Programmieraufgaben

(1) (a) Bestimmen Sie experimentell mit einem Testprogramm die relative Maschinengenauigkeit `eps` in python, indem Sie folgenden Pseudo-Code umsetzen:
`eps` $\leftarrow 1.0$
solange $1.0 + \texttt{eps} > 1.0$ **tue**
| `eps` $\leftarrow$ `eps/2`
Ende

(b) Zeichnen Sie den Graphen der Funktion

$$f\colon \left(0, 10^{-15}\right] \to \mathbb{R}\colon x \mapsto \frac{(1+x)-1}{x}$$

sowie der Funktion

$$f\colon \left(0, 10^{-15}\right] \to \mathbb{R}\colon x \mapsto \frac{\mathrm{e}^x - 1}{x}$$

mit python und erklären Sie jeweils den Verlauf der Graphen.

(2) (a) Bestimmen Sie jeweils die absolute Kondition der folgenden Funktionen und diskutieren Sie diese experimentell, indem Sie $|f(\tilde{x}) - f(x)|$ und $|\tilde{x} - x|$ vergleichen. Betrachten Sie hierbei verschiedene Werte für x und $\tilde{x}$.

(i) $f\colon \mathbb{R} \to \mathbb{R} : x \mapsto \mathrm{e}^x$

(ii) $f\colon (0, \infty) \to \mathbb{R} : x \mapsto \ln(x)$

(b) Die Funktion $f\colon [0, \infty) \to \mathbb{R} : x \mapsto x\sqrt{x+1} - x\sqrt{x}$ kann äquivalent zu $f(x) = \frac{x}{\sqrt{x+1}+\sqrt{x}}$ umgeformt werden. Betrachten Sie folgende Algorithmen zur Auswertung von f.

Funktion f_1
| **Übergabe:** x
| $s \leftarrow x + 1$
| $t \leftarrow \sqrt{s}$
| $u \leftarrow x \cdot t$
| $v \leftarrow \sqrt{x}$
| $w \leftarrow x \cdot v$
| **Rückgabe:** $u - w$

Funktion f_2
| **Übergabe:** x
| $s \leftarrow x + 1$
| $t \leftarrow \sqrt{s}$
| $u \leftarrow \sqrt{x}$
| $v \leftarrow t + u$
| **Rückgabe:** $\frac{x}{v}$

Implementieren Sie beide Algorithmen in python. Berechnen Sie damit den Funktionswert an der Stelle $x = 10^8$ sowie die absoluten Fehler $|f_1(\tilde{x}) - f_1(x)|$ sowie $|f_2(\tilde{x}) - f_2(x)|$ bei gestörten Werten $\tilde{x}$ von $x = 10^8$. Berechnen Sie technologieunterstützt die absolute Konditionszahl der Funktion f in $x = 10^8$. Welcher der beiden Algorithmen ist warum vorzuziehen?

(3) (a) Schreiben Sie eine Funktion *efun* in python, welche eine Näherung an die Exponentialfunktion

$$\mathrm{e}^x = \sum_{n=0}^{\infty} \frac{x^n}{n!}$$

berechnet. Als Übergabewert sollen eine reelle Zahl x und die Toleranz `tol` verwendet werden. Die einzelnen Summanden $s_n = \frac{x^n}{n!}$ lassen sich rekursiv durch

$$s_0 = 1$$
$$s_n = \frac{x}{n} \cdot s_{n-1} \quad n \geq 1$$

berechnen. Sobald $|s_n| < \texttt{tol}$ ist, soll die Berechnung beendet werden und die Näherungslösung zurückgegeben werden.

(b) Schreiben Sie eine neue Funktion *efun2*, welche bei Übergabe einer reellen Zahl x und Toleranz `tol` eine Näherung von e^x nach folgender Vorschrift berechnet:

- Falls $x < 0$ bestimmen Sie erst e^{-x} näherungsweise mit Teilaufgabe (a) unter Verwendung der vorgegebenen Toleranz und berechnen Sie anschließend den Kehrwert $\frac{1}{\mathrm{e}^{-x}}$. Um beispielsweise e^{-10} zu berechnen, berechnen Sie erst e^{10} mit obigem Algorithmus und bestimmen Sie dann den Kehrwert.
- Für $x \geq 0$ berechnen Sie e^x mit dem Algorithmus aus Aufgabe (a) unter Verwendung der vorgegeben Toleranz.

Berechnen Sie den relativen Fehler Ihrer Näherungen für ausgewählte Werte von x sowie `tol` (beispielsweise $\texttt{tol} = 0.1, 0.01, 0.001\ldots$) für beide Funktionen *efun* und *efun2*.

Theorieaufgaben

(4) Gegeben sei die Funktion

$$f\colon (-\infty, 1) \to \mathbb{R}\colon x \mapsto \begin{cases} \frac{\log(1-x)+x\,\mathrm{e}^x}{x^2}, & x \neq 0 \\ \frac{1}{2}, & x = 0\,. \end{cases}$$

(a) Zeigen Sie, dass f in 0 differenzierbar ist.

(b) Bestimmen Sie die absolute Kondition des Problems für $x = 0$.

(c) Welches Ergebnis erhalten Sie, wenn Sie den obigen Ausdruck für $x = 10^{-10}$ in python berechnen? Wie erklären Sie sich das Ergebnis im Lichte von (b)?

(d) Formulieren Sie einen stabilen Algorithmus für die Berechnung von $f(x)$ für x nahe 0.

(5) Die Nullstellen des Polynoms $x^2 - 2px - q$ können mit den Formeln

$$x_1 = p + \sqrt{p^2 + q}, \qquad x_2 = p - \sqrt{p^2 + q}$$

berechnet werden. Im Folgenden seien $p, q > 0$ und $p \gg q$ vorausgesetzt. Zeigen Sie:

(a) Das Problem ist gut konditioniert.

(b) Obiger Algorithmus ist numerisch instabil (im Sinne der Vorwärtsanalyse).

(c) Finden Sie mit Hilfe des *Satzes von Vietá* ($x_1 x_2 = -q$) einen numerisch stabilen Algorithmus.