V. Schulz
M. Klar

SS 2022
Submission: 24.05.21, until 12:15

Numerical Optimization - Sheet 7

If you are a student in mathematics please solve the exercises with no tag and the ones with the tag **Mathematics**. If you are a data science student please solve the problems with no tag and those with the tag **Data Science**. Submissions with tags other than your subject count as bonus points. The tag **Programming** marks programming exercises.

Ex 1 **Data Science** (3 Points)

Show that for the problem

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

for $Q \in \mathbb{R}^{n\times n}$ positive definite and symmetric and $b \in \mathbb{R}^n$, the Newton method converges after one step.

Solution 1:
The optimal point is fulfills

$$\begin{aligned}\nabla f(x^*) &= Qx^* - b = 0\\ \Rightarrow x^* &= Q^{-1}b\end{aligned}$$

The first iterate of the Newton method is given by

$$x_1 = x_0 + d = x_0 + Q^{-1}(-\nabla f(x_0)) = x_0 - Q^{-1}(Qx_0 - b) = x^*.$$

Ex 2 (3 Points)

Apply Theorem 3.9 and Corollary 3.10 to show that a Newton method with Wolfe line search converges if the spectra of the Hessians satisfy $\sigma(\text{Hess}f(x^k)) \subset [m, L]$ with $0 < m \leq L$ independent of k.

Solution 2:
As

$$-\nabla f_k d_k = \nabla f_k \ \text{Hess}f_k^{-1}\nabla f_k \geq \frac{1}{L}\|\nabla f_k\|^2$$

and

$$\|d_k\| = \|Hessf_k^{-1}\nabla f_k\| \leq \|Hessf_k^{-1}\|\|\nabla f_k\| \leq \frac{1}{m}\|\nabla f_k\|$$

we obtain

$$\cos\vartheta_k := \frac{-\nabla f_k d_k}{\|\nabla f_k\|\|d_k\|} \geq \frac{\|\nabla f_k\|^2}{\|\nabla f_k\|\|d_k\|L} \geq \frac{m}{L} =: c_3 > 0.$$

Hence the conditions of Theorem 3.9 and Corollary 3.10 are fulfilled and Corollary 3.10 can be applied which yields the assertion.

Ex 3 **Programming** (12 Points)

Implement the following quasi Newton and the Newton method. Your function has to provide the interface `minimization_alg(f, gf, Hessf, x0, callback)` where the first three arguments are callable objects which evaluate a function, its gradient and Hessian with respect to the Euclidean scalar product. The value `x0` is the starting value and `callback` is a callback function (see below).

(i) **SR1: Broydens symmetric rank-1-update**
Implement `broydensymm1(f, gf, Hessf, x0, callback)` and run `test(broydensymm1)`.

(ii) **BFGS (Broyden-Fletcher-Goldfarb-Shanno)**
Implement `bfgs(f, gf, Hessf, x0, callback)` and run `test(bfgs)`.

(iii) **Newton method**
Implement `newton(f, gf, Hessf, x0, callback)` and run `test(newton)`.

Hints:

- You might want to set an error tolerance and a maximal number of iterations as optional parameters of your algorithm.
- Please call the function `callback` in your algorithm in the form `callback(xk)` where `xk` is the current iterate. This allows the function `test` in the module `newton_cg_test` to plot your iterates and the errors.
- You find the function `armijo` in `newton_cg_test`.
- The broyden and bfgs update fail under certain conditions (see lecture). If that happens simply restart the current Hessian approximation by `B = np.eye(n)`.