Institut für Analysis und Numerik
Universität Magdeburg
Thomas Richter

24.04.2020

Lösung zu: Übung Nr. 2 zur Algorithmische Mathematik I
Wintersemester 2019/2020

Aufgabe 2.1:

Wir betrachten das Distributivgesetz und seine Umsetzung auf dem Computer

$$f(x,y,z) = x(y+z) = xy + xz.$$

a) Man berechne die Konditionszahlen der Auswertung der Funktion $f(\cdot)$ in Bezug auf x, y, z. Wann ist die Auswertung schlecht konditioniert?

Es gilt:

$$\kappa_x = \partial_x f(x,y,z)\frac{x}{f(x,y,z)} = (y+z)\frac{x}{x(y+z)} = 1.$$
$$\kappa_y = \partial_y f(x,y,z)\frac{y}{f(x,y,z)} = x\frac{y}{x(y+z)} = \frac{y}{y+z} = \frac{1}{1+\frac{z}{y}}$$
$$\kappa_z = \partial_z f(x,y,z)\frac{z}{f(x,y,z)} = x\frac{z}{x(y+z)} = \frac{z}{y+z} = \frac{1}{1+\frac{y}{z}}.$$

Die Konditionierung in Bezug auf Eingabefehler in x ist immer gut. Das ist eigentlich klar, denn x geht nur als Produkt ein. Die Multiplikation ist stets gut konditioniert.
In Bezug auf y und z ist die Konditionierung schlecht, falls $y \approx -z$. Hier kommt es zur Auslöschung wesentlicher Stellen.

b) Welche der beiden Varianten zur Auswertung ist stabiler

$$(i) \quad a_1 \coloneqq y + z,\ a_2 \coloneqq x a_1, \quad (ii) \quad a_1 \coloneqq xy,\ a_2 \coloneqq xz,\ a_3 \coloneqq a_1 + a_2.$$

Man nutze zur Begründung der Entscheidung die Vorwärtsanalyse.

Variante (i): Zunächst $(y+z) \approx (y+z)(1+\epsilon_1)$. Danach Multiplikation mit x also ist das gestörte Ergebnis:

$$(y+z)(1+\epsilon_1)\cdot x(1+\epsilon_2) = (y+z)x\big(1+\epsilon_1+\epsilon_2+\epsilon_1\epsilon_2\big) \quad = (y+z)x + (y+z)x(\epsilon_1+\epsilon_2) + \mathcal{O}(eps^2)$$

Der relative Fehler ergibt sich zu

$$\frac{\tilde{f}-f}{f} = \frac{(y+z)x(\epsilon_1+\epsilon_2)+\mathcal{O}(eps^2)}{(y+z)x} = \epsilon_1+\epsilon_2+\mathcal{O}(eps^2).$$

D.h. der Fehler sollte maximal $2eps$ betragen. Die Methode ist stabil.

Variante (ii): zunächst $xy(1+\epsilon_1)$ dann $xz(1+\epsilon_2)$ danach die Summe, also

$$\tilde{f} = \big(xy(1+\epsilon_1)+xz(1+\epsilon_2)\big)(1+\epsilon_3) = \big(xy+xz+xy\epsilon_1+xz\epsilon_2\big)+\big(xy+xz+xy\epsilon_1+xz\epsilon_2\big)\epsilon_3$$

Der rel. Fehler ist

$$\frac{\tilde{f}-f}{f} = \frac{y(\epsilon_1+\epsilon_3)+z(\epsilon_2+\epsilon_3)+\mathcal{O}(eps^2)}{y+z}$$

Abschätzen (Vorzeichen der epsilon ist nicht bekannt) durch

$$\Big|\frac{\tilde{f}-f}{f}\Big| \leqslant \frac{(|y|+|z|)2eps}{|y+z|} + +\mathcal{O}(eps^2),$$

d.h. der Fehler kann bei $y+z \approx 0$ beliebig verstärkt werden. Variante (i) ist stabiler. Generell sollten erst die schlecht konditionierten Operationen durchgeführt werden.

Hinweis Es scheint, dass das Distributivgesetz keine Fehlerverstärkugn hat, wenn man Variante (i) wählt. Das ist ein Widerspruch zur schlechten Konditionierung. Dies liegt daran, dass die Analyse vereinfacht ist. Die KOnditionierung misst den Einfluss von Rundungsfehlern in den Eingaben. D.h. eigentlich müssten wir dies hier analysieren:

$$\tilde{f}(\tilde{x},\tilde{y},\tilde{z}) = \big(y(1+\epsilon_y)+z(1+\epsilon_z)\big)(1+\epsilon_1)z(1+\epsilon_z)(1+\epsilon_2)$$

Programmieraufgabe 2.2:

Wir untersuchen die Funktion

$$f : \mathbb{R} \to \mathbb{R}, \quad f(x) = \sin\left(\frac{1}{x^2 + \frac{1}{20}}\right).$$

a) (keine Programmierung) Man gebe eine Abschätzung für ein möglichst kleines Intervall $[a, b]$ an, in dem sämtliche Nullstellen der Funktion liegen müssen.

Es ist $(x^2 + 0.05)^{-1} > 0$. $\sin(x)$ hat für $x \in (0, \pi)$ keinen Vorzeichenwechsel. Für

$$\frac{1}{x^2 + 0.05} < \pi \quad \Leftrightarrow \quad x^2 > \frac{1}{\pi} - 0.05$$

hat die Funktion daher sicher keinen Vorzeichenwechsel mehr. Dies ist z.B. für $x > 0.6$, bzw. $x < -0.6$ erfüllt. Alle Nullstellen müssen daher im Intervall $(-0.6, 0.6)$ liegen. Dies sagt nicht, dass es auch Nullstellen gibt, denn außerhalb des Intervalls ist die Funktion immer positiv, sowohl für $x > 0.3$ als auch für $x < -0.3$. Wir haben nur gezeigt, dass es keinen weiteren Vorzeichenwechsel mehr geben kann.

---

b) Aufbauend auf der Intervallschachtelung (Algorithmus 6.2, Skript Version 24.04.2020) approximiere man sämtliche Nullstellen z_i der Funktion $f(x)$. Dabei gebe man die gefundenen Nullstellen mit einem Fehler von maximal 10^{-4} an. Dies kann sichergestellt werden, wenn die verbleibende Intervallgröße $|b - a|$ den Wert 10^{-4} erreicht. Zur Wahl geeigneter Startwerte verwende man entweder eine graphische Darstellung oder man versuche, eine automatische Methode zu entwickeln.

```python
import numpy as np

def fkt(x):
    return np.sin(1.0/(x*x+1.0/20.0))

def intervall(a,b,tol):      # Intervallschachtelung
    assert a<b,"Intervallgrenzen falsch"
    assert tol>0,"Toleranz falsch"
    fa = fkt(a)
    fb = fkt(b)
    assert fa*fb<0,"Kein Vorzeichenwechsel"
```

```
    while b-a>tol:            # Fehlertoleranz erreicht?
        m = 0.5*(a+b)         # Berechnung Mittelpunkt
        fm = fkt(m)           # f im Mittelpunkt
        if fm==0:             # NS gefunden
            return m
        if fm*fa<0:           # VZW links
            fb = fm
            b = m
        else:                 # VZW rechts
            fa = fm
            a = m
    return 0.5*(a+b)          # Mittelwert als beste Approximation

def alle(l,r,h):
    assert l<r,"Intervallgrenzen falsch"
    assert h>0,"Schrittweite falsch"

    a=l
    while a<r:    # Suche geeignete Startintervalle (a,b)
        b=a+h     # Suche Startintervall mit schrittweite h
        while (fkt(a)*fkt(b)>=0) and (b<r+h):
            b=b+h
        if (fkt(a)*fkt(b)<0):
            print(intervall(a,b,1.e-4))
        a=b       # naechstes Intervall

alle(-0.6,0.6,0.05)
```

mit den 12 Nullstellen:

```
-0.5180175781249998
-0.3304199218749999
-0.23686523437499996
-0.17202148437499998
-0.11684570312500002
-0.05522460937500001
0.05521240234374999
0.11684570312499999
0.17202148437499998
0.23686523437499996
0.3304199218749999
0.5180175781249998
```

Abgabe der Übungen (5er-7er Gruppen) sowie der Kurzfragen (2er Gruppen) per Mail an `algomath@ovgu.de`.