Technische Universität Chemnitz
S. Reiß

Chemnitz, 08.11.2011
Abgabe: 18.11.2011

Grundlagen der Optimierung
Übung 6

1. In der Ebene sei $K = \{x \in \mathbb{R}^2 : \|x\|_2 \leq \frac{1}{2}\}$ eine Kreisfläche und $Q = \{x \in \mathbb{R}^2 : \|x\|_\infty = 2\}$ die Vereinigung der Seiten eines Quadrates. Beschreibe die Minkowski-Summe $K + Q$ (Skizze). (2 Punkte)

2. Zeige: Ist $C \subseteq \mathbb{R}^n$ konvex, dann ist auch das Innere (*interior*) $\operatorname{int} C$ und der Abschluss (*closure*) $\operatorname{cl} C$ konvex. (4 Punkte)

3. Für eine Menge $S \subseteq \mathbb{R}^n$ ist die *konvexe Hülle* $\operatorname{conv}(S) := \bigcap_{\substack{S \subseteq C \subseteq \mathbb{R}^n \\ C \text{ konvex}}} C$. Zeige:

   (a) $\operatorname{conv}(S)$ ist die bzgl. Mengeninklusion kleinste konvexe Menge im $\mathbb{R}^n$, die S enthält. (2 Punkte)

   (b) $\operatorname{conv}(S)$ ist die Menge aller Konvexkombinationen von Punkten aus S, d.h. $\operatorname{conv}(S) =$
   $$\left\{ \sum_{i=1}^{k} \alpha_i x_i : k \in \mathbb{N}, \sum_{i=1}^{k} \alpha_i = 1, x_i \in S \text{ und } \alpha_i \geq 0 \text{ für } i = 1, \ldots, k \right\}.$$
   (2 Punkte)

   (c) Eine Menge $C \subseteq \mathbb{R}^n$ ist genau dann konvex, wenn sie alle (endlichen) Konvexkombinationen ihrer Elemente enthält. (2 Punkte)

4. Die *abgeschlossene konvexe Hülle* $\overline{\operatorname{conv}}\, S$ einer Menge $S \subseteq \mathbb{R}^n$ ist der Schnitt aller abgeschlossenen konvexen Mengen $C \subseteq \mathbb{R}^n$ mit $S \subseteq C$. Zeige: Die abgeschlossene konvexe Hülle von S ist der Abschluss der konvexen Hülle von S ($\overline{\operatorname{conv}}\, S = \operatorname{cl} \operatorname{conv} S$). Gib ein Beispiel einer abgeschlossenen Menge S an, deren konvexe Hülle *nicht* abgeschlossen ist. (4 Punkte)

5. Entwickle und implementiere in Matlab das Verfahren des steilsten Abstiegs. Die Routine soll folgende Form haben:

   ```
   function x = steepest_descent(@fun,xstart).
   ```

   Der Parameter `@fun` sei hierbei ein Handle auf eine Funktion der Form

   ```
   [val,grad]=function_name(point),
   ```

   mit `xstart` werde der Startpunkt übergeben. Der Rückgabewert `x` sei eine Matrix, die die Folge der Iterationspunkte als Spaltenvektoren enthält, d.h. `x` $= [x_1 x_2 \ldots x_k]$. Benutze für linesearch die selbstentwickelte Routine (ergänzt, um eine Rückgabevariable, die angibt, ob ein Wolfepunkt gefunden wurde) oder die Musterlösung von der Lehrveranstaltungsseite. Wähle die Konstanten in den Wolfe-Bedingungen wie folgt: $c_1 = 10^{-4}$, $c_2 = 0.9$. (Vorschlag aus Nocedal/Wright aufgrund praktischer Erfahrungen) Wähle für $\alpha_{\max}$ die Schrittweite des letzten Iterationsschrittes, d. h. $\alpha_{\max} = \frac{\|x_{k+1} - x_k\|}{\|\nabla f(x_{k+1})\|}$. Gibt linesearch keinen Wolfepunkt zurück, so verdopple die Intervalllänge und rufe es erneut auf usw. Abbruch, wenn $\|\nabla f(x_k)\|$ hinreichend klein oder zu viele Iterationen. Bitte nicht das Austesten der Funktion vergessen. (4 Punkte)