Übungen zur Vorlesung „Physik auf dem Computer"
Sommersemester 2018

Übungsgruppenleiter:
Robin Bardakcioglu – `rhb@itp1.uni-stuttgart.de; Do. 14:00 - 15:30 Uhr, 5.331`
Johannes Reiff – `jreiff@itp1.uni-stuttgart.de; Di. 14:00 - 15:30 Uhr, 4.141`
Matthias Feldmaier – `fem@itp1.uni-stuttgart.de; Do. 14:00 - 15:30 Uhr, 4.141`

Übungsblatt 12 Ausgabe: 11. Juli 2018

Dienstagsübung: schriftliche Abgabe 15.07.18, Besprechung 17.07.18
Donnerstagsübungen: schriftliche Abgabe 17.07.18, Besprechung 19.07.18

Aufgabe 28: Eindimensionale Nullstellensuche

a) Laden Sie das Programmpaket der Aufgabe. Es enthält die Datei `nullstellensuche.cc`, die dafür vorbereitet ist, eine Nullstellensuche für eine vorgegebene Funktion mit einer Bisektion, dem Newton-Verfahren und dem Sekanten-Verfahren durchzuführen. Im Unterverzeichnis `Algorithmen` befinden sich drei Dateien, die für den Quelltext der drei Verfahren vorbereitet sind.

   Sie können entweder den Quelltext aus den drei Dateien in das Hauptprogramm kopieren oder die Datei `Makefile` im Verzeichnis der Aufgabe so ergänzen, dass das Compilieren des Programms mit `make` in der vorliegenden Struktur funktioniert. Lesen Sie dazu die Kommentare in der Datei `Makefile`. Es fehlt ein Eintrag, damit `make` das Unterverzeichnis beachtet. Die implizite Regel für `.exe`-Dateien muss ebenfalls noch angepasst werden, um die `.o`-Dateien des Unterverzeichnisses anzupassen.

b) **ToDo:** Vervollständigen Sie den Algorithmus zur Bisektion. Schreiben Sie ihn so, dass er die Konvergenz als erreicht ansieht, wenn sowohl $|f(a)| + |f(b)|$ als auch $|b - a|$ den Wert `tol` nicht übersteigen. Testen Sie den Algorithmus, indem Sie das Programm mit sinnvollen Startwerten laufen lassen.

   (2 Punkt(e), Votier)

c) **ToDo:** Vervollständigen Sie den Algorithmus zum Newton-Verfahren. Wählen Sie als Kriterium für die Konvergenz, dass $|x_n - x_{n-1}|$ und $|f(x_n)|$ den Wert `tol` nicht übersteigen. Testen Sie den Algorithmus, indem Sie das Programm mit sinnvollen Startwerten laufen lassen.

   (2 Punkt(e), Votier)

d) **ToDo:** Vervollständigen Sie den Algorithmus zum Sekanten-Verfahren. Verwenden Sie als Konvergenzkriterium, dass $|x_0 - x_1|$ und $|f(x_0)|$ kleiner sind als der Wert `tol`. Testen Sie den Algorithmus, indem Sie das Programm mit sinnvollen Startwerten laufen lassen.

   (2 Punkt(e), Votier)

e) Fügen Sie dem Programm entsprechend des schon vorhandenen Musters eine neue Funktion und eine Ableitung hinzu, die $f(x) = \cos(x)$ auswerten. Suchen Sie eine Nullstelle von $\cos(x)$ mit dem Newton- und dem Sekantenverfahren, wobei Sie im Newtonverfahren $x_0 = 0$ und im Sekantenverfahren $x_0 = -1$, $x_1 = 1$ verwenden.

ToDo: Beschreiben Sie, was passiert. Erklären Sie, warum die Algorithmen scheitern.

(2 Punkt(e), Votier)

Aufgabe 29: Mehrdimensionale Nullstellensuche

a) Laden Sie aus dem Webangebot zur Vorlesung das Programmpaket zu dieser Aufgabe und öffnen Sie die Datei `Newton.cc`. Sie enthält ein Programm, das für eine vorgegebene Funktion eine mehrdimensionale Nullstellensuche durchführt. Die dafür benötigte Jacobi-Matrix wird numerisch gebildet. Gehen Sie das Programm durch und versuchen Sie, alle Schritte zu verstehen.

ToDo: Lassen Sie das Programm laufen und finden Sie mindestens drei verschiedene Nullstellen der Funktion `funk1`.

(2 Punkt(e), Votier)

b) Legen Sie eine Kopie des Programms an und implementieren Sie in dieser eine Funktion, die die Jacobi-Matrix für die Funktion `funk1` analytisch an einem gegebenen Punkt $\boldsymbol{x}$ auswertet.

ToDo: Bauen Sie diese Funktion in das mehrdimensionale Nullstellenverfahren ein. Verzichten Sie auf die numerische Auswertung. Überzeugen Sie sich, dass Sie dieselben Nullstellen erhalten wie zuvor mit der numerischen Implementierung.

(4 Punkt(e), Votier)

Aufgabe 30: Optimierung mit der Methode des steilsten Abstiegs

a) Laden Sie das Programmpaket zu dieser Aufgabe aus dem Webangebot zur Vorlesung. In diesem finden Sie die Datei `Abstieg.cc`, die für eine mehrdimensionale Optimierung einer Beispielfunktion vorbereitet ist.

ToDo: Ergänzen Sie den fehlenden Algorithmus zur Methode des steilsten Abstiegs mit einer Armijo-Schrittweitensteuerung. Testen Sie Ihren Algorithmus anhand der Beispielfunktion.

(6 Punkt(e), Votier)

b) Legen Sie eine Kopie Ihres Programms an und implementieren Sie in dieser die Suche nach dem Minimum der Rosenbrock-Funktion

$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2 \ .$$

ToDo: Versuchen Sie mit mehreren Startpunkten das Minimum aufzuspüren. Gelingt das immer?

(4 Punkt(e), Votier)

c) Ändern Sie das Programm so ab, dass es in jedem Schritt des Gradientenabstiegsverfahrens (nicht in der Schrittweitensteuerung!) die aktuellen Werte von x, y und $f(x, y)$ in eine Datei schreibt. Ordnen Sie die Daten in Spalten nach dem Muster

$n =$ Nummer der Iteration $\quad x \quad y \quad f(x, y)$

an.

ToDo: Erstellen Sie drei verschiedene Datendateien.

(3 Punkt(e), Votier)

d) Komplexe grafische Darstellungen von Daten lassen sich mit der `matplotlib` aus Python sehr einfach gewinnen. Gehen Sie die Datei `plot_2d.py` durch. Sie erstellt ein Höhenprofil der Rosenbrock-Funktion und zeichnet in dieses mit Hilfe der Datendateien (Wichtig: Sie müssen nach obigem Muster angelegt sein!) ein, wie sich die Gradientenabstiegsmethode den Weg zum Minimum bahnt. Editieren Sie das Python-Skript so, dass es den richtigen Namen Ihrer Datendatei verwendet und eine Abbildung dazu erstellt. Das Python-Skript wird durch den Aufruf von

```
1 python plot_2d.py
```

gestartet.

ToDo: Erklären Sie damit die langsame Konvergenz des Verfahrens und erstellen Sie ebenfalls ein Diagramm, in welchem der Wert von $f(x, y)$ über der Zahl der Iterationen aufgetragen ist.

(3 Punkt(e), Votier)

Beachten Sie, dass Sie für die Bearbeitung dieser Aufgabe eine Python-Installation mit den Modulen `numpy` und `matplotlib` benötigen.