Dirk Praetorius, Samuel Ferraz-Leite
Christoph Überhuber

Sommersemester 2009
03. Juni 2009

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. 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* `lva.student.tuwien.ac.at` *interpretiert werden können. In den folgenden Aufgaben sollen* **Schleifen** *und* **Function-Handles** *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 $j = 1, \dots, n$. 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

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

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

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

Der mathematische Beweis für diesen Algorithmus folgt in der Vorlesung zur Numerischen Mathematik. Zunächst schreibe man die Funktion so, dass die Matrix $(p_{j,m})_{j,m=1}^{n}$ vollständig aufgebaut wird. Speichern Sie die Funktion unter `neville.m` ins Verzeichnis `serie10`. Sie können den Code testen, indem Sie für ein bekanntes Polynom p als Funktionswerte $y_j = p(x_j)$ wählen.

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`. Speichern Sie den Source-Code unter `neville2.m` in das verzeichnis `serie10`.

Aufgabe 93*. Für eine differenzierbare Funktion $f : [a, b] \to \mathbb{R}$ kann man die Ableitung $f'(x)$ in einem festen Punkt $x \in \mathbb{R}$ durch den einseitigen Differenzenquotienten

$$\Phi(h) := \frac{f(x + h) - f(x)}{h} \quad \text{für } h > 0$$

approximieren. Nach Taylor-Formel gilt für $f \in C^2(\mathbb{R})$ die Fehlerabschätzung $|f'(x) - \Phi(h)| = \mathcal{O}(h)$. Beweisen Sie diese Aussage mathematisch! Schreiben Sie eine Funktion `diff(f,x,h0,tau)`, die für $h_n := 2^{-n} h_0$ die Folge der $\Phi(h_n)$ berechnet, bis gilt

$$|\Phi(h_n) - \Phi(h_{n+1})| \leq \begin{cases} \tau & \text{für } |\Phi(h_n)| \leq \tau, \\ \tau\,|\Phi(h_n)| & \text{anderenfalls.} \end{cases}$$

Die Funktion liefere in diesem Fall die vollständige Folge $(\Phi(h_0), \dots, \Phi(h_n))$ der Iterierten zurück. Speichern Sie den Source-Code unter `diff.m` in das Verzeichnis `serie10`.

Aufgabe 94*. Eine Variante zur Berechnung einer Nullstelle einer Funktion $f : [a, b] \to \mathbb{R}$ ist das *Newton-Verfahren*. Ausgehend von einem Startwert x_0 definiert man induktiv eine Folge (x_n) wie folgt: Zu gegebenem x_k sei x_{k+1} die Nullstelle

der Tangente an den Graphen von f im Punkt $(x_k, f(x_k))$, d.h. $x = x_{k+1}$ erfüllt $0 = f(x_k) + f'(x_k)(x - x_k)$. Auflösen nach x zeigt

$$x_{k+1} = x_k - f(x_k)/f'(x_k).$$

Man realisiere das Newton-Verfahren in einer Funktion `newton(f,fprime,x0,tau)`, wobei die Iteration abgebrochen wird, falls entweder

$$|f'(x_n)| \le \tau$$

oder

$$|f(x_n)| \le \tau \quad \text{und} \quad |x_n - x_{n-1}| \le \begin{cases} \tau & \text{für } |x_n| \le \tau, \\ \tau|x_n| & \text{sonst} \end{cases}$$

gilt. In zweiten Fall werde $\widetilde{z} := x_n$ als Approximation der gesuchten Nullstelle zurückgegeben. Im ersten Fall werde mit Fehlermeldung abgebrochen. — Neben x_n sollen zusätzlich die Folgen $(x_0, \ldots, x_n)$ der Nullstellen und der dazugehörigen Funktionswerte zurückgegeben werden.

Aufgabe 95. Schreiben Sie eine Matlab-Prozedur `testsqrt`, die für gegebenes $x > 0$ und $\tau > 0$ die Wurzel $\sqrt{x}$ einerseits mittels des Bisektionsverfahrens aus der Vorlesung und andererseits mittels der Funktion aus Aufgabe 94 approximiert, d.h. die positive Nullstelle z der Funktion $f(t) := t^2 - x$ numerisch berechnet. Die Prozedur gebe für beide Verfahren den relativen Fehler $|\sqrt{x} - z|/\sqrt{x}$ sowie die Anzahl der benötigten Iterationsschritte aus.

Aufgabe 96. Für $x > 0$ konvergiert die Folge

$$x_1 := \frac{1}{2}(1 + x), \quad x_{n+1} := \frac{1}{2}\Big(x_n + \frac{x}{x_n}\Big) \quad \text{für } n \ge 1$$

gegen $\sqrt{x}$. Man schreibe eine Funktion `sqrt_`, die für gegebene $x > 0$ und $\varepsilon > 0$ als Ergebnis das erste Folgenglied $y = x_n$ zurückgibt, für das gilt

$$\frac{|x_n - x_{n+1}|}{|x_n|} \le \varepsilon \quad \text{oder} \quad |x_n| \le \varepsilon.$$

Welcher Zusammenhang besteht mit dem Newton-Verfahren aus Aufgabe 94?

Aufgabe 97. Das Newton-Verfahren aus Aufgabe 94 benötigt neben der Funktion `f` auch eine Funktion `fprime`, die die Ableitung f' der Funktion f auswertet. Alternativ kann man $f'(x_k)$ durch den Differenzenquotienten $\Phi_h(x_k)$ ersetzen. Man realisiere dieses Vorgehen, z.B. für $h = \tau$. Man erweitere Aufgabe 95 um dieses Verfahren und vergleiche mit den anderen.

Aufgabe 98. Die Quotientenfolge $(a_{n+1}/a_n)_{n\in\mathbb{N}}$ zur Fibonacci-Folge $(a_n)_{n\in\mathbb{N}}$,

$$a_0 := 1, \quad a_1 := 1, \quad a_n := a_{n-1} + a_{n-2} \quad \text{für } n \ge 2,$$

konvergiert gegen den goldenen Schnitt $(1 + \sqrt{5})/2$. Insbesondere konvergiert die Differenz

$$b_n := \frac{a_{n+1}}{a_n} - \frac{a_n}{a_{n-1}}$$

gegen Null. Man schreibe eine Funktion `cauchy`, die zu gegebenem $k \in \mathbb{N}$ die kleinste Zahl $n \in \mathbb{N}$ mit $|b_n| \le 1/k$ zurückgibt.

Aufgabe 99. Der einseitige Differenzenquotient aus Aufgabe 93 konvergiert lediglich mit Ordnung $\alpha = 1$. Beweisen Sie mithilfe der Taylorschen Formel für $f \in \mathcal{C}^3(\mathbb{R})$ die Fehlerabschätzung

$$\frac{f(x + h) - f(x - h)}{2h} - f'(x) = \mathcal{O}(h^2)$$

für den zentralen Differenzenquotienten.

Aufgabe 100. Schreiben Sie eine Funktion `diff2(f,x,h0,tau)`, die für $h_n := 2^{-n}h_0$ die Folge der zentralen Differenzenquotienten

$$\Psi(h_n) := \frac{f(x + h_n) - f(x - h_n)}{2h_n}$$

berechnet, bis gilt

$$|\Psi(h_n) - \Psi(h_{n+1})| \le \begin{cases} \tau & \text{für } |\Psi(h_n)| \le \tau, \\ \tau\,|\Psi(h_n)| & \text{anderenfalls.} \end{cases}$$

Die Funktion liefere in diesem Fall die vollständige Folge $(\Psi(h_0), \ldots, \Psi(h_n))$ der Iterierten zurück. Vergleichen Sie die Funktionen `diff` und `diff2` miteinander. Wie können Sie die mathematische Voraussage über das Konvergenzverhalten überprüfen?