V. Schulz
M. Klar

SS 2022
Submission: 21.06.21, until 12:15

Numerical Optimization - Sheet 10

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 (3 Points)

Let $f, g : \mathbb{R} \to \mathbb{R}$ be smooth functions. Consider the problem

$$\min_{\alpha\in\mathbb{R}} g(f(\alpha)) =: \min_{\alpha\in\mathbb{R}} F(\alpha).$$

(i) The concatenation $g(f(\alpha))$ can be made implicit by the constraints $g(c)$ s.t. $f(\alpha) - c = 0$. Rewrite the given problem as constrained optimzation problem without explicit concatenation (Don't forget the additional optimization parameter c).

(ii) Compute the first order necessary optimality conditions (Theorem 2.24) of the constrained problem.

(iii) Draw a connection between the adjoint λ in the optimal point and the derivative of F.

Solution 1:

(i)

$$\begin{aligned} &\min_{\alpha,c\in\mathbb{R}} g(c) \\ &\text{s.t } c = f(\alpha). \end{aligned}$$

(ii) The lagrangian is of the form

$$\mathcal{L}(\alpha, c, \lambda) = g(c) + \lambda(f(\alpha) - \hat{c}),$$

hence the KKT-Conditions state in particular, that

$$\nabla_\lambda \mathcal{L}(\hat{\alpha}, \hat{c}, \hat{\lambda}) = f(\hat{\alpha}) - \hat{c} = 0, \quad (1)$$

$$\nabla_c \mathcal{L}(\hat{\alpha}, \hat{c}, \hat{\lambda}) = g'(\hat{c}) - \hat{\lambda} = 0. \quad (2)$$

The derivative of F is given by the chain rule

$$F' = g'(f(\alpha))f'(\alpha).$$

Hence, in a stationary point we find $\hat{c} = f(\hat{\alpha})$ and therefore $\hat{\lambda} = g'(f(\hat{\alpha}))$, which is the first factor in the derivative of F.

Remark: In the language of deep neural networks we would say that the value $f(\alpha)$ is back-propagated. This propagation is clearly visible due to the adjoint λ in case of the constrained reformulation.

Ex 2 **Programming** (2+2 Points)

Consider the example *Warm-up: numpy* on the pytorch tutorials page. The tutorial solves

$$\min_{a,b,c,d\in\mathbb{R}} \sum_{i=1}^{N} \|a + bx_i + cx_i^2 + dx_i^3 - \sin(x_i)\|_2^2, \tag{3}$$

for samples $x_i \in [-\pi, \pi]$ using a gradient descent method.

(i) Propose a well known, classical method to solve the above problem in one step (you don't need to implement it).

(ii) Change the tutorial's code such that it solves

$$\min_{a,b,c,d\in\mathbb{R}} \sum_{i=1}^{N} \|a + b\,\sigma(c + dx_i) - \operatorname{sign}(x_i + 0.5)\|_2^2, \tag{4}$$

where $\sigma(x) = 1/(1 + exp(-x))$ with $\sigma' = \sigma(1 - \sigma)$, and

$$\operatorname{sign}(x) = \begin{cases} \frac{x}{|x|},\ x \neq 0 \\ 0, \text{ otherwise,} \end{cases}$$

on the same interval $x_i \in [-\pi, \pi]$.

Ex 3 (2+1+1 Points)

This exercise is a simple example for a minimization problem over a random variable where the stochastic gradient (SG) method with constant step size does not converge. To that end, let $f_r : [-1, 1] \to \mathbb{R}$ be defined as $x \mapsto (x + r)^2$ for $r \in [-1, 1]$. Consider the optimization problem

$$\min_{x\in[-1,1]} \int_{-1}^{1} f_r(x)\, \frac{1}{2} dr. \tag{5}$$

(i) Compute the exact solution of problem (5) by hand.

(ii) Show that for a constant step size $0 < \alpha \leq 0.5$ the iterates x^k of the SG-method do not leave the domain $[-1, 1]$.

(iii) Show that for a constant step size $0 < \alpha \leq 0.5$ the algorithm does not converge to the solution.

Remark: The above problem can be interpreted as

$$\min_{x\in[-1,1]} \mathbb{E} f(x)$$

if we assume that $r \sim \mathcal{U}[-1, 1]$ is uniformly distributed on $[-1, 1]$. In order to show, that the algorithm does not converge it suffices to consider the distribution of $x^+ = x^* - \alpha \nabla f_r(x^*)$ where x^* already is the solution.

Solution 3:

(i) In this part we compute $f(x) := \mathbb{E}_r f_r(x)$. Note, that in practice this function is not known, and the stochastic gradient method does not depend on the knowledge of this function.

$$
\begin{aligned}
f(x) &= \int_{-1}^{1} f_r(x)\frac{1}{2}dr = \int_{-1}^{1} (x+r)^2\frac{1}{2}dr \\
&= \int_{-1}^{1} (x^2 + 2xr + r^2)\frac{1}{2}dr = x^2 + \left[\frac{2}{4}xr^2\right]_{-1}^{1} + \left[\frac{1}{6}r^3\right]_{-1}^{1} \\
&= x^2 + \frac{1}{3}.
\end{aligned}
$$

Now as we know f we ca compute the solution of problem (5), which is $x^* = 0$.

(ii) We now want to apply the SG-method. Before we can do so we make (a rather technical) observation, namely: *If we apply the SG-method the iterates x_k stay in $[-1, 1]$.*

The SG-method for parameter x and a random function $f_r(x)$ is given by

```
choose a starting value x = x0
choose a learning rate alpha

for k=1,...,maxit:
    choose a random number r
    evaluate g_r = gradient(f_r(x))
    update x = x - alpha * g_r
```

The derivative of f_r at x is given by

$$g(x) = 2(x + r).$$

Hence, if $x_1 \in [-1, 1]$ we obtain via induction that

$$|x_{k+1}| = |x_k - 2\alpha(x_k + r_k)| = |x_k(1 - 2\alpha) + 2\alpha r_k| \leq |x_k|(1 - 2\alpha) + 2\alpha|r_k| \leq 1.$$

This means that we can safely apply the SG-method as the iterate x_k will not leave the domain $[-1, 1]$.

(iii) As given in the hint we only consider the distribution of $x_{k+1} = x_k - \alpha\nabla f_r(x_k)$, where $x_k = x^* = 0$ is already the solution. We show now that the variance $\mathbb{E}\|x^* - x^+\|^2$ is positive and that it does not depend on k, so that there is no convergence.

As $x^* = 0$ and we assume that $x_k = 0$ we get that

$$\mathbb{E}\|x^* - x_{k+1}\|^2 = \mathbb{E}\|x^* - (x_k - 2\alpha(x_k + r))\|^2 = \mathbb{E}\|2\alpha r\|^2,$$

where the term on the left does not depend on k anymore. Finally we can compute

$$
\begin{aligned}
\mathbb{E}\|2\alpha r\|^2 &= \int_{-1}^{1} (2\alpha r)^2\frac{1}{2}\,dr = \int_{-1}^{1} 4r^2\alpha^2\frac{1}{2}\,dr \\
&= 2\alpha^2\left[\frac{1}{3}r^3\right]_{-1}^{1} = \frac{4\alpha^2}{3} > 0,
\end{aligned}
$$

which again does not depend on k. Hence, there is no convergence, and we would observe an oscillating behavior even when the algorithm gets close to the solution. The oscillation would be smaller for smaller α but it cannot vanish.

Ex 4 (2* Bonus- Points)

Assume you are given an input sample $x^{(i)} \in \mathbb{R}^n$ which could be an image or sound etc. which you feed into a trained neural network $N : \mathbb{R}^n \to \mathbb{R}^d$. So we interpret the network as fuction which maps an input to a probability distribution on $\{1, \ldots, d\}$ and forget its parameterization for now. The input sample $x^{(i)} \in \mathbb{R}^n$ has a true class, namley $i \in \{1, \ldots, d\}$. Assume for now that the network maps the sample to its correct class, i.e. assigns the highest probability to class i. The following optimization problem aims to find a distortion of $x^{(i)}$ by a small vector $\Delta x \in \mathbb{R}^n$.

Let $0 < \varepsilon < 1$, and $k, i \in \{1, \ldots, d\}$, $k \neq i$. Let moreover $x^{(i)} \in \mathbb{R}^n$ be a fixed input sample of class i and $N_j(x)$ be the j-th component of $N(x)$ for $j \in \{1, \ldots, d\}$. Consider the problem

$$\begin{aligned} &\min_{\Delta x \in \mathbb{R}^n} \|\Delta x\|_2^2 \\ &\text{s.t. for all } j \in \{1, \ldots, d\},\ j \neq k \\ &N_k(x^{(i)} + \Delta x) \geq (1 + \varepsilon) N_j(x^{(i)} + \Delta x). \end{aligned}$$

What is the purpose of the given problem? What would be the consequence of the accessibility of very good solutions to it?

Solution 4:

(i) The Problem tries to find the minimal distortion of $x^{(i)}$ such that the network classifies it as class $k \neq i$. A solution $x^{(i)} + \Delta x$ is called (targeted) adversarial example.

(ii) It seems that many network architectures are prone to adversarials which are that small, that they are unrecognizable for humans. Hence a foto or voice can look or sound unchanged while beeing recognized as totally different for the network.