Technische Universität München
Zentrum Mathematik, M1
Prof. Dr. Bastian von Harrach
Dipl.-Math. Boris von Loesch

3. Übung zu *Nichtlineare Optimierung: Grundlagen* (WS 2010/2011)

Aufgabe 3.1: **(ca. 5 Punkte)**
Sei $f\colon \mathbb{R}^n \to \mathbb{R}$ stetig differenzierbar und gleichmäßig konvex. Zeigen Sie, dass das Gradientenverfahren (Algorithmus 2.3.5) entweder endlich in einem globalen Minimum von f terminiert oder die erzeugte Folge (x^k) gegen ein globales Minimum von f konvergiert.
Tipp: Zeigen Sie zuerst, dass alle Niveaumengen einer gleichmäßig konvexen Funktion kompakt sind.

Aufgabe 3.2 (MC): **(ca. 7 Punkte)**
Bewertung: -1/0/1 Punkte für falsche/keine/richtige Antwort.

In dieser Aufgabe sei $f\colon \mathbb{R}^n \to \mathbb{R}$ stets eine stetig differenzierbare Funktion.

a) Es seien $x \in \mathbb{R}^n$ und $s \in \mathbb{R}^n$, wobei s *keine* Abstiegsrichtung von f im Punkt x sei. Dann gilt: $\nexists \tau > 0$ mit $f(x+\sigma s) < f(x)\ \forall \sigma \in (0,\tau]$. **wahr** ☐ **falsch** ☐

b) Wir betrachten das Gradientenverfahren mit Armijoregel und wenden es auf f an. In der Vorlesung wurde die *globale Konvergenz* dieses Verfahrens bewiesen. Also gilt:
Das Verfahren konvergiert gegen das globale Minimum von f. **wahr** ☐ **falsch** ☐

c) Beim allgemeinen Abstiegsverfahren wird gefordert, dass man von x^k ausgehend als Suchrichtung s^k eine Abstiegsrichtung wählt.
Man muss also $s^k = -\lambda \nabla f(x^k)$ wählen mit $\lambda > 0$. **wahr** ☐ **falsch** ☐

d) Wendet man das Gradientenverfahren mit Armijoregel (Algorithmus 2.3.5) auf f an, so gilt: Das Verfahren terminiert endlich oder erzeugt eine Folge, die mindestens einen Häufungspunkt besitzt. **wahr** ☐ **falsch** ☐

e) Es sei $x^0 \in \mathbb{R}^n$ ein Punkt mit $\nabla f(x^0) \neq 0$. Weil der negative Gradient eine Abstiegsrichtung ist, gilt für $x^1 := x^0 - \nabla f(x^0)$: $f(x^1) < f(x^0)$. **wahr** ☐ **falsch** ☐

f) Es sei $x^0 \in \mathbb{R}^n$ Startpunkt für das Gradientenverfahren mit Armijoregel, angewendet auf f. Die Niveaumenge $N_f(x^0)$ sei beschränkt. Das Verfahren erzeuge die Folge (x^k).
Weil $N_f(x^0)$ beschränkt ist, folgt mit dem globalen Konvergenzsatz, dass es eine Teilfolge von (x^k) gibt, die gegen einen stationären Punkt konvergiert. **wahr** ☐ **falsch** ☐

g) Sei (s^k) eine durch Algorithmus 2.4.1 erzeugte Folge von Suchrichtungen mit $s^k = -\nabla f(x^k)/k$.
Jede Teilfolge $(s^k)_K$ ist zulässig. **wahr** ☐ **falsch** ☐

Aufgabe 3.3 (Analysis-Wiederholung): **(ca. 3 Punkte)**
Sei $F\colon \mathbb{R}^n \to \mathbb{R}^m$ eine stetig differenzierbare Funktion. Zeigen Sie die Identität

$$F(y) - F(x) = \int_0^1 F'(x + t(y-x))(y-x)\,\mathrm{d}t.$$

Hinweis: Das Integral über vektorwertige Funktionen ist komponentenweise zu verstehen. Für $G\colon \mathbb{R} \to \mathbb{R}^m$ gilt also

$$\int_a^b G(t)\,\mathrm{d}t := \begin{pmatrix} \int_a^b G_1(t)\,\mathrm{d}t \\ \vdots \\ \int_a^b G_m(t)\,\mathrm{d}t \end{pmatrix}.$$

Aufgabe 3.4 (Programmieraufgabe): **(ca. 7 Punkte)**
Implementieren Sie das Gradientenverfahren aus der Vorlesung (Algorithmus 2.3.5) mit der Armijo-Schrittweitenregel in MATLAB. Teilen Sie dazu das Verfahren in zwei Funktionen auf:

1. Eine Funktion `Armijo`, die bei gegebenen Daten die Schrittweite berechnet und zurückgibt:

   ```
   function [sigma] = Armijo(f, gfx, x, s, beta, gamma)
   ```

   mit Eingabeparameter

   - `f`: Ein Funktionshandle auf die Funktion f (vgl. hierzu *feval* in der Matlab-Hilfe)
   - `gfx`: Der Gradient ausgewertet an der Stelle `x`
   - `x`: Die aktuelle Iterierte
   - `s`: Die Suchrichtung
   - *beta*, *gamma*: Die Parameter der Armijo-Suche.

2. Eine Funktion `GradientenVerf`, welche die Funktion `Armijo` verwendet und folgendes Interface hat:

   ```
   function [x,steps] = GradientenVerf(f, gf, x0, eps, maxsteps)
   ```

   Die Eingabeparameter sind:

   - `f`, `gf`: Ein Funktionshandle auf die Funktion f bzw. den Gradienten von f
   - `x0`: Der Startpunkt x^0
   - *eps*: Die Abbruchtoleranz für die Norm des Gradienten: $\|\nabla f(x^k)\|_2 \leq \varepsilon$
   - `maxsteps`: Die maximale Anzahl an Schritten, die durchgeführt werden sollen.

   a) Testen Sie Ihre Implementierung am Beispiel der *Rosenbrockfunktion*

   $$f(x) := 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

   mit Startpunkt $x^0 = (-1.2, 1)^T$ und Parametern maxsteps $= 10\,000$, $\beta = 0.5$ und $\gamma = 10^{-4}$. Führen Sie das Verfahren für $\varepsilon = 10^{-1}$, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5} aus. Für $\varepsilon = 10^{-3}$ sollte das Verfahren ca. 5231 Iterationen benötigen und den Punkt $x = (0.9992, 0.9984)^T$ zurückgeben.

   b) Lassen Sie die Höhenlinien der Rosenbrockfunktion zeichnen (Matlab-Befehl *contour*) und zeichnen Sie die Folge der durch Ihr Verfahren erzeugten Iterierten (x^k) ein. Können Sie erklären, warum das Gradientenverfahren so ineffizient ist?

Geben Sie bitte Ihren Quellcode und die Ausgabe ab.

Abgabe:

- Bitte geben Sie bis **Mittwoch, den 8.12.2010, 11 Uhr** Ihre Lösungen im Briefkasten im Untergeschoss ab.
- Die Aufgaben werden in den Übungsgruppen vom 8.12.2010 - 16.12.2010 besprochen.