Universität Konstanz Sommersemester 2014
Fachbereich Mathematik und Statistik
Prof. Dr. Stefan Volkwein
Roberta Mancini, Carmen Gräßle, Laura Lippmann, Kevin Sieg

Optimierung

http://www.math.uni-konstanz.de/numerik/personen/volkwein/teaching/

Program 3 (6 Points)

Submission by E-Mail: 2014/07/16, 10:00 h

Implementation of (Quasi-)Newton method.

Part 1: Implement the (Quasi-)Newton algorithm

```
X = newtonmethod(fhandle, x0, epsilon, t0, alpha, beta, Nmax, amax)
```

according to the algorithm: with initial point `x0`, tolerance `epsilon` for the termination

Algorithm 1 Newton Method

Require: $f : \mathbb{R}^n \to \mathbb{R}$, initial guess x, stopping tolerance $\varepsilon > 0$, maximal iteration number N_{max}, maximal iteration number for Armijo a_{max};

1: $k = 1$;
2: Compute the initial Hessian (exact or $H_k = I$)
3: **while** $tol > \varepsilon$ **and** $n < N_{max}$ **do**
4: Compute a descent direction d^k solving $H_k d_k = -\nabla f(x_k)$;
5: Compute a stepsize t_k using Armijo rule (see Program 1);
6: Set $x^{k+1} = x^k + t_k d^k$;
7: Set $k = k + 1$;
8: Update the Hessian H_k
9: **end while**

condition $\|\nabla f(x_k)\| < \epsilon$, and parameters `t0`, `alpha` and `beta` for the Armijo rule and `Nmax` and `amax` maximum amount of iterations in the loop and Armijo loop.

The program should return a matrix `X = [x0; x1; x2; ...]` containing the iteration history.

Please note: The `newtonmethod` function must accept only one function handle, which provides the needed values! The function should automatically recognize if the Hessian matrix is provided by the functional handle (i.e., no additional parameters or flags should be added in the input parameters of the `newtonmethod` function).

Hint: Use the Matlab function `nargout` to identify if the Hessian is provided by the function.

Part 2: Test your program using two versions of the known `Rosenbrock.m` function (see Program 1 and 2).

1. The first one has to return the Rosenbrock function value and its gradient computed in `x`:

```
function [f, gradf] = rosenbrock1(x);
```

2. The second one has to return in addition the Hessian matrix computed in `x`:

```
function [f, gradf, hessf] = rosenbrock2(x).
```

In the first case, the `newtonmethod` should recognize that the information on the Hessian is missing, so it must provide an approximation of it using BFGS method. The update of the Hessian replacement must be given according to the code in Algorithm 2.

Use the same parameters setting provided in Program 1 but with `N=1000`, using as start-

Algorithm 2 BFGS Update

Require: H, y, s;
1: **if** $y^\top s > 0$ **then**
2: Update $H = H + \frac{yy^\top}{y^\top s} - \frac{Hs(Hs)^\top}{s^\top Hs}$
3: **else**
4: Reset $H = I$
5: **end if**

ing points $[1; -0.5]$ and, then, $[-1.5, -1]$. Explain the results you get and use suitable plots for showing `X`.