Dirk Praetorius
Christoph Überhuber

Sommersemester 07
30. Mai 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. In den folgenden Aufgaben sollen noch einmal* ***Schleifen*** *geübt werden.*

Aufgabe 91*. Zu gegebenen reellen Stützstellen $x_1 < \cdots < x_n$ und Funktionswerten $y_j \in \mathbb{R}$ garantiert die Lineare Algebra ein eindeutiges Polynom $p(t) = \sum_{j=1}^{n} a_j t^{j-1}$ vom Grad $n-1$ mit $p(x_j) = y_j$ für alle $j = 1, \ldots, n$. Nun sei $t \in \mathbb{R}$ fixiert und $p(t)$ gesucht. Man kann $p(t)$ mit dem *Neville-Verfahren* berechnen, ohne zunächst den Koeffizientenvektor $a \in \mathbb{R}^n$ berechnen zu müssen: Dazu definiere man für $j, m \in \mathbb{N}$ mit $m \geq 2$ und $j + m \leq n + 1$ die Werte

$$\begin{aligned} p_{j,1} &:= y_j, \\ p_{j,m} &:= \frac{(t - x_j)p_{j+1,m-1} - (t - x_{j+m-1})p_{j,m-1}}{x_{j+m-1} - x_j}. \end{aligned}$$

Es gilt dann $p(t) = p_{1,n}$. Man schreibe eine Funktion `neville`, die den Auswertungspunkt $t \in \mathbb{R}$ sowie die Vektoren $x, y \in \mathbb{R}^n$ übernimmt und $p(t)$ mittels Neville-Verfahren berechnet. Dazu berücksichtige man das folgende schematische Vorgehen

$$\begin{array}{ccccccccccccc} y_1 & = & p_{1,1} & \longrightarrow & p_{1,2} & \longrightarrow & p_{1,3} & \longrightarrow & \cdots & \longrightarrow & p_{1,n} & = & p(t) \\ & & & \nearrow & & \nearrow & & & & \nearrow & & & \\ y_2 & = & p_{2,1} & \longrightarrow & p_{2,2} & & & & \nearrow & & & & \\ & & & \nearrow & & & & & & & & & \\ y_3 & = & p_{3,1} & \longrightarrow & \vdots & & & & & & & & \\ \vdots & & \vdots & & \vdots & \nearrow & & & & & & & \\ y_{n-1} & = & p_{n-1,1} & \longrightarrow & p_{n-1,2} & & & & & & & & \\ & & & \nearrow & & & & & & & & & \\ y_n & = & p_{n,1} & & & & & & & & & & \end{array} \tag{2}$$

Der mathematische Beweis für diesen Algorithmus folgt in der Vorlesung zur Numerischen Mathematik. Zunächst schreibe man die Funktion, wobei man die Matrix $(p_{j,m})_{j,m=1}^{n}$ vollständig aufbaue.

Aufgabe 92*. Man kann das Neville-Verfahren aus Aufgabe 91 so programmieren, dass zur Speicherung der Werte *keine* Matrix $(p_{j,m})_{j,m=1}^{n}$ aufgebaut wird, sondern die gegebenen y_j-Werte geeignet überschrieben werden. Dadurch wird kein weiterer Speicher benötigt. Man realisiere dieses Vorgehen in einer Funktion `neville2`.

Aufgabe 93*. Eine effiziente Implementierung des einseitigen Differenzenquotienten $\Phi(h)$ aus Aufgabe 86 verwendet die vorherigen Werte $\Phi(h_0), \ldots, \Phi(h_n)$, indem man (theoretisch!) das Interpolationspolynom p_n vom Grad $n-1$ zu den Punkten $(h_j, \Phi(h_j))$ für $j = 1, \ldots, n$ betrachtet, d.h. $p_n(h) \approx \Phi(h)$, und dieses mit dem Neville-Verfahren bei $h = 0$ auswertet. Man bezeichnet dieses Vorgehen als *Richardson-Extrapolation des einseitigen Differenzenquotienten.* (Einen Konvergenzbeweis für dieses Verfahren sehen Sie in der Vorlesung zur Numerischen Mathematik.) Mit $h_n := 2^{-n} h_0$ betrachten wir die Folge der $y_n := p_n(0)$. Man schreibe eine Funktion `richardson`, die neben dem Funktionspointer einer Funktion f, den Auswertungspunkt x, die erste Schrittweite $h_0 > 0$ sowie die Toleranz $\tau > 0$ übernimmt und $y_{n+1} \approx f'(x)$ zurückliefert, sobald gilt

$$|y_n - y_{n+1}| \leq \begin{cases} \tau, & \text{falls } |y_{n+1}| \leq \tau, \\ \tau\,|y_{n+1}| & \text{anderenfalls.} \end{cases}$$

Verwenden Sie bei der Realisierung die Funktion `neville` aus Aufgabe 91.

Aufgabe 94. Bei der Richardson-Extrapolation des einseitigen Differenzenquotientens aus Aufgabe 93 kann man sich folgende Beobachtung zum Neville-Verfahren zu Nutze machen: Bei einer effizienten Implementierung des Neville-Verfahrens gemäß Aufgabe 92 überschreibt man den Vektor y durch die Diagonale $(p_{1,n}, p_{2,n-1}, \ldots, p_{n,1})$, und $p_{1,n}$ ist der gesuchte Wert. Fügt man nun einen weiteren Interpolationsknoten (x_{n+1}, y_{n+1}) hinzu, so muss man nicht noch einmal das vollständige Schema rechnen, sondern lediglich die „neue Diagonale“ $(p_{1,n+1}, p_{2,n}, \ldots, p_{n+1,1})$. Mit dieser Beobachtung muss man in jedem Schritt der Richardson-Extrapolation nur noch eine Schleife durchlaufen, nicht mehr zwei!

Aufgabe 95. 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.

Aufgabe 96. Man kombiniere das Aitkin-Verfahren aus Aufgabe 95 mit dem einseitigen Differenzenquotienten $\Phi(h)$ aus Aufgabe 86: 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 $\tau > 0$ übernimmt und $y_{n+1} \approx f'(x)$ zurückliefert, sobald gilt

$$|y_n - y_{n+1}| \leq \begin{cases} \tau, & \text{falls } |y_{n+1}| \leq \tau, \\ \tau\,|y_{n+1}|, & \text{anderenfalls.} \end{cases}$$

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

Aufgabe 97. 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. Elementare Rechnung zeigt dann

$$p = \frac{\log(|x_{n-2} - x|/|x_{n-1} - x|)}{\log(|x_{n-1} - x|/|x_n - x|)} \quad \text{und} \quad c = \frac{|x_{n-1} - x|}{|x_n - x|^p},$$

d.h. aus den Gleichungen für die Fehler $|x_{n-1} - x|$ und $|x_n - x|$ berechnet man die Unbekannten p und c. Leiten Sie diese Gleichheiten her! Schreiben Sie eine Funktion `convorder`, die für eine gegebene Folge $(x_n)_{n=1}^N$ und einen Grenzwert x die experimentelle Konvergenzrate $p, c \in \mathbb{R}^{N-2}$ berechnet und zurückgibt. Wiederholen Sie Aufgabe 84, wobei Sie für jedes der drei Verfahren zur Berechnung der Wurzel $\sqrt{x}$ tabellarisch n, x_n, $|\sqrt{x} - x_n|$ und im Fall $n \geq 2$ auch p und c ausgeben.

Aufgabe 98. Falls der Grenzwert x in Aufgabe 97 unbekannt ist, kann man die Dreiecksungleichung verwenden, um den Fehler $|x_n - x|$ durch die berechenbare Größe $|x_n - x_{n+1}|$ abzuschätzen. Derselbe Ansatz wie in Aufgabe 97 erlaubt dann die Berechnung der experimentellen Konvergenzordnung über die Formeln

$$p = \frac{\log(|x_{n-2} - x_{n-3}|/|x_{n-1} - x_{n-2}|)}{\log(|x_{n-1} - x_{n-2}|/|x_n - x_{n-1}|)} \quad \text{und} \quad c = \frac{|x_{n-1} - x_{n-2}|}{|x_n - x_{n-1}|^p},$$

Modifizeren Sie die Funktion `convorder` so, dass der Grenzwert x ein optionaler Parameter ist: Wird x übergeben, so verwende man die Formel aus Aufgabe 97 und liefere $c, p \in \mathbb{R}^{n-2}$. Wird der Grenzwert nicht übergeben, so liefere die Funktion $c, p \in \mathbb{R}^{n-3}$ gemäß der letzten Formel. Zu optionalen Funktionsparametern siehe `help nargin` und `help varargin`.

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 $|f(y_n)| \leq \tau$ und $|y_n - y_{n-1}| \leq \tau \max\{|y_n|, |y_{n-1}|\}$ 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 83 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.