgt 0.586239420457.

Literatur

[1] Pepper, P., (2006): *Programmieren mit Java: Eine grundlegende Einführung für Informatiker und Ingenieure*. Berlin, Heidelberg, Springer Verlag.

[2] o. V. (September 2015): *Differentialrechnung* https://de.wikipedia.org/wiki/Differentialrechnung

[3] o. V. (September 2015): *Numerische Differentiation* https://de.wikipedia.org/wiki/Numerische_Differentiation

[4] o. V. (September 2015): *Numerical Differentiation* https://en.wikipedia.org/wiki/Numerical_differentiation

Anhang

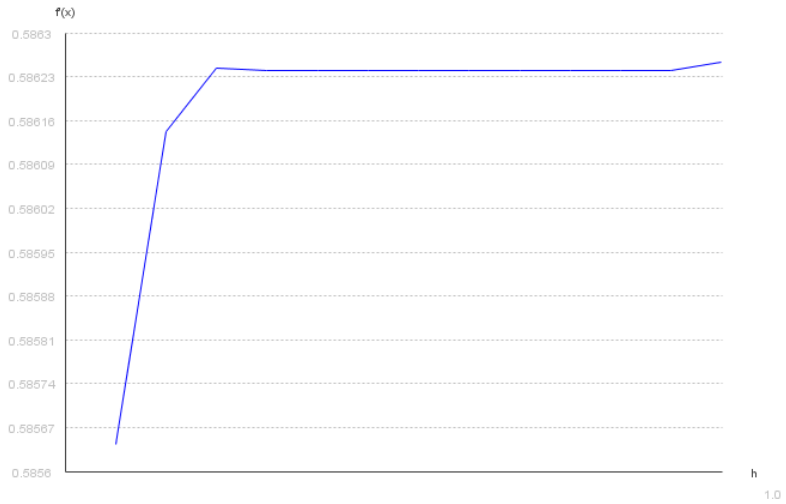

Abbildung 5: Plot von h^2

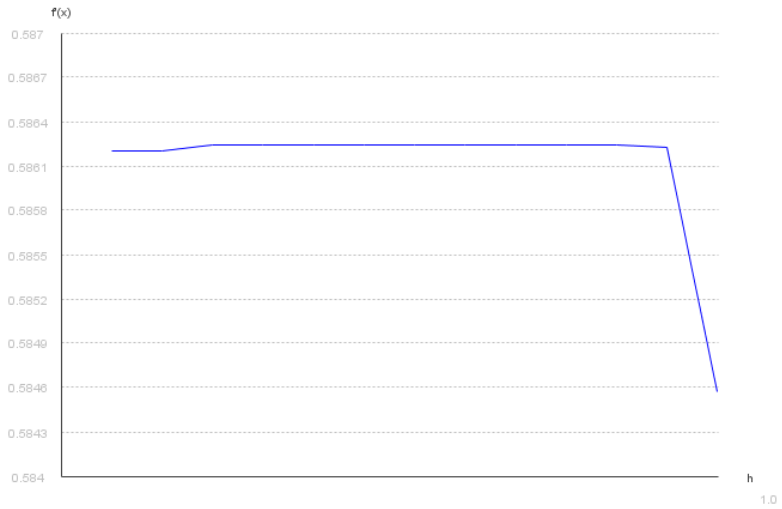

Abbildung 6: Plot von h^4

Erklärung

Ich versichere, dass ich die vorliegende Arbeit ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Quellen und Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form weder veröffentlicht, noch einer anderen Prüfungsbehörde vorgelegt.

28. September 2015 Thomas Fiedler