Prof. Dr. J. Stöckler

Lehrstuhl VIII
Universität Dortmund

# 6. Blatt Programmierpraktikum
# WS 2003/04 (Stöckler/Charina-Kehrein)

Testierungstermin ist Mittwoch, 28.01.04, 14:00-19:00, M944.
Internetseite:
`www.mathematik.uni-dortmund.de/lsviii/veranstaltungen/num/NP/index.html`

**Aufgabe 1** Zur Berechnung der Nullstellen einer Funktion $f : \mathbb{C} \to \mathbb{C}$ wird das Newton-Verfahren für komplexe Funktionen

$$z_{k+1} = z_k - \frac{f(z_k)}{f'(z_k)},\ k = 0, 1, \ldots$$

benutzt. Falls das Newton-Verfahren für einen Startpunkt $z_0$ konvergiert, dann ist $z^* = \lim_{k\to\infty} z_k$ eine Nullstelle der Funktion $f$.

a) Schreiben Sie eine Matlab-Funktion 'newton.m', die das Newton-Verfahren für komplexe Funktionen implementiert. Die Matlab-Funktion 'newton.m' soll eine weitere Matlab-Funktion aufrufen, die die Funktionswerte von $f$ und $f'$ bestimmt. Der Name dieser weiteren Matlab-Funktion soll einer der Eingabeparameter der Funktion 'newton.m' sein.

b) Schreiben Sie eine Matlab-Funktion

*function [field]=runnewton(f)*

die mit Hilfe 'newton.m' die Nullstellen der Funktion $f$ für die Startpunkte $z_0(x, y) = x + iy$, $x = -2 : 0.01 : 2$, $y = -2 : 0.01 : 2$, bestimmt. Die maximale Anzahl an Iterationen für das Newton-Verfahren soll 20 sein. Der Ausgabeparameter *field* ist ein Array mit Komponenten $field(x, y) = z_{20}$, falls das Verfahren nach 20 Iterationsschritte für $z_0(x, y)$ konvergiert, und $field(x, y) = NaN$ sonst.

c) Schreiben Sie eine Matlab-Funktion

*function [bfield]=check(field,point, tol),*

wobei der Ausgabeparameter $bfield(x, y) = 1$ sei, falls $|point - field(x, y)| <$ tol, und sonst 0 sei.

d) Zeichnen Sie in vier separaten Fenstern die Werte *bfield(x,y)*, $x = -2 : 0.01 : 2$, $y = -2 : 0.01 : 2$, die gleich 1 sind, wenn *point* nacheinander die Nullstellen der Funktion $f(z) = z^3 - 1$ und $NaN$ durchläuft.

**Aufgabe 2** Zu berechnen ist die Lösung $x \in \mathbb{R}^n$ des linearen Gleichungssystems

$$Ax = b, \quad \text{mit } A \in \mathbb{R}^{n\times n},\ b \in \mathbb{R}^n.$$

Schreiben Sie eine Matlab-Funktion 'sor.m' mit dem Funktionskopf *function [x,ite,hist]=sor(A,b,omega,tol,max)*

die für einen vorgegebenen Relaxationsparameter *omega* das obige Gleichungssystem für den Startvektor mit den Komponenten $x_k^0 = b_k/a_{kk}$, $k = 1, \ldots, n$, löst. Der parameter *tol* soll die zu erreichende Fehlertoleranz ($\|x^{t+1} - x^t\|_\infty$) angeben, *max* die maximale Anzahl an Iterationen. Wird die maximale Anzahl überschritten, so soll der Algorithmus mit einer Fehlermeldung abbrechen. Wird stattdessen die erforderliche Toleranz erreicht, so soll in *ite* die Anzahl der benötigten Iterationen abgelegt werden. Die Matrix *hist* beinhaltet dann die bis dahin auftretenden Korrekturen. Dieser soll aber nur dann aufgebaut werden, falls die Ausgabeparameterliste dies erfordert.