Numerical Optimization Lab 02: Steepest Descent

Francesco Della Santa

Abstract

In this lesson, we will learn to implement the *steepest descent* optimization method.

1 Introduction

Create a working folder where you will save all the files for this and the future Numerical Optimization laboratories; then, download from the web page of the course the file *test_functions.mat*. Save the file in the working folder previously created for the laboratories.

2 Exercises

Exercise 1 (Steepest Descent Implementation)**.** Write a Matlab function *steepest_descent.m* that implement the *steepest descent* optimization method (see Appendix A) and that takes the following inputs and outputs.

INPUTS:

- `x0`: a *column vector* of n elements representing the starting point for the optimization method;
- `f`: a *function handle* variable that, for each column vector $\boldsymbol{x} \in \mathbb{R}^n$, returns the value $f(\boldsymbol{x})$, where $f : \mathbb{R}^n \to \mathbb{R}$ is the *loss function* that have to be minimized;
- `gradf`: a *function handle* variable that, for each column vector $\boldsymbol{x} \in \mathbb{R}^n$, returns the value $\nabla f(\boldsymbol{x})$ as a *column* vector, where $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$ is the *gradient* of f;
- `alpha`: a *real scalar value* characterizing the step length of the optimization method;
- `kmax`: an *integer scalar value* characterizing the maximum number of iterations of the method;
- `tolgrad`: a *real scalar value* characterizing the tolerance with respect to the norm of the gradient in order to stop the method.

OUTPUTS:

- `xk`: the last vector $\boldsymbol{x}_k \in \mathbb{R}^n$ computed by the optimization method before it stops;
- `fk`: the value $f(\boldsymbol{x}_k)$;
- `gradfk_norm`: the euclidean norm of $\nabla f(\boldsymbol{x}_k)$;
- `k`: index value of the last step executed by the optimization method before stopping;
- `xseq`: a matrix/vector in $\mathbb{R}^{n \times k}$ such that each column j is the j-th vector $\boldsymbol{x}_j \in \mathbb{R}^n$ generated by the iterations of the method.

Once you have written the function, test it using the data inside the file *test_functions.mat* and plot together:

- the loss f (given in *test_functions.mat*) using the Matlab function `contour`;
- the sequence `xseq`.

A Steepest Descent

Let the function $f : \mathbb{R}^n \to \mathbb{R}$ be given. The steepest descent method is an iterative optimization method that, starting from a given vector $\boldsymbol{x}_0 \in \mathbb{R}^n$, computes a sequence of vectors $\{\boldsymbol{x}_k\}_{k\in\mathbb{N}}$ characterized by

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha \boldsymbol{p}_k \,, \quad \forall\, k \geq 0 \,, \tag{1}$$

where the descent direction $\boldsymbol{p}_k$ is the steepest one, i.e. $\boldsymbol{p}_k = -\nabla f(\boldsymbol{x}_k)$, and the step length factor $\alpha \in \mathbb{R}$ is arbitrarily chosen.