Exercises for Lecture Course on Numerical Optimization (NUMOPT)
Albert-Ludwigs-Universität Freiburg – Winter Term 2015-2016

Exercise 6: Newton Type Optimization and Globalization Strategies

(to be completed during exercise session on Dec 2, 2015 or sent by email to dimitris.kouzoupis@imtek.uni-freiburg.de before Dec 4, 2015)

Prof. Dr. Moritz Diehl, Dimitris Kouzoupis and Andrea Zanelli

Aim of this exercise is to implement another Newton type algorithm that we saw in class, namely the BFGS method, and combine it with globalization strategies to regulate its step size.

Exercise Tasks

1. **Convergence of damped Newton's method:** Let f be a twice continuously differentiable function satisfying $LI \succeq \nabla^2 f(x) \succeq mI$ for some $L > m > 0$ and let x^* be the unique minimizer of f over $\mathbb{R}^n$.

   (a) Show that for any $x \in \mathbb{R}^n$:

   $$f(x) - f(x^*) \geq \frac{m}{2}||x - x^*||^2,$$

   (1 point)

   (b) Let $\{x_k\}_{k\geq 0}$ be the sequence generated by the damped Newton's method with constant stepsize $t_k = \frac{m}{L}$. Show that:

   $$f(x_k) - f(x_{k+1}) \geq \frac{m}{2L}\nabla f(x_k)^\top (\nabla^2 f(x_k))^{-1} \nabla f(x_k).$$

   (2 points)

   (c) Show that $x_k \to x^*$ as $k \to \infty$.

   (2 points)

2. **Hanging chain, revisited:** In this task you will solve the unconstrained minimization problem of the hanging chain using the BFGS method in combination with back tracking and the Armijo condition. Consider the non-convex case where the $N-1$ springs can be both compressed and expanded from their rest length $L_i = L/(N-1)$. Recall that in this case the objective function is given by:

   $$V_{\text{chain}}(y, z) = \frac{1}{2}\sum_{i=1}^{N-1} D(\sqrt{(y_i - y_{i+1})^2 + (z_i - z_{i+1})^2} - L_i)^2 + g_0 \sum_{i=1}^{N} m\, z_i. \tag{1}$$

   The provided MATLAB function `[F] = hc_obj(x,param)` returns the value of this nonlinear function for a given set of parameters defined in the data structure `param` and a point `x` ordered as $\mathrm{x} = [y_1^\top, \ldots, y_N^\top, z_1^\top, \ldots, z_N^\top]^\top$.

   (a) Write your own MATLAB function `[F,J] = finite_difference(fun,x,param)` that calculates the function value and the Jacobian of function `fun` at `x` using finite differences. Note that the argument `fun` is a *function handle*. You can then call your function using the syntax `[F,J] = finite_difference(x,@hc_fun,param)` to evaluate the Jacobian of our objective at `x`.

   (2 points)

(b) Complete the template file `main.m` to find the rest position of the hanging chain using the steepest descent method with backtracking and the Armijo condition.

(2 points)

(c) Now update your code to perform BFGS updates on your Hessian approximation how many iterations does your new scheme need to converge?

Remark: The BFGS Hessian approximation is guaranteed to be positive-definite if and only if the curvature condition $s^T y > 0$ holds. A common workaround to ensure that the search direction is always a descent direction is to check weather this condition holds or not and to skip the BFGS update in case positive-definiteness is not guaranteed.

(2 points)

This sheet gives in total 11 points.