Barbara Hammer, Fabian Hinder
Universität Bielefeld

Applied Optimization (WS 2024/2025)
Exercise Sheet No. 2

Upload Date: 2024-10-23.
Submission Deadline: In groups of three until **2024-11-06, 7:55** to the Moodle.
Return date: 2024-11-11/14 in the tutorials.

In case you have questions do not hesitate to ask your tutor or to contact the tutor team at apopt@techfak.uni-bielefeld.de.

Remark: For each task, do not only provide the final solution but also a full derivation ("step-by-step") for your solution.

1 Analytic Optimization

(*4 points*)

Consider the minimization problem

$$\min_{x\in\mathbb{R}} \quad \left(1-\frac{1}{2}x^2\right)\cdot\exp\left(-\frac{1}{2}x^2\right)$$

(a) *(1 pts.)* Compute the first and second derivative of the objective function with respect to x.

(b) *(1 pts.)* Set the first derivative equal to zero and solve for x.

(c) *(1 pts.)* Inspect the second derivative of your solutions and identify local minima.

(d) *(1 pts.)* Check the limits of the domain against your local minima and thus verify that you found global minima.

2 Geometric Gradients

(*2 points*)

Let $a, b \in \mathbb{R}^2$ be two points in general position and $x \in \mathbb{R}^2$ another point. Denote by $\theta(x) = \angle(\overline{bx}, \overline{ba})$ the angle around b between x and a. Find the direction of the gradient of θ. Find the direction of the gradient of θ, i.e., the unit-length vector that points in the same direction as the gradient up to a sign.
Hint: Recall that the gradient always points in the direction of steepest ascent. Thus, first, determine which direction does not change the function and is thus orthogonal to the gradient.

Remark: The pseudo code of the gradient descent algorithm:

1: **function** GRADIENT_DESCENT(gradient function $\nabla_{\vec{x}} f : \mathbb{R}^m \to \mathbb{R}^m$, starting value $\vec{x} \in \mathbb{R}^m$, step size η, stopping threshold ϵ)
2: **while** $\|\nabla_{\vec{x}} f(\vec{x})\| > \epsilon$ **do**
3: $\vec{x} \leftarrow \vec{x} - \eta \cdot \nabla_{\vec{x}} f(\vec{x})$
4: **end while**
5: **return** $\vec{x}$.
6: **end function**

We refer to line 3**:** as the optimization step and to η as the learning rate.
Remark: Recall that the limit $\lim_{x\to y} f(x)$ exists if and only if there exists a constant c such that for every sequence x_n with $x_n \xrightarrow{n\to\infty} y$ it holds $f(x_n) \xrightarrow{n\to\infty} c$. In this case we define $\lim_{x\to y} f(x) := c$.
Remark: Recall that a function f is differentiable at x_0 if and only if the limit $\lim\limits_{h\to 0} \frac{f(x_0+h)-f(x_0)}{h}$ exists; we call this limit the derivative of f at x_0.

3 Gradient Based Optimization

(*8 points*)

(a) *(2 pts.)* Consider the function $f(x) = \tanh((x+1.0)(x-0.8)(x+0.1))$. Plot the function $d(t) = (f(t) - f(0))/t$ and mark the value $f'(0)$ in the same plot. What do you observed for the values of d close to 0? And why?

(b) *(2 pts.)* Let $f : \mathbb{R} \to \mathbb{R}$ be a continuous differentiable function and assume that $f'(x_0) \neq 0$. Give an graphical explanation why there has to exist an $\eta > 0$ such that

$$f(x_0) > f\left(x_0 - \eta f'(x_0)\right)$$

Hint: Recall that $f'(x_0)$ is the slope of f at x_0

(c) *(2 pts.)* Let $f : \mathbb{R}^n \to \mathbb{R}$ be a continuous differentiable function. Assuming that $\nabla_{\mathbf{x}} f(\mathbf{x}_0) \neq 0$, why does there exists an $\eta > 0$ such that

$$f(\mathbf{x}_0) > f\left(\mathbf{x}_0 - \eta \nabla_{\mathbf{x}} f(\mathbf{x}_0)\right)?$$

What is the relationship between this inequality and the gradient descent optimization step? An informal argument is sufficient, a mathematical proof is accepted as well, of course.
Hint: Use part (b) and consider the function $g(t) = f\left(\mathbf{x}_0 - t\nabla_{\mathbf{x}} f(\mathbf{x}_0)\right)$ at $t = 0$.

(d) *(2 pts.)* Consider the function $f(\mathbf{x}) = 1/100(x_1^2+x_2^2) - 1/2(\cos(3x_1 - 3x_2) + \cos(3x_1 + 3x_2))$, here $\mathbf{x} = (x_1, x_2)^\top$ are the components. Calculate the gradient $\nabla_{\mathbf{x}} f$ of f at $\mathbf{x}_0 = (0.1, 5)$. Plot the function $g(t) = f(\mathbf{x}_0 + t\nabla_{\mathbf{x}} f(\mathbf{x}_0))$ in some neighborhood of 0. What is the relationship between the minima of g and f?
Hint: We have already studied the function f on the last sheet.

4 Modeling

(*6 points*)

You observe the population size $x(t)$ of an animal species on a group of islands over time t. The animals are not preyed upon by any predator and you regularly add a fixed number of individuals $c > 0$ that is the same for each island. Also, each island i provides different food sources, resulting in different growth rates r_i of the local population. All in all, for a growth rate $r \in \mathbb{R}$ and an addition amount c, we can approximate the population size for each island as a function of time given by the following differential equation: $\frac{\mathrm{d}x}{\mathrm{d}t} = rx + c$ which has the solution

$$x_{r,c,k}(t) = k\exp(rt) - \frac{c}{r},$$

where k is an additional parameter.

Suppose you are given a data set that documents the time of observation t, the island ID i, and the number of individuals $x_i(t)$. Use this data to estimate the growth rate r_i for each island, i.e. find r_i, c, k_i such that $x_{r_i,c,k_i}(t) \approx x_i(t)$ for all i. Keep in mind, that the model is only an approximation.

(a) *(2 pts.)* Formalize the problem described above. Explain your choices! Note that the above description does not lead to a unique formalization, but there are a few issues which you can decide in a reasonable way.
Hint: You also have to argue for the usage of error functions like MSE or MAE.

(b) *(0.5 pts.)* Turn your formalization into an optimization problem in standard from.

(c) *(2 pts.)* Implement your approach: Write a Python-script that randomly generates a starting point x_0 according to a normal distribution and optimize your function using an optimization algorithm of your choice to obtain a point x_1. We provide the dataset in the Moodle. You may use Numpy-method `loadtxt('population.csv', skiprows=1,delimiter=',')` to load it, the column ordering is `time`, `island_id`, `population_size`.

(d) *(1.5 pts.)* Discuss your results. Do you obtain a good solution? What difficulties did you encounter?