Prof. Dr. Karsten Urban
Dipl.-Math. oec. Klaus Stolle
Institut für Numerische Mathematik
Universität Ulm

SoSe 2017
Blatt 08
11.06.2017

Angewandte Numerik 1

Besprechung in den Tutorien in der Woche vom 19.06.2017 bis 23.06.2017

Für dieses Übungsblatt gibt es 11 Theorie- und 17 Matlab-Punkte, sowie 6 Theorie- und 7 Matlab-Zusatzpunkte. Punkte, die mit einem * gekennzeichnet sind, sind Zusatzpunkte.
Die 60-Prozent-Grenzen liegen aktuell (inklusive Blatt 08) bei 95,4 Theoriepunkten und 74,4 Matlabpunkten.

Aufgabe 32 *(Programmieraufgabe: Bisektionsverfahren)* (4M+3M Punkte)

a) Schreiben Sie eine Matlab-Function `[a, b] = bisektion(f, a, b)`, die die Lösung einer Gleichung $f(x) = 0$ im Intervall $[a, b]$ auf zwei Nachkommastellen genau bestimmt. Rückgabewert soll das auf zwei Nachkommastellen genau bestimmte Intervall sein. Protokollieren Sie den Iterationsverlauf dadurch, dass Sie in jedem Schritt die Intervallgrenzen ausgeben.

b) Testen Sie Ihr Programm an den folgenden Gleichungen und Startwerten:

$$\begin{aligned} x^2 - 2 = 0, &\quad a = 1,\ b = 3, \\ \sin(x) - \cos(2x) = 0, &\quad a = 0,\ b = 1, \\ x^3 - 7x^2 + 11 = 5, &\quad a = 2.7,\ b = 6.5. \end{aligned}$$

Interpretieren Sie den Iterationsverlauf und die berechneten Ergebnisse.

Hinweis: Sie können *anonyme Funktionen* verwenden:
`<Funktionsname> = @(<Argumentliste>) Funktionsbeschreibung` legt eine Variable vom Typ `function handle` an, die dann als Parameter an eine andere Funktion, beispielsweise Ihre Funktion `bisektion`, übergeben werden kann. Für die erste Testgleichung $x^2 - 2 = 0$ legt `f1 = @(x) x^2-2` die Variable `f1` an, die Sie mit `[a, b] = bisektion(f1, a, b)` an die Matlab-Funktion übergeben können.

Aufgabe 33 *(Intervallschachtelung)* (5T Punkte)

Das Bisektionsverfahren beruht auf dem Prinzip der Intervallschachtelung:
Sei $I_0 = [a_0, b_0]$ ein Anfangsintervall. Eine Annäherung an einen Punkt $x^* \in I_0$ kann man mit Hilfe der Intervallschachtelung erreichen. Dazu definiert man iterativ $I_n = [a_n, b_n]$ mit

$$a_{n+1} = \begin{cases} a_n\,, & \text{falls } x^* \le \frac{b_n+a_n}{2} \\ \frac{b_n+a_n}{2}\,, & \text{falls } x^* > \frac{b_n+a_n}{2} \end{cases} \qquad \text{sowie} \qquad b_{n+1} = \begin{cases} \frac{b_n+a_n}{2}\,, & \text{falls } x^* \le \frac{b_n+a_n}{2} \\ b_n\,, & \text{falls } x^* > \frac{b_n+a_n}{2}. \end{cases}$$

Wie viele Schritte benötigt das Bisektionsverfahren für das zweite Beispiel aus Aufgabe 32 (also $I_0 = [0, 1]$), um eine Genauigkeit von mindestens 10^{-8} zu erreichen? Bestimmen Sie $n_0 \in \mathbb{N}$, sodass $|x^* - y| < 10^{-8}$ für alle $y \in I_{n_0}$ gilt.

Aufgabe 34 *(Programmieraufgabe: Konvergenz Bisektionsverfahren)* (2M+3M+2T+5M*+3T* Punkte)

a) Erweitern Sie Ihre Matlab-Funktion `bisektion` aus Aufgabe 32 so, dass Sie die gewünschte Genauigkeit `tol` als Parameter übergeben können. Ferner soll Ihre erweiterte Matlab-Funktion `xk = bisektion1(f, a, b, tol)` einen Vektor `xk` mit den Iterationswerten x_k aller durchgeführten Iterationen zurückgeben.

b) Schreiben Sie ein Matlab-Skript `testKonvergenz`, in dem Sie mit Hilfe Ihrer erweiterten Funktion `bisektion1` die Nullstelle der Funktion $f : \mathbb{R} \to \mathbb{R}$ mit $f(x) = x^2 - \cos x$ auf 10 Nachkommastellen genau berechnen. Wählen Sie als Startintervall das Intervall $[0, 3]$.

Visualisieren Sie, wie sich der Fehler $|x_k - x^*|$ während des Bisektionsverfahrens entwickelt. Plotten Sie dazu zu jedem Iterationsschritt (x-Achse) den jeweiligen Fehler (y-Achse). Wählen Sie zur besseren Darstellung für den Fehler eine logarithmische Skala (Matlab-Befehl `semilogy`).

Sie können für die exakte Lösung x^* das Ergebnis der Matlab-Anweisung `fzero(f, 1)` verwenden, wobei `f` ein function handle für die Funktion f ist.

c) Interpretieren Sie das Ergebnis. Liegt Konvergenz im Sinne der *Definition 5.1.1* vor? Begründen Sie Ihre Aussage.

d) Erweitern Sie Ihr Testprogramm und Ihre Grafik auch um die Regula Falsi und um die Sekantenmethode. Schreiben Sie dazu eine Matlab-Funktion `xk = regulaFalsi(f, a, b, tol, maxIt)`, die die Regula Falsi implementiert, und eine Matlab-Funktion `xk = sekanten(f, a, b, tol, maxIt)`, die die Sekantenmethode implementiert. Ihre Funktionen sollen die gleichen Rückgabewerte und Parameter wie Ihre Funktion `bisektion1` haben. `maxIt` ist eine obere Begrenzung für die Anzahl der durchgeführten Iterationen. Ergänzen Sie Ihre Grafik aus Aufgabenteil b) um die Darstellung der jeweiligen Fehler der Regula Falsi und der Sekantenmethode.

e) Interpretieren Sie die Ergebnisse. Liegt bei diesen beiden Verfahren Konvergenz im Sinne der *Definition 5.1.1* vor? Wie würde sich die Zahl der benötigten Iterationen der einzelnen Verfahren ändern, wenn Sie die gewünschte Genauigkeit `tol` variieren würden.

Aufgabe 35 *(Programmieraufgabe: Newton-Verfahren)* (3T*+3M+2M*+2M+2T+2T Punkte)

a) Lösen Sie das nichtlineare Gleichungssystem $x^2 = 5$ näherungsweise, indem Sie von Hand drei Iterationen des Newton-Verfahrens durchführen. Verwenden Sie als Startwert $x_0 = 1$.

b) Schreiben Sie eine Matlab-Funktion `xk = newton(f, df, x0, tol, maxIt)`, die eine Nullstelle einer Funktion f mit Hilfe des Newton-Verfahrens näherungsweise berechnet. Dabei soll der Parameter `f` die Funktion f als *function handle*, der Parameter `df` die Ableitung der Funktion f als *function handle* und der Parameter `x0` der Startwert sein. Die weiteren Parameter `tol` und `maxIt` sowie der Rückgabewert `xk` sollen die gleiche Bedeutung wie bei den Matlab-Funktionen `regulaFalsi` und `sekanten` aus Aufgabe 32 haben.

c) Testen Sie Ihre Matlab-Funktion `newton` an der Funktion $f(x) = \cos x - x$ und dem Startwert $x_0 = 0.1$.

d) Erweitern Sie Ihr Testprogramm und Ihre Grafik aus Aufgabe 32 auch um das Newtonverfahren. Wählen Sie als Startwert $x_0 = 3$.

e) Interpretieren Sie das Ergebnis.

f) Variieren Sie den Startwert für das Newton-Verfahren. Testen Sie beispielsweise auch den Startwert $x_0 = 0$. Wie erklären Sie sich das Ergebnis?

Hinweise:
Die Programmieraufgaben sind in Matlab zu erstellen. Der Source Code muss strukturiert und dokumentiert sein. Senden Sie **spätestens 24 Stunden vor Ihrem Tutorium** alle Matlab-Files und alle Ergebnisse in einer E-mail mit dem Betreff **Loesung-Blatt08** an **angewandte.numerik@uni-ulm.de**.