Optimization Algorithms
Exercise 3

Marc Toussaint
Learning & Intelligent Systems Lab, TU Berlin
Marchstr. 23, 10587 Berlin, Germany

Winter 2020/21

- During the tutorials the tutor will ask students randomly to present their solutions or questions. What you present does not have to be a perfect solution, but please try to prepare something.
- You will need to present your solutions via zoom by sharing the screen, e.g. by running code online, showing a picture of a hand-written solution, or a LaTeX'ed solution. Please be prepared for this.

1 Recap: Newton & Gauss-Newton basics

a) Consider a quadratic function $f(x) = \frac{1}{2}x^\top Ax + b^\top x + c$ with $A \in \mathbb{R}^{n\times n}$ symmetric and positive definite, $b \in \mathbb{R}^n$, $c \in \mathbb{R}$. What is the minimum of that function?

b) Write down an explicit equation for the Newton iterates, i.e. find $x_k = \dots$ for the k-th iterate which only depends on A, b, c and x_0. Assume a fixed, constant step size $\alpha \in \mathbb{R}$. When does it converge? How fast does it converge?

c) Let $f(x) = \|\phi(x)\|^2$ be a sum-of-squares cost for the features $\phi : \mathbb{R}^n \to \mathbb{R}^d$. Derive the Gauss-Newton method from a linear approximation (first order Taylor) of ϕ.

2 Getting started with NLopt

a) Follow the NLopt python tutorial at https://nlopt.readthedocs.io/en/latest/NLopt_Tutorial/#example-in-pytho

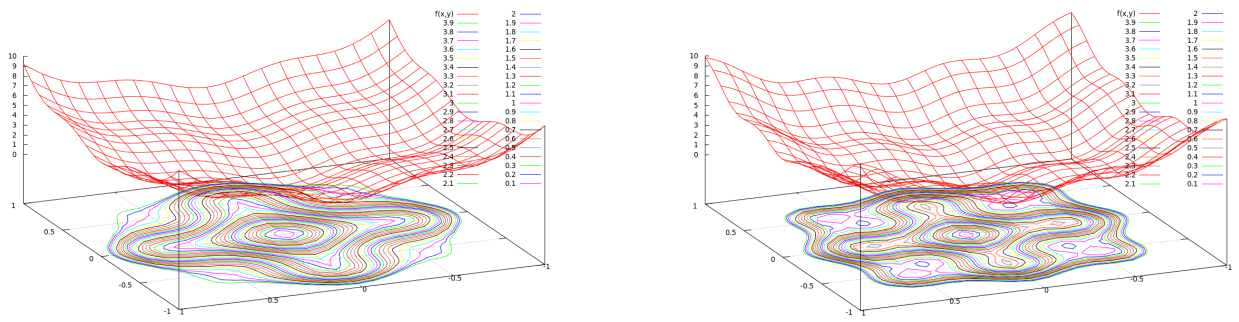
Modify this to record the optimization path (which was called `trace_xy` in the first exercise). Storing the optimization path could be done within the optimization method, or in the problem implementation – I highly recommend to do it within the problem implementation: Whenever a method queries the problem, store the query (x and $f(x)$). Use your function+path plotting method to display the problem cost function and optimization path.

b) Use the same NLopt method to minimize our two functions

$$f_{\text{sq}}(x) = x^\top Cx \ , \tag{1}$$

$$f_{\text{hole}}(x) = \frac{x^\top Cx}{a^2 + x^\top Cx} \ . \tag{2}$$

3 Gauss-Newton example



In $x \in \mathbb{R}^2$ consider the function

$$f(x) = \phi(x)^\top \phi(x) \ , \quad \phi(x) = \begin{pmatrix} \sin(ax_1) \\ \sin(acx_2) \\ 2x_1 \\ 2cx_2 \end{pmatrix}$$

The function is plotted above for $a = 4$ (left) and $a = 5$ (right, having local minima), and conditioning $c = 1$. The function is non-convex.

a) Extend your Newton method to use the Gauss-Newton approximation of the Hessian to solve the unconstrained minimization problem $\min_x f(x)$ for a random start point in $x \in [-1, 1]^2$. Compare the algorithm for $a = 4$ and $a = 5$ and conditioning $c = 3$ with gradient descent.

b) Optimize the function using NLopt, e.g. using L-BFGS https://nlopt.readthedocs.io/en/latest/NLopt_Algorithms/#low-storage-bfgs

c) Test your Gauss-Newton method on the robotics problem interfaces as described in the next exercise.

4 Voluntary: Python interfaces to large-scale problems

We uploaded a python module for you to interface. This python module implements benchmarks problems on which you can test your own solvers.

So far we only tested the direct installation in Ubuntu 18.04 – in the course of the next week we will also provide a docker and check options for Windows.

Please see the following notebook: https://github.com/MarcToussaint/optimization-course/blob/main/tutorials/how_to_query_an_NLP.ipynb

It implements a simple NLP which has methods to evaluate a new query x. This evaluation returns a feature vector $\phi(x)$, together with its Jacobian $J(x) = \frac{\partial}{\partial x}\phi(x)$, which directly defines the sum-of-squares problems $f(x) = \phi(x)^\top \phi(x)$.

Ubuntu users, try to install the module following the README.md, and test your Gauss-Newton solver.