Prof. Dr. Dirk Lebiedz
M.Sc. Pascal Heiter

Universität Ulm
Institut für Numerische Mathematik
Wintersemester 2014/2015

Numerische Optimierung - Übungsblatt 1

(Besprechung: Mittwoch, 22. Oktober 2014)

Hinweise

- Die Vorleistung besteht aus der erfolgreichen Bearbeitung (70%) der Aufgaben. Die Erfassung erfolgt durch ein Votiersystem.
- Die Übungen finden wöchentlich statt und haben ein Programmieranteil, der in Zweiergruppen vorgestellt wird.
- Matlabaufgaben sind **zu Zweit** einzureichen, indem Sie eine Mail an den Übungsleiter mit folgendem Betreff senden

  `[NumOpt]: BlattXX - Name1, Name2.`

Aufgabe 1 *(Konvergenz des Gradientenverfahren)*

Sei $A \in \mathbb{R}^{n\times n}$ s.p.d. und $b \in \mathbb{R}^n$. Zeigen Sie, dass das Gradientenverfahren zum Lösen des Optimierungsproblemen

$$\min_{x\in\mathbb{R}^n} \ \frac{1}{2}x^TAx - b^Tx$$

linear konvergiert, d.h.

$$\exists\, 0 < c < 1 : \qquad ||x_{k+1} - x^*||_A \leq c\,||x_k - x^*||_A$$

wobei (x_k) die Iterierten des Gradientenverfahrens sind und x^* die Lösung des Optimierungsproblems ist.

Aufgabe 2 *(Gradientenverfahren)*

a) Implementieren Sie das Gradientenverfahren in `MATLAB` in einer Datei `gradientMethod.m` unter Verwendung der Armijo-Regel zur Schrittweitensteuerung (siehe Homepage - wählen Sie `s = 1.0`) mit folgendem Funktionsaufruf

   ```
   [X,fx,nit] = gradientMethod(f,gradf,x,sigma,beta,tol)
   ```

   mit den Parametern

   - `f` - die Funktion $f : \mathbb{R}^n \to \mathbb{R}$ als *function handle*
   - `gradf` - der Gradient von f als *function handle*
   - `x` - der Startpunkt $x \in \mathbb{R}^n$
   - `sigma,beta` - die Konstanten $\sigma \in (0, 1)$ und $\beta \in (0, 1)$ für die Armijo-Regel
   - `tol` - die Toleranz für das Abbruchkriterium

   Zurückgegeben werden soll das Minimum `x` $= x^*$, der optimale Funktionswert `fx` $= f(x^*)$ und die Anzahl der Iterationsschritte `nit`.

b) Schreiben Sie ein Skript `test_gradient.m`, um diese Methode an der Rosenbrockfunktion

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

   zu testen. Wählen Sie `x = [-1.9,2]`, `tol = 1e-8`, `sigma = 0.5` und `beta = 0.5`.

Aufgabe 3 *(Newtonverfahren)*

a) Implementieren Sie das Newtonverfahren in `MATLAB` in einer Datei `newtonMethod.m` mit und ohne Schrittweitensteuerung (Armijo-Regel) mit folgendem Funktionsaufruf

```
[X,fx,nit] = newtonMethod(f,gradf,hessf,x,tol,sigma,beta)
```

mit den Parametern

- `f` - die Funktion $f : \mathbb{R}^n \to \mathbb{R}$ als *function handle*
- `gradf` - der Gradient von f als *function handle*
- `hessf` - die Hesse-Matrix von f als *function handle*
- `x` - der Startpunkt $x \in \mathbb{R}^n$
- `tol` - die Toleranz für das Abbruchkriterium
- `optional: sigma,beta` - die Konstanten $\sigma \in (0, 1)$ und $\beta \in (0, 1)$ für die Armijo-Regel

Sofern `sigma` und `beta` übergeben wird, soll die Schrittweite gesteuert werden, ansonsten nicht. Zurückgegeben werden soll der gesamte Pfad, d.h. `X` $= [x_0, x_1, x_2, ..., x^*]$, der optimale Funktionswert `fx` $= f(x^*)$ und die Anzahl der Iterationsschritte `nit`.

b) Schreiben Sie ein Skript `test_newton.m`, um diese Methode **mit** Schrittweitensteuerung an der Rosenbrockfunktion

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

zu testen. Wählen Sie `x = [-1.9,2]`, `tol = 1e-8`, `sigma = 0.5` und `beta = 0.5`.

c) Vergleichen Sie die Konvergenzrate des Gradientenverfahren mit der Rate des Newtonverfahren für dieses Beispiel. Was fällt auf?

d) Schreiben Sie ein Skript `test_newton2.m`, welches das Optimierungsproblem

$$\min_{x \in \mathbb{R}} f(x) := x \arctan x - \frac{1}{2} \log(x^2 + 1)$$

mittels Newtonverfahren **ohne** Schrittweitensteuerung löst. Verwenden Sie dazu `tol` $= 10^{-12}$ und die Startwerte $x = 0.5$ und $x = 4$. Diskutieren Sie die Ergebnisse.

e) Schreiben Sie ein Skript `test_newton_ls.m`, dass das Optimerungsproblem aus d) mittels Newtonverfahren **mit** Schrittweitensteuerung löst und verwenden Sie $x_0 = 4$ als Startwert und `tol` $= 10^{-12}$. Diskutieren Sie die numerischen Ergebnisse. Was fällt auf?

Aufgabe 4 *(Quasi-Newton-Verfahren)*

Die Wahl der Suchrichtung d im Punkt x_k ist charakteristisch für numerische Optimierungsalgorithmen. Dabei haben Sie die folgenden zwei Verfahren bereits kennen gelernt:

- Gradientenverfahren: $I_{n \times n} d = -\nabla f(x_k)$
- Newtonverfahren: $\nabla^2 f(x_k) d = -\nabla f(x_k)$.

Ein Quasi-Newton-Verfahren verzichtet explizit auf die Hessematrix und approximiert diese in jedem Schritt. Demnach wird das Gleichungssystem

$$W_k d = -\nabla f(x_k)$$

gelöst, eine zulässige Schrittweite t bestimmt und $x_{k+1} = x_k + td$ berechnet. Danach wird eine Aktualisierung der Approximation der Hessematrix vorgenommen, durch folgende Rechenvorschrift nach Broyden-Fletcher-Goldfarb-Shanno

$$W_{k+1} = W_k + \frac{y_k y_k^T}{y_k^T s} - \frac{W_k (s s^T) W_k}{s^T W_k s} \qquad \textbf{(BFGS)}$$

mit $y_k := \nabla f(x_{k+1}) - \nabla f(x_k)$ und $s = td$. Die Wahl von W_0 ist dabei beliebig. Wählt man $W_0 = I_{n \times n}$, so entspricht der erste Schritt genau dem ersten Schritt des Gradientenverfahrens.

a) Implementieren Sie das Quasi-Newtonverfahren in `MATLAB` in einer Datei `quasiNewtonMethod.m` unter Verwendung der Armijo-Regel zur Schrittweitensteuerung mit folgendem Funktionsaufruf

```
[X,fx,nit] = quasiNewtonMethod(f,gradf,x,tol,sigma,beta)
```

mit den Parametern

- `f` - die Funktion $f : \mathbb{R}^n \to \mathbb{R}$ als *function handle*
- `gradf` - der Gradient von f als *function handle*
- `x` - der Startpunkt $x \in \mathbb{R}^n$
- `tol` - die Toleranz für das Abbruchkriterium
- `sigma,beta` - die Konstanten $\sigma \in (0, 1)$ und $\beta \in (0, 1)$ für die Armijo-Regel

Zurückgegeben werden soll der gesamte Pfad, d.h. `X` $= [x_0, x_1, x_2, ..., x^*]$, der optimale Funktionswert `fx` $= f(x^*)$ und die Anzahl der Iterationsschritte `nit`. Wählen Sie für die Approximation der Hessematrix oben genannte Vorschrift (BFGS) und setzen Sie $W_0 = I_{n\times n}$.

b) Laden Sie die Datei `compareMethods.m` von der Vorlesungshomepage herunter und testen Sie das Skript mit folgenden Parametern

- $\sigma = 0.5$, $\beta = 0.5$ und $x_0 = (1.9, 2)^T$
- $\sigma = 0.9$, $\beta = 0.2$ und $x_0 = (1.9, 2)^T$
- $\sigma = 0.1$, $\beta = 0.5$ und $x_0 = (0, 2)^T$.

Diskutieren Sie die numerischen Ergebnisse und erläutern Sie die Stärken und Schwächen der einzelnen Verfahren.