MS-E2122 - Nonlinear optimization Project Work 1
Prof. Fabricio Oliveira **DEADLINE: Friday 05.11.2021 23:55**

Please submit your answers to Mycourses before the given deadline as a separate files (.pdf file with the report and Jupyter notebook .ipynb files if there are any). **Do not compress them into a single (zip or equivalent) archive**.

Introduction

In this project, you will learn how to implement and compare different methods to solve unconstrained optimisation problems. In addition, you will learn how to use performance profiles to compare (or *benchmark*) different optimisation methods when applied to a collection of benchmark instances. You are expected to implement some of the unconstrained optimisation algorithms introduced in the course and analyse their performances using the information presented in the lectures and exercise sessions so far.

You are asked to solve four problems. The first three problems consist of 2-dimensional functions. In the last problem, you will solve 2 sets of 100 random instances of a large-scale quadratic problem. You are expected to implement two variants of four optimisation methods: each method uses two distinct line searches. In addition, you must write a report with a detailed analysis on the performances of these optimisation methods when solving the four problems.

Learning objectives

1. Learn by practice some of the algorithms for unconstrained optimisation to obtain a deeper understanding of the methods and challenges concerning their implementation;
2. Learn how to perform a structured comparison of different algorithms using performance profiles;
3. Learn how to report numerical results that are scientifically sound and reproducible.

General requirements

1. Students can work individually or in pairs to complete this project. Groups of three or more individuals will not be accepted.
2. Students are expected to **submit a report** (see details in the "Report format" section) and **the notebook skeleton filled with your code** used for the project. It will be tested for correctness and, therefore, is part of the assessment. The notebook can be downloaded here.
3. **Rename your notebook according to your student number before submission. For example, if your student number is 112233, rename the notebook as 112233.ipynb**
4. **Deadline** for submitting the project report and the notebook **Friday 05.11.2021**. Late submissions will not be accepted.

Specific project requirements

The project consists of four tasks, described below.

(1) Consider the function

$$f_1(x_1, x_2) = 2(0.5x_1^2 + 4x_2^2) - 0.5x_1x_2$$

with starting point $(x_1, x_2) = (-10, 10)$. You are required to find the optimal solution of f_1, if possible, using the following algorithms:

(a) Gradient method;

(b) Gradient method with momentum ('heavy-ball' method).

The gradient method with momentum (heavy-ball method) is a variant of the gradient method in which the direction d_k at iteration k is obtained by means of a convex combination between the the gradient $\nabla f(x_k)$ of f at x_k and the previous direction d_{k-1}. By doing so, the step at iteration k takes into account the gradient at two distinct points, thus considering approximate the curvature information to mitigate the often observed zig-zagging of pure gradient methods.

The method can be summarised as follows. Further reference about this and other variants of the gradient method with extrapolation can be found at *D. Bertsekas, Nonlinear programming, 2016, Athena Publ. pg 80* and in several other references.

Algorithm 1 Gradient method with momentum (heavy ball)

initialise. tolerance $\epsilon > 0$, weight $w \in [0, 1)$, initial point x_0, iteration count $k = 0$.
while $||\nabla f(x_k)|| > \epsilon$ **do**
  $d_k = (1 - w)\left(-\frac{\nabla f(x_k)}{||\nabla f(\overline{x})||}\right) + wd_{k-1}$
  $\overline{\lambda}_k = \arg\min_{\lambda \in \mathbf{R}} \{f(x_k + \lambda d_k)\}$
  $x_{k+1} = x_k + \overline{\lambda}_k d_k$
  $k = k + 1$
end while
return x_k.

You are required to implement versions of the above methods with exact and inexact line search methods. The exact line search must be the bisection method with the initial interval $\lambda \in [0, 10]$. The inexact line search must be Armijo's rule with $\alpha = 0.01$ and $\beta = 0.7$ (all of those are set as standard values in the function headers of the skeleton provided).

You are expected to provide the rationale for parametrising the 'heavy-ball' method weight $w \in [0, 1)$. You are also expected to perform comparison between the four variants of the two methods and analyse their convergence (i.e., whether they converged to an optimal point, if the point of convergence is a local or global optimum, and the number of iterations and average time taken (in total and per iteration) for convergence). We recommend that you use tables and level curve plots to support your analyses of the convergence behaviour exhibited by the methods you implemented.

(2) Consider the function

$$f_2(x_1, x_2) = e^{x_1 + 2x_2 - 0.1} + e^{x_1 - 2x_2 - 0.1} + e^{-x_1 - 0.2}$$

with the starting point $(x_1, x_2) = (-2.5, -3.5)$. You are required to find the optimal solution of f_2, if possible, using the following algorithms:

(a) Newton's method;

(b) BFGS method.

You must implement the 4 variants (2 above methods using exact and inexact line search) and compare their performance. You must use the same standard parametrisation from task (1)

for the line searches. Like in item (1), you are expected to provide a performance comparison and a convergence analysis of all methods.

(3) Employ the exact variants (i.e., the versions using exact line search) of the four methods from items (1) and (2) to find the optimal for the following function:

$$f_3(x_1, x_2) = (x_1^2 + x_2 - 10)^2 + (x_1 + x_2^2 - 15)^2$$

with the starting point $(x_1, x_2) = (-0.5, -2)$. As in the previous items, you are expected to provide a performance comparison and convergence analysis for all 4 methods.

(4) You are required to apply the 8 variants of the 4 methods to the quadratic function

$$f(x) = (1/2)x^\top Ax - b^\top x \text{ with } x \in \mathbf{R}^{100}.$$

You must apply the 8 variants to two sets of 100 distinct problem instances with randomly generated A and b (where A is constructed to be positive definite). The sets will differ with respect to the condition numbers of the matrices A in the function. In these experiments, you are expected to assess and compare the performance of the algorithms using performance profiles. Skeleton codes for generating the random instances and plotting the performance profiles will be provided.

For tasks (1), (2), and (3), the report must include a critical assessment of the performance of each algorithm for a maximum of 1000 iterations (i.e., if the solution obtained is optimal, number of iterations taken, time per iteration, and total time).

For the task (4), the report must include performance profiles comparing the 8 algorithm variants. Information concerning how to interpret performance profiles can be found on this link: https://link.springer.com/article/10.1007/s101070100263.

Report format

A Latex template for the report is provided and can be downloaded here. Font sizes, margins, and other layout settings will be set for the template and are not to be tampered with.

The report must be 10 pages at maximum, including figures. References are not necessary, but if used, they are not included in the page limit. Reports not complying with these guidelines will have their grades penalised.

The report must consist of the following sections:

1. **Project description**

   Describe the project setting, such as the computational platform used (operating system, processor type and clock speed, memory, if own computer was used; mention CS Jupyterlab server otherwise), the algorithmic variants implemented and tested. All details that allow replicating the numerical experiments must be described in this section.

2. **Numerical results**

   Create a subsection for each of the tasks (1) - (4). For the 2-dimensional functions (1), (2), and (3), the algorithms must be compared in terms of their convergence, iteration count, and solution time. You are expected to give reasoning for the observed performance based on the information presented in this course. A structured comparison between required methods in each part (1), (2), and (3) is expected for the 2-dimensional functions. Provide evidence making use of plots and tables to support your arguments.

   For the set of problems (4), present the performance profiles for the two different condition number settings, and provide a detailed comparison between the algorithms based on these profiles. Discuss also how the two different line searches and condition number settings affect the performance of the algorithms.

3. **Discussion and conclusions**

   Discuss the main aspects that affect the behaviour of each algorithm, considering the functions they have been applied, connecting your observations to concepts presented in the course. For example, you can provide an analyses whether aspects such the nature (e.g., if polynomial or exponential), convexity, or conditioning of the functions have any influence on the observed performance of the method.

Assessment

The report will be assessed according to the following criteria:

(1) Correctness of the algorithms - if the implementations are all correct.

(2) Technical quality of analysis - adequate statements and precise analyses, supported by experimental evidence and theoretical knowledge.

(3) Format, clarity, and presentation - tables and plots that are pristine, easily readable, and support the arguments made; use of precise technical language.

Each criterion (1), (2), and (3) will be graded between 1 and 5 and the final grade will be the rounded average of the criteria grades.