Praktische Mathematik I

Wintersemester 2002/03
Prof. Dr. Angela Kunoth, Marcel Arndt

Rheinische Friedrich-Wilhelms-Universität

Aufgabenblatt 1

Ausgabe: 24.10.2002, Abgabe der Lösungen: 31.10.2002, 10:30
Abgabe der Programmieraufgaben: 11.-15.11.2002, genauer Termin nach Vereinbarung.

Aufgabe 1:
Sei $x > 0$. Untersuche, für welche Exponenten $\alpha, \beta \in \mathbb{R}$ gilt:

$$\begin{aligned} &x^\alpha = O(x^\beta) \text{ für } x \to 0 \qquad && x^\alpha = o(x^\beta) \text{ für } x \to 0 \\ &x^\alpha = O(x^\beta) \text{ für } x \to 1 && x^\alpha = o(x^\beta) \text{ für } x \to 1 \\ &x^\alpha = O(x^\beta) \text{ für } x \to \infty && x^\alpha = o(x^\beta) \text{ für } x \to \infty \end{aligned}$$

(6 Punkte)

Aufgabe 2:
Seien $n \in \mathbb{N}$ und $A, B \in \mathbb{R}^{n \times n}$ Matrizen.
a) Bestimme, wieviele elementare Additionen und wieviele Multiplikationen reeller Zahlen zur Berechnung des Produkts $C := AB$ mit der gewöhnlichen Summenformel notwendig sind. Eine elementare Rechenoperation koste die Zeit 1. Drücke die benötigte Zeit für die Matrixmultiplikation mit Hilfe der Landausymbole aus.
b) Sei nun n eine Zweierpotenz, also $n = 2^m$ mit $m \in \mathbb{N}$. Strassen schlug folgendes Verfahren[1] vor (siehe auch Vorlesung LA I): Zerlege A und B in jeweils vier $\frac{n}{2} \times \frac{n}{2}$-Matrizen

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \qquad B = \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix}.$$

Berechne die sieben Hilfsprodukte

$$\begin{aligned} P_1 &= (A_{11} + A_{22})(B_{11} + B_{22}) \qquad & P_5 &= (A_{11} + A_{12})B_{22} \\ P_2 &= (A_{21} + A_{22})B_{11} & P_6 &= (A_{21} - A_{11})(B_{11} + B_{12}) \\ P_3 &= A_{11}(B_{12} - B_{22}) & P_7 &= (A_{12} - A_{22})(B_{21} + B_{22}), \\ P_4 &= A_{22}(B_{21} - B_{11}) \end{aligned}$$

dann gilt:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{pmatrix} = \begin{pmatrix} P_1 + P_4 - P_5 + P_7 & P_3 + P_5 \\ P_2 + P_4 & P_1 + P_3 - P_2 + P_6 \end{pmatrix}$$

Damit reduziert sich die Zahl der Multiplikationen kleinerer Matrizen von acht auf sieben gegenüber dem naiven Verfahren, während die Zahl der Additionen gestiegen ist[2]. Dieses Verfahren wird nun rekursiv zur Berechnung der sieben kleineren Produkte angewandt.
Zeige, daß die benötige Zeit $T(n)$ für die Multiplikation zweier $n \times n$-Matrizen der Rekurrenzrelation

$$T(n) = 7T\left(\frac{n}{2}\right) + \frac{9}{2}n^2, \qquad T(1) = 1$$

genügt. Folgere daraus, daß die Matrixmultiplikation nach Strassen die Komplexität $O(n^{\log_2 7}) \approx O(n^{2.807})$ besitzt.
c) Zeige, daß $O(n^2)$ eine untere Schranke für die Komplexität der Matrixmultiplikation ist. (15 Punkte)

Bemerkung: Der derzeit beste bekannte Algorithmus zur Matrixmultiplikation stammt von D. Coppersmith und S. Winograd[3] aus dem Jahr 1990 und besitzt die Komplexität $O(n^{2.376})$.

[1] V. Strassen, *Gaussian elimination is not optimal,* Numerische Mathematik, 13:354–356 (1969).
[2] Es gibt eine Variante von Winograd, die nur 15 statt 18 Additionen benötigt.
[3] D. Coppersmith, S. Winograd, *Matrix Multiplication via Arithmetic Progressions,* Journal of Symbolic Computation, 9:251–280 (1990).

Aufgabe 3:
Gegeben sei die Matrix

$$A = \begin{pmatrix} 0 & a & b \\ a & 0 & c \\ b & c & 0 \end{pmatrix}$$

mit Konstanten $a, b, c \in \mathbb{R}$.
a) Berechne die relativen Konditionszahlen der Berechnung der Koeffizienten $p(a, b, c) = -a^2 - b^2 - c^2$ und $q(a, b, c) = -2abc$ des charakteristischen Polynoms $\chi_A(x) = x^3 + p(a, b, c)x + q(a, b, c)$ von A. Ist das ein gut oder ein schlecht konditioniertes Problem?
b) Zeige, daß die Berechnung der Eigenwerte $\lambda_i(p, q)$ von A mittels der Nullstellen von χ_A schlecht konditioniert ist.
Hinweis: Berechne $\frac{\partial \lambda_i}{\partial p}$ bzw. $\frac{\partial \lambda_i}{\partial q}$ implizit. (10 Punkte)

Aufgabe 4:
a) Bestimme die Binärdarstellung der Dezimalzahlen 108 und -30 sowie ihre Bitdarstellungen als 8-bit signed integer.
b) Betrachte die (binäre) Gleitkommadarstellung bestehend aus normalisierten Gleitkommazahlen mit Exponent $e \in \{-3, \ldots, 3\}$ und denormalisierten Gleitkommazahlen mit Exponent $e = -3$. Die Mantissenlänge betrage 4 bit. Bestimme die Darstellungen der Zahlen

$$\frac{5}{2}, \qquad \frac{1}{3} \qquad \text{und} \qquad \frac{-1}{10}.$$

Berechne (in der Gleitkommadarstellung)

$$\frac{5}{2} - \frac{1}{10} \qquad \text{und} \qquad \frac{5}{2} \cdot \frac{1}{3}.$$

(10 Punkte)

Aufgabe 5:
Gegeben seien die Integrale

$$I_k = \frac{1}{e} \int_0^1 e^x x^k \mathrm{d}x$$

für $k \in \mathbb{N}$.
a) Zeige die Abschätzungen

$$\frac{1}{e(k+1)} < I_k < \frac{1}{k+1} \qquad \text{und} \qquad 0 < I_{k+1} < I_k.$$

b) Zeige die Rekursionsformel

$$I_{k+1} + (k+1)I_k = 1.$$

Damit lassen sich also die Integrale $I_0, I_1, \ldots, I_K$ rekursiv berechnen:

Vorwärtsrekursion:	$I_{k+1} := 1 - (k+1)I_k$	$k = 0, 1, 2, \ldots, K-1$
Rückwärtsrekursion:	$I_{k-1} := \dfrac{1 - I_k}{k}$	$k = K, K-1, \ldots, 1$

falls Startwerte I_0 bzw. I_K bekannt sind.
c) Anstelle der exakten Startwerte I_0 bzw. I_K seien genäherte Startwerte $\tilde{I}_0$ bzw. $\tilde{I}_K$ gegeben. Wie lauten die absoluten Fehler $\Delta I_k = I_k - \tilde{I}_k$ bei Vorwärts- und Rückwärtsrekursion? (10 Punkte)

Programmieraufgabe 1:
Schreibe ein C-, C++- oder Javaprogramm, das die Integrale $I_0, I_1, \ldots, I_K$ aus Aufgabe 5 für $K = 30$ mittels Vorwärts- und Rückwärtsrekursion berechnet. Verwende als Startwert für die Vorwärtsrekursion den exakten Wert

$$I_0 = \frac{e-1}{e}$$

und als Startwert für die Rückwärtsiteration die Werte

$$I_K = \frac{1}{2}\left(\frac{1}{e(K+1)} + \frac{1}{K+1}\right) \qquad \text{und} \qquad I_K = 10^6,$$

also das arithmetische Mittel der Abschätzungen oben und einen vollkommen schlechten Wert. Stelle die Ergebnisse tabellarisch dar und beurteile sie, auch im Hinblick auf Aufgabe 5 c). (10 Punkte)

Beispielprogramm in C:
Das folgende kleine Beispielprogramm berechnet die Sinusfunktion an einigen Stellen und gibt die Werte tabellarisch aus.

```
#include <stdlib.h>
#include <stdio.h>
#include <math.h>

int main()
{
    /* Variablen deklarieren */
    int k;
    double x;
    double y[9];

    /* Schleife: k = 0, 1, 2, ..., 8 */
    for (k=0; k<9; k=k+1)
        {
        /* Werte berechnen */
        x = k * M_PI / 8;
        y[k] = sin(x);

        /* Ausgabe */
        printf("k=%d x=%e y[k]=%e\n", k, x, y[k]);
        }
    return 0;
}
```

Kurzanleitung zur Verwendung:
Starte einen Editor:

```
nedit beispiel.c &
```

Gib das Programm ein und speichere es ab. Kompiliere es mit

```
gcc beispiel.c -o beispiel -lm
```

Führe es aus:

```
beispiel
```