Universität Konstanz
Fachbereich Mathematik und Statistik
Prof. Dr. Michael Junk
M. Gubisch, O. Lass, R. Mancini, S. Trenz
Sommersemester 2012

Ausgabe: 02.07.2012
Abgabe: 09.07.2012, 10:00 Uhr, by email

Optimierung
3. Program

Implementation of (Quasi-)Newton method.

Part 1: Implement the (Quasi-)Newton algorithm, according to the lectures:

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

with initial point `x0`, tolerance `epsilon` for the termination condition $\|\nabla f(x_k)\| < \epsilon$, and parameters `t0`, `alpha` and `beta` for the Armijo rule.

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. Use the same parameters setting provided in Program 1. Explain the results you get in both cases and use suitable plots for showing `X`.