Dirk Praetorius
Michele Ruggeri

Wintersemester 2014/15
18.11.2014

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 8

Aufgabe 8.1. 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. Schreiben Sie eine Funktion `double* diff(double x, double h0, double tau, int* n)`, die für $h_n := 2^{-n} h_0$ die Folge der $\Phi(h_n)$ berechnet, bis gilt

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

Die Funktion soll mit einer beliebigen reellwertigen Funktion `double f(double x)` arbeiten. Die Funktion liefere in diesem Fall die vollständige Folge $(\Phi(h_0), \dots, \Phi(h_n))$ der Iterierten zurück. Beachten Sie, dass auch die Länge des Vektors „zurückgegeben“ werden muss. Speichern Sie den Source-Code unter `diff.c` in das Verzeichnis `serie08`.

Aufgabe 8.2. Schreiben Sie eine Funktion `void unique(double * x, int n)`, die einen Vektor $x \in \mathbb{R}^n$ aufsteigend sortiert, doppelte Einträge streicht und den Vektor in gekürzter Form zurückgibt. Die Funktion soll also beispielsweise den Vektor $x = (4, 3, 5, 1, 4, 3, 4) \in \mathbb{R}^7$ durch den Vektor $x = (1, 3, 4, 5) \in \mathbb{R}^4$ überschreiben. Die Länge n der Vektoren ist dynamisch zu realisieren. Schreiben Sie ein aufrufendes Hauptprogramm, in dem n und $x \in \mathbb{R}^n$ eingelesen werden und das Ergebnis der Funktion `unique` ausgegeben wird. Speichern Sie den Source-Code unter `unique.c` in das Verzeichnis `serie08`.

Aufgabe 8.3. Gegeben sei eine stetige Funktion $f : [a, b] \to \mathbb{R}$, d.h. eine Funktion ohne Sprünge. Es gelte

$$f(a) \cdot f(b) \leq 0.$$

Dann hat f eine Nullstelle z_0, die im Folgenden mittels Bisektion (= Intervallhalbierung) approximiert werden soll: Der Bisektionsalgorithmus arbeitet wie folgt: In jedem Bisektionsschritt definiert man $c := (a + b)/2$ als Intervallmittelpunkt. Aufgrund der Voraussetzung gilt

$$f(a) \cdot f(c) \leq 0 \quad \text{oder} \quad f(c) \cdot f(b) \leq 0.$$

Im Fall $f(a) \cdot f(c) \leq 0$ liegt eine Nullstelle im Intervall $[a, c]$, und man ersetzt daher b durch c. Im Fall $f(c) \cdot f(b) \leq 0$ liegt eine Nullstelle im Intervall $[c, b]$, und man ersetzt daher a durch c. In beiden Fällen geht man also vom Intervall $[a, b]$ zu einem Teilintervall halber Länge über. Schreiben Sie eine Funktion `bisection`, die als Parameter a, b und eine Toleranz $\tau > 0$ übernimmt und den Bisektionsschritt wiederholt, bis man vom Startintervall $[a, b]$ zu einem Intervall $[a, b]$ mit Länge $|b - a| \leq \tau$ übergegangen ist. In diesem Fall gebe man a zurück. Es gilt dann $|a - z_0| \leq |a - b| \leq \tau$, d.h. a ist eine Approximation einer Nullstelle z_0 bis auf eine Genauigkeit von τ. Als Testfunktion verwende man $f(x) = x^2 + \exp(x) - 2$ auf $[0, \infty)$, die man als eigene Funktion realisiere. Schreiben Sie ferner ein Hauptprogramm, das $b, \tau > 0$ einliest und die Approximation von z_0 ausgibt. Speichern Sie den Source-Code unter `bisection.c` in das Verzeichnis `serie08`.

Aufgabe 8.4. Modifizieren Sie das Bisektionsverfahren aus Aufgabe 8.3 so, dass nicht nur die approximative Nullstelle $a \approx z_0$, sondern die gesamte Folge $(a_0, \dots, a_N)$ von Approximationen zurückgegeben wird, d.h. man speichere in jedem Bisektionsschritt die linke Intervallgrenze.

Aufgabe 8.5. Alternativ zum Bisektionsverfahren aus Aufgabe 8.3 kann eine Nullstelle von $f : [a, b] \to \mathbb{R}$ auch mit dem *Sekantenverfahren* berechnet werden. Dabei sind x_0 und x_1 gegebene Startwerte und man definiert induktiv x_{n+1} als Nullstelle der Geraden durch $(x_{n-1}, f(x_{n-1}))$ und $(x_n, f(x_n))$, d.h.

$$x_{n+1} := x_n - f(x_n)\,\frac{x_{n-1} - x_n}{f(x_{n-1}) - f(x_n)}$$

Schreiben Sie eine Funktion `sekante(x0,x1,tau)` die die Folge der Iterierten berechnet, bis entweder

$$|f(x_n) - f(x_{n-1})| \le \tau$$

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. Es werde dann x_n als Approximation einer Nullstelle z_0 von f zurückgegeben. Im ersten Fall gebe man zusätzlich eine Warnung aus, dass das numerische Ergebnis vermutlich falsch ist. Verwenden Sie das Beispiel aus Aufgabe 8.3 zum Test. Schreiben Sie ein aufrufendes Hauptprogramm, in dem x_0 und x_1 eingelesen werden und x_n ausgegeben wird. Speichern Sie den Source-Code unter `sekante.m` in das Verzeichnis `serie08`.

Aufgabe 8.6. 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 $(x_n)_{n\in\mathbb{N}}$ durch

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

Man realisiere das Newton-Verfahren in einer Funktion `newton`, 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. Im ersten Fall gebe man zusätzlich eine Warnung aus, dass das numerische Ergebnis vermutlich falsch ist. Die Funktion soll mit einer beliebigen reellwertigen Funktion `double f(double x)` und Ableitung `double fstrich(double x)` arbeiten. Schreiben Sie ein aufrufendes Hauptprogramm, in dem x_0 eingelesen und x_n ausgegeben wird.Speichern Sie den Source-Code unter `newton.c` in das Verzeichnis `serie08`.

Aufgabe 8.7. Das Newton-Verfahren aus Aufgabe 8.6 benötigt neben der Funktion `f` auch eine Funktion `fstricht`, die die Ableitung f' der Funktion f auswertet. Alternativ kann man $f'(x_k)$ durch den Differenzenquotienten $\Phi_h(x_k)$ aus Aufgabe 8.1 ersetzen. Realisieren Sie dieses Vorgehen indem Sie eine Funktion `newton(f,x0,h0,tau)` schreiben, die zur Approximation der Abeitung $f'(x_k)$ das Ergebnis von `diff(xk,h0,tau,n)` verwendet. Speichern Sie den Source-Code unter `newton2.c` in das Verzeichnis `serie08`.

Aufgabe 8.8. Was ist ein Gleitkommazahlsystem? Aus welchen Bestandteilen setzt sich eine Gleitkommazahl zusammen? Wie bestimmt man daraus ihren Wert? Was verbirgt sich hinter den Symbolen `Inf`, `-Inf` und `NaN`? Was ist die Maschinengenauigkeit `eps`? Was ist eine normalisierte Gleitkommazahl? Was ist ein implizites erstes Bit?