Applied Optimization
Exercise 0 - Programming environment setup

Heng Liu Nicolas Gallego

Handout date: 19.09.2019

Submission deadline: 25.09.2019, 23:59 h

Hand-in instructions:

Please hand-in **only one** compressed file named after the following convention:

`Exercise`*n*`-GroupMemberNames.zip`

where n is the number of the current exercise sheet.

This file should contain:

- **Only** the files you changed (headers and source). It is up to you to make sure that all files that you have changed are in the zip.
- A `readme.txt` file containing a description on how you solved each exercise (use the same numbers and titles) and the encountered problems.
- Other files that are required by your `readme.txt` file. For example, if you mention some screenshot images in `readme.txt`, these images need to be submitted too.
- Submit your solutions to ILIAS before the submission deadline.

Implementing objective functions (40 pts)

First you need to download `aopt19-exercise0.zip` and extract into a folder of your choice. Follow the instructions in the exercise slides and compile the project.
In optimization terminology, we denote the function we want to minimize or maximize an objective function. Below are several functions that you need to implement in the code.

Quadratic function: $f(x) = \frac{1}{2}x^T Ax + bx + c$, with $A \in \mathbb{R}^{n\times n}$ and $b, x \in \mathbb{R}^n$.

2D Let $n = 2$, such that our variable is now a 2D vector in $x = (x_1, x_2)^T$. In this case, implement the 2D quadratic function:

$$f(x) = \frac{1}{2}(x_1^2 + \gamma x_2^2)$$

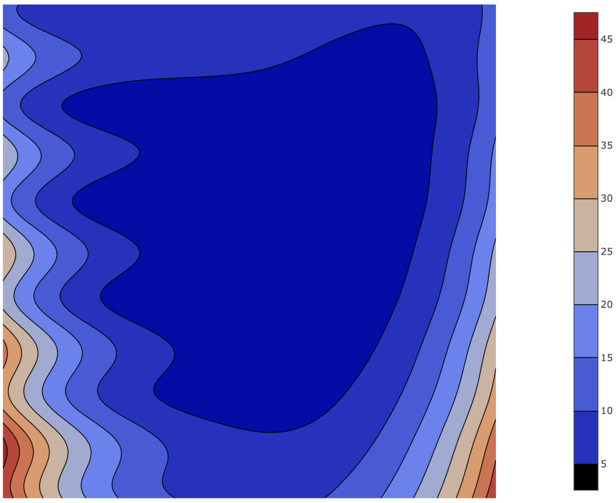

Figure 1: non convex function visualized as height map with isocontours

in the `f(...)` function in `FunctionQuadratic2D.hh`. By default, γ is set to -1 in the code. Evaluate it on reasonable domain and generate a .csv file to plot. The tool to export the .csv file is provided. Once the project is compiled, one can find a command line tool called "csv_exporter" in the build folder. Run it to see the usage. Afterwards, one needs to visualize the data on the website and submit a screenshot of it.

ND In applied optimization, the problem is often in higher dimension. Implement the n-dimension quadratic function given above in the `f(...)` function in `FunctionQuadraticND.hh`. The matrix A and vector b,c are initialized that one can use directly. For this function, there is no need to export data and visualize.

A non-convex function: Consider the following expression with variables x,y in $\mathbb{R}$.

$$h(x,y) = (y - x^2)^2 + cos^2(4 * y) * (1 - x)^2 + x^2 + y^2$$

implement it and evaluate it for visualization. Figure 1 depicts a visualization for comparison in the ranges $x \in [-2,2], y \in [-2,2]$ on a grid of 100×100.

Grid Search (60 pts)

Grid search refers to an exhaustive search for an optimum (minimum or maximum value) function value. One can sample the variables as a grid (here we use uniform grid for simplicity) over a certain domain and evaluate the function values of all the grid points. Compare the function values and return the optimum in the end. One needs to implement grid search algorithms for the functions above that output the minimum function value among the grid points and the corresponding variable value. Specifically, the functions `grid_search_2d(...)` and `grid_search_nd(...)` need to be implemented in the `GridSearch.hh` file. The `grid_search_nd(...)` should work for any specified n input from the command line. Run the command line tool ″grid_search″ to see the usage. Experiment with different functions and different parameters. See how inefficient the method is for high dimension optimization.