Dirk Praetorius
Christoph Überhuber

Wintersemester 2006/07
12. Jänner 2007

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 10

Die Aufgaben mit Stern (*) sind bis zur nächsten Übung vorzubereiten und werden dort abgeprüft. Die übrigen Aufgaben dienen nur Ihrer Übung und mir als zusätzliche Grundlage für den Prüfungsstoff in den schriftlichen Tests. Kopieren Sie bitte den Source-Code in ein Unterverzeichnis `serie10` *Ihres Home-Verzeichnisses. Überprüfen Sie bitte vor der Übung, ob Ihre Source-Codes mit* `matlab` *auf der* `cad.zserv.tuwien.ac.at` *interpretiert werden können.*

Aufgabe 89*. Das Δ^2-Verfahren von Aitkin ist ein Verfahren zur Konvergenzbeschleunigung von Folgen. Für eine injektive Folge $(x_n)_{n\in\mathbb{N}}$ mit $\lim_n x_n = x$ definiert man

$$y_n := x_n - \frac{(x_{n+1} - x_n)^2}{x_{n+2} - 2x_{n+1} + x_n}$$

Unter gewissen zusätzlichen Voraussetzungen an die Folge $(x_n)_{n\in\mathbb{N}}$ gilt dann

$$\lim_{n\to\infty} \frac{y_n - x}{x_n - x} = 0,$$

d.h. die Folge $(y_n)_{n\in\mathbb{N}}$ konvergiert schneller gegen x als $(x_n)_{n\in\mathbb{N}}$. Man schreibe eine Funktion `aitkin`, die für einen Vektor $x \in \mathbb{R}^n$ mit Länge $n \geq 3$ den Vektor $y \in \mathbb{R}^{n-2}$ berechnet. Den Source-Code speichere man ins Verzeichnis `serie10`.

Aufgabe 90*. Man kombiniere das Aitkin-Verfahren aus Aufgabe 89 mit dem einseitigen Differenzenquotienten $\Phi(h)$ aus Aufgabe 53: Mit $h_n := 2^{-n}h_0$ betrachten wir die Folge der $x_n := \Phi(h_n)$ und erhalten daraus die Folge (y_n). Man schreibe eine Funktion `diffaitkin`, die neben dem Funktionspointer einer Funktion f, den Auswertungspunkt x, die Schrittweite $h_0 > 0$ sowie die Toleranz $\varepsilon > 0$ übernimmt und $y_{n+1} \approx f'(x)$ zurückliefert, sobald

$$|y_n - y_{n+1}| \leq \varepsilon \max\{|y_n|, |y_{n+1}|\}$$

gilt. In jedem Schritt gebe man h, $|y_{n+1} - y_n|$ sowie y_{n+1} aus. Den Source-Code speichere man ins Verzeichnis `serie10`. Als Beispiel betrachte man die Berechnung von $e = \exp(1)$ und $\varepsilon = 10^{-12}$. Man vergleiche die Anzahl der Iterationen mit und ohne (d.h. $y_n = x_n$) Aitkin-Verfahren.

Aufgabe 91*. Die Spaltensummennorm einer Matrix $A \in \mathbb{R}^{m\times n}$ ist durch

$$\|A\| = \max_{k=1,\dots,n} \sum_{j=1}^{m} |A_{jk}|$$

definiert. Man schreibe eine Funktion `spaltennorm(A)`, die die Spaltensummennorm einer Matrix A berechnet. Den Source-Code speichere man ins Verzeichnis `serie10`.

Aufgabe 92. Man schreibe eine Prozedur `pascal(k)`, die die ersten k-Stufen des Pascal'schen Dreiecks ausgibt: Jede Zeile dieses Schemas beginnt und endet mit 1. Die restlichen Zahlen werden als Summe nebeneinanderstehender Zahlen der vorhergegangenen Zeile gebildet. Für $k = 5$ gilt beispielsweise

```
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

Realisieren Sie das Pascalsche Dreieck möglichst rechenökonomisch, insbesondere ohne die Darstellung der Einträge über den Binomialkoeffizienten.

Aufgabe 93. Man schreibe eine rekursive Funktion `detlaplace`, die die Determinante $\det(A)$ einer Matrix $A \in \mathbb{R}^{n\times n}$ mit Hilfe des Laplaceschen Entwicklungssatzes berechnet. Sie können Ihre Funktion mit der Matlab-Funktion `det` verifizieren.

Aufgabe 94. Alternativ kann man die Determinante einer Matrix $A \in \mathbb{R}^{n\times n}$ über die normalisierte LU-Zerlegung aus Aufgabe 37 berechnen. Es gilt nämlich $\det(A) = \det(L)\det(U) = \det(U) = \prod_{j=1}^{n} u_{jj}$. Schreiben Sie eine Funktion `detLU`, die die Determinante einer Matrix A mittels der normalisierten LU-Zerlegung berechnet. Sie können Ihre Funktion mit der Matlab-Funktion `det` verifizieren. Die Berechnung der LU-Zerlegung können sie in Matlab mittels `lu` überprüfen.

Aufgabe 95. Gegeben Sei ein sortierter Vektor x der Länge n. Man schreibe eine Funktion `findbisection(x,y)`, die einen Index i zurückgibt, für den $x_i = y$ gilt. Falls y nicht in x vorkommt, werde 0 zurückgegeben. Naive Realisierung führt auf Aufwand n im worst case, d.h. man durchsucht den Vektor von vorne nach hinten und das gesuchte y steht entweder an der letzten Stelle in x bzw. kommt überhaupt nicht in x vor. Wie kann man den Bisektionsalgorithmus aus Aufgabe 67 geeignet modifizieren, um einen Algorithmus mit worst case Aufwand $\log(n)$ zu erhalten?

Aufgabe 96. Für eine Matrix $A \in \mathbb{R}^{m\times n}$ ist die transponierte Matrix $A^T \in \mathbb{R}^{n\times m}$ durch $(A^T)_{jk} = A_{kj}$ definiert. Schreiben Sie eine Funktion `transpose`, die die transponierte Matrix mittels Schleifen berechnet und zurückgibt. In Matlab wird die Transponierte einer Matrix `A` durch `A'` gegeben.

Aufgabe 97. Es sei $A \in \mathbb{R}^{n\times n}$ eine symmetrische und positv definite Matrix, d.h. $Ax \cdot x > 0$ für $x \neq 0$. Dann existiert eine eindeutige untere Dreiecksmatrix $L \in \mathbb{R}^{n\times n}$ mit Diagonalelementen $\ell_{jj} > 0$ für alle $j = 1 \dots, n$ und $A = LL^T$, d.h. A besitzt eine spezielle LU-Zerlegung mit $U = L^T$. Man bezeichnet diese Faktorisierung als *Cholesky-Faktorisierung*. Leiten Sie aus der Formel für das Matrix-Matrix-Produkt $A = LL^T$ eine Formel für die Einträge von L her, und schreiben Sie eine Funktion `cholesky`, die L zurückliefert. Sie können Ihre Funktion in Matlab mittels `chol` verifizieren.

Aufgabe 98. Für eine konvergente Folge $(x_n)_{n\in\mathbb{N}}$ mit Grenzwert x spricht man von Konvergenzordnung $p \geq 1$, falls es eine Konstante $c > 0$ gibt mit $|x_{n+1} - x| \leq c\,|x_n - x|^p$ für alle $n \in \mathbb{N}$. Mit dem Ansatz $|x_{n+1} - x| = c|x_n - x|^p$ kann man für fixiertes n die Unbekannten p und c bestimmen. Mit der Dreiecksungleichung ist die experimentelle Konvergenzordnung dann durch

$$p_n = \frac{\log(|x_{n+3} - x_{n+2}|/|x_{n+2} - x_{n+1}|)}{\log(|x_{n+2} - x_{n+1}|/|x_{n+1} - x_n|)} \quad \text{und} \quad c_n = \frac{|x_{n+3} - x_{n+2}|}{|x_{n+2} - x_{n+1}|^{p_n}}$$

definiert. Schreiben Sie eine Funktion `convorder`, die für einen Vektor $x \in \mathbb{R}^n$ die Vektoren $p, c \in \mathbb{R}^{n-3}$ berechnet und zurückgibt.

Aufgabe 99. Um numerisch Nullstellen von Funktionen zu finden, formuliert man Probleme häufig in Fixpunktform: Die positive Nullstelle x^* der Funktion $f(x) = x^2 + \exp(x) - 2$ ist beispielsweise Fixpunkt der Funktion $\phi(x) = \log(2 - x^2)$, d.h. $\phi(x^*) = x^*$. Man definiert nun die Iteriertenfolge $x_n := \phi(x_{n-1})$ für einen Startwert x_0. Falls die Folge $(x_n)_{n\in\mathbb{N}}$ konvergiert, konvergiert sie aus Stetigkeitsgründen gegen einen Fixpunkt von ϕ und damit gegen x^*. Realisieren Sie diese Fixpunktiteration mit/ohne Aitkinsche Konvergenzbeschleunigung. Die Iteration soll abgebrochen werden, sobald entweder $|f(y_n)| \leq \varepsilon_1$ oder $|y_n - y_{n-1}| \leq \varepsilon_2 \max\{|y_n|, |y_{n-1}|\}$ gilt, dabei bezeichnen y_n die Folgenglieder der Aitkin-Folge (bzw. $y_n = x_n$). Wie verhält sich die numerische Konvergenzordnung?

Aufgabe 100. Auch das Newton-Verfahren aus Aufgabe 50 ist eine Fixpunktiteration gemäß Aufgabe 99 mit $\phi(x) = x - f(x)/f'(x)$. Welche experimentelle Konvergenzordnung beobachtet man? Üblicherweise konvergiert das Newton-Verfahren nur lokal, d.h. wenn der Startwert x_0 hinreichend dicht an der Nullstelle x^* liegt. Verifizieren Sie dies für $f(x) = \arctan(x)$! Visualisieren Sie sich die Iteriertenfolge geeignet, um die Konvergenz bzw. Divergenz in Abhängigkeit von x_0 zu sehen.