C6.2/B2. Continuous Optimization

Numerical Laboratory 1 (optional)

The first numerical lab will take place on Monday, February 10th (Week 4), 16.00–18.00 (may finish sooner) in L1; it is entirely optional and there is no hand-in or marks. Please bring your laptop to the lab, with Matlab on it if possible. The lab sessions will be held in Matlab (but you may use other programming languages if you prefer).

The tasks for the first lab are given below. We encourage you to make progress on them before the lab, if possible, so that you can make the most of the lab (and bring your workings with you to the lab); but preliminary code will be provided to start you on the tasks if you have not managed to do so.

(**Algorithm implementation**)

1. Implement Steepest Descent Method with Backtracking Armijo linesearch as described in the lectures. You may want to start by implementing the linesearch routine first, at a given point x and for a given descent direction s.

   Please write your code for a general, not a specific, objective. You will need to carefully consider how to provide an(y) objective and its derivatives to your algorithm/code. You may want to use separate files for the objective values and gradients (and second derivatives) and explicitly code their expressions (for each of the test functions you will use).

2. Using the same linesearch technique, modify your steepest descent code so that it uses the Newton direction (instead of the steepest descent direction) on each iteration.

   Add the following feature: at each iteration, check the positive definiteness or otherwise of the second-derivative matrix (for example, by calculating its left-most eigenvalue). If this matrix is not positive definite, then consider modifying the Hessian in such a way as to make it positive definite and use this modified Hessian to calculate the search direction. You may want to use the techniques 1 or 2 on the 'Modified Newton's method' slides of Lecture 5.

(**Test functions**)

(a) Simple test functions and debugging: to check your codes are working, test them on very simple functions such as $f(x) = x^2$, $x \in \mathbb{R}$; and $f(x_1, x_2) = x_1^2 + x_2^2$, $(x_1, x_2) \in \mathbb{R}^2$, for which you know the solutions.

(b) Recalling questions on Problem Sheet 1 and Problem Sheet 2, test your code on the following functions

$$f(x_1, x_2) = \frac{1}{2}(ax_1^2 + x_2^2),$$

where $a > 0$ and Rosenbrock's function,

$$f(x_1, x_2) = 10(x_2 - x_1^2)^2 + (1 - x_1)^2.$$

In your tests, you may want to assess how the number of iterations varies with: the choice of starting point, the choice of accuracy tolerance in the termination condition, the problem scaling. Assess the local rate of convergence for both steepest descent and Newton's method with linesearch.

(c) (advanced) For other low-dimensional standard optimization test functions, please see for example, `https://en.wikipedia.org/wiki/Test_functions_for_optimization`. You may want to apply your codes to some of these functions.