V. Schulz
M. Klar

SS 2022
Submission: 14.06.21, until 12:15

Numerical Optimization - Sheet 9

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 (4 Points)

Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex and twice continuously differentiable and let $L > 0$ be a uniform upper bound for the largest Eigenvalue of $\mathrm{Hess}f(x)$ for all $x \in \mathbb{R}^n$. Show that ∇f is Lipschitz continuous.

Hint: Use the fundamental theorem of calculus for $\phi : [0,1] \to \mathbb{R}^n$, $\phi(t) := \nabla f(x - t(x-y))$.

Solution 1:
We need to show that for all $x, y \in \mathbb{R}^n$

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|. \tag{1}$$

We define $\phi : [0,1] \to \mathbb{R}^n$,

$$\phi(t) := \nabla f(x - t(x-y)) \tag{2}$$

and obtain that due to

$$\nabla f(y) - \nabla f(x) = \phi(1) - \phi(0) = \int_0^1 \phi'(t)dt = \int_0^1 \mathrm{Hess}f(x - t(x-y))(y-x)dt, \tag{3}$$

that

$$\|\nabla f(y) - \nabla f(x)\| \leq \int_0^1 \|\mathrm{Hess}f(x - t(x-y))\|dt \; \|y - x\|.$$

As (for any $z \in \mathbb{R}^n$) $\mathrm{Hess}f(z)$ is symmetric and positive semi-definite, and $\sigma(\mathrm{Hess}f(z)) \leq L$ we have that $\|\mathrm{Hess}f(z)\| \leq L$. We can therefore conclude

$$\|\nabla f(y) - \nabla f(x)\| \leq L\|y - x\|.$$

Ex 2 (4 Points)

Assume the situation in Chapter 3.5 (Gauss-Newton Method). Let $\tau, \sigma > 0$ and $q \in \mathbb{R}$. You are given the function

$$L(q) := \frac{1}{\sqrt{2\pi}\tau} \exp\left(-\frac{1}{2}\frac{q^2}{\tau^2}\right) \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{1}{2}\sum_{i=1}^n \frac{(\overline{\alpha}_i - \alpha(t_i, q))^2}{\sigma^2}\right),$$

and the optimization problem

$$\max_q L(q).$$

Reformulate the problem as *regularized* non-linear least squares problem (i.e. least squares objective plus something else) analogously to the lecture.

Solution 2:

$$
\begin{aligned}
&\max_q L(q) \\
\Leftrightarrow &\min_q -\ln(L(q)) \\
\Leftrightarrow &\min_q -\left[\ln\left(\frac{1}{\sqrt{2\pi\tau}}\right) - \frac{1}{2}\frac{q^2}{\tau^2} + n\ln\left(\frac{1}{\sqrt{2\pi\sigma}}\right) - \frac{1}{2}\sum_{i=1}^{n}\frac{(\overline{\alpha}_i - \alpha(t_i,q))^2}{\sigma^2}\right] \\
\Leftrightarrow &\min_q \frac{1}{2}\frac{q^2}{\tau^2} + \frac{1}{2}\sum_{i=1}^{n}\frac{(\overline{\alpha}_i - \alpha(t_i,q))^2}{\sigma^2} \\
\Leftrightarrow &\min_q \frac{1}{2}\frac{\sigma^2}{\tau^2}\|q\|_2^2 + \frac{1}{2}\|F(q)\|_2^2
\end{aligned}
$$

Ex 3 (4 Points)

Let $r_i : \mathbb{R} \to \mathbb{R}, f \mapsto r_i(f)$ for $i = 1,\ldots,n$ be strongly convex and twice continuously differentiable. You are given the optimization problem

$$\min_{x\in\mathbb{R}^d} \sum_{i=1}^{n} r_i(F_i(x)), \tag{4}$$

where each $F_i(x)$ is $\mathbb{R}-$valued and twice continuously differentiable for $i = 1,\ldots,n$. Assume that the derivative

$$\frac{\partial r_i}{\partial f}(F_i(x^*)) \approx 0$$

in the solution $x^* \in \mathbb{R}^n$ and derive a variant of the Newton algorithm for optimization analogous to Gauss-Newton which is tailored to the above problem.

Solution 3:
We define

$$f(x) := \sum_{i=1}^{n} r_i(F_i(x)), \tag{5}$$

and obtain the following gradients with respect to the Euclidean scalar product

$$\nabla f(x) = \sum_{i=1}^{n} r_i'(F_i(x))\nabla F_i(x),$$

and

$$\text{Hess } f(x) = \sum_{i=1}^{n} \nabla F_i(x)\text{diag}[r_i''(F_i(x))]\nabla F_i(x)^\top + \sum_{i=1}^{n} \text{Hess } F_i(x)\ r_i'(F_i(x)).$$

Due to the assumption that $r_i'(F_i(x^*)) \approx 0$ we can ignore the second term and use as approximation for the Hessian only the first part. The diagonal matrix $\text{diag}[r''(F_i(x))]$ is always coercive as r is strongly convex.

Ex 4 **Programming** (9 + 2 Bonus Points)

The module `gauss_newton` contains a function `generate_probabilities(gamma=0)` which generates a data set $(t_i, \overline{\alpha}_i^\gamma)$ where $t_i \in \mathbb{R}$ and $\overline{\alpha}_i^\gamma \in (0,1)$ for $i = 1, \ldots, 10$. The parameter γ controls the noise in the data. If $\gamma = 0$ there is no noise. The data is modeled by a function

$$F_i(x) = \sigma(x_1 t_i + x_2) \in (0,1) \text{ for } i = 1, \ldots, 10,$$

where σ is the sigmoidal function $\sigma(t) = \frac{1}{1+e^{-t}}$ with derivative $\sigma'(t) = \sigma(t)(1 - \sigma(t))$.

(i) Solve the problem

$$\min_{x \in \mathbb{R}^2} \sum_{i=1}^{n} r_i(F_i(x)),$$

for

$$r_i(F_i(x)) = -\log(F_i(x))\overline{\alpha}_i - \log(1 - F_i(x))(1 - \overline{\alpha}_i),$$

and a data set with $\gamma = 0$. To that end implement by yourself a variant of the Newton algorithm for optimization analogous to Gauss-Newton (see Exercise 3). Iterate until the size of the search direction is sufficiently small, i.e. until $\|\Delta x_k\| < \delta$ for some tolerance $\delta > 0$. The solution is $x^* = (6, 1)$.

(ii) Solve the above problem also for γ = 1.e-1, 1.e-2 and 0. Plot the norms of the search directions $\|\Delta x_k\|$ against the iteration count k and use a logarithmic scale in the $y-$axis.

(iii) (Bonus) Explain the connection between the above optimization problem and the Kullback-Leibler divergence.

Hint: The module `gauss_newton` contains the functions `generate_probabilities(gamma=0)`, `armijo()`, `sigmoidal()`, and `dsigmoidal()`.