OPTIMIZATION - OPTIMALISATIE

EXERCISE SESSION 4: GRADIENT DESCENT AND QUASI-NEWTON METHODS

Note: The content of this LaTeX file can also be found in the accompanying Jupyter Notebook script `S4-GD-qNewton.ipynb`.

In Exercise session 2, we have used *scipy.optimize.linprog* to solve linear programs. In this Exercise Session, we will write our own minimization algorithms for minimizing nonlinear objective functions, ignoring constraints for now. More specifically, we will handle problems of the form

$$\min_{x \in \mathbb{R}^n} f(x),$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is a convex, continuously differentiable function.

To compare the different optimization methods considered in this exercise session, we introduce three different functions which will be minimized.

The first function is a simple toy function, given by

$$f_{\text{toy}}(x) = x_0^2 + 2x_1^2$$

of which the gradient is given by

$$\nabla f_{\text{toy}}(x) = \begin{bmatrix} 2x_0 \\ 4x_1 \end{bmatrix}$$

and the Hessian by

$$\nabla^2 f_{\text{toy}}(x) = \begin{bmatrix} 2 & \\ & 4 \end{bmatrix}.$$

This function has a single global minimizer at $x^\star = (0, 0)^\top$.

Exercise 1 (**Rosenbrock function).**

The second function considered in this session is the Rosenbrock function, given by

$$f_{\text{rosen}}(x) = 100\left(x_2 - x_1^2\right)^2 + (1 - x_1)^2$$

(a) Compute the gradient of f_{rosen}.

(b) Compute the Hessian of f_{rosen}.

(c) Show that $x^* = (1, 1)^T$ is the only stationary point of the function.

(d) Show that the Hessian matrix at x^* is positive definite. Deduce that x^* is a local minimizer.

(e) Implement $f_{\text{rosen}}, \nabla f_{\text{rosen}}$ and $\nabla^2 f_{\text{rosen}}$ in `S4-GD-qNewton.ipynb`.

Exercises 2-5 (**Gradient descent and quasi-Newton methods).**

In these exercises, you will implement the following methods for unconstrained optimization and test their performance on the previously introduced f_{toy} and f_{rosen} functions.

(Exercise 2) Gradient descent method

(Exercise 3) Gradient descent with Armijo linesearch

(Exercise 4) Exact Newton's method

(Exercise 5) BFGS (Broyden-Fletcher-Goldfarb-Shanno)

For the specific assignment and auxiliary Python code, we refer to `S4-GD-qNewton.ipynb`.

Exercise 6 (**Logistic regression).**

As a final, real-world example, consider the logistic regression function previously discussed in Exercise Session 1. This function is given by

$$f_{\log} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{log}(1 + \mathbf{exp} - b_i(x^\top a_i)) + \frac{\mu}{2N}||x||_2^2$$

Let $\sigma_i := \frac{1}{1+\mathbf{exp} - b_i(x^\top a_i)}$ and define

$$A := \begin{bmatrix} a_1^\top \\ \vdots \\ a_N^\top \end{bmatrix}, \quad s := \begin{bmatrix} b_1(\sigma_1 - 1) \\ \vdots \\ b_N(\sigma_n - 1) \end{bmatrix} \quad \text{and} \quad \Sigma := \begin{bmatrix} \sigma_1(1-\sigma_1) & & \\ & \ddots & \\ & & \sigma_N(1-\sigma_N) \end{bmatrix}.$$

Then, as shown in Exercise Session 1 the gradient is given by

$$\nabla f_{log} = \frac{1}{N} A^\top s + \frac{\mu}{N} x$$

and the Hessian by

$$\nabla^2 f_{log} = \frac{1}{N} A^\top \Sigma A + \frac{\mu}{N} I_n.$$

An implementation of these functions has already been provided in `S4-GD-qNewton.ipynb`. Two datasets will be considered, namely the a1a dataset, where we aim to predict whether a person makes $50000 a year based on social features (age, education, gender...), and the colon-cancer dataset.

Test the gradient descent method with and without linesearch on the logistic regression problem. Test both the *a1a* and the *colon-cancer* datasets and for regularizer μ equal to 100, 1 and 0.01. What do you observe when you decrease the regularizer? What is the relation with the condition number of the problem? Why is the performance of Newton's method and BFGS significantly worse for the *colon-cancer* dataset in terms of computation time? *hint: Look at the number of datapoints and features of each dataset.*