Numerical Optimization Lab 10: Constrained Steepest Descent

Francesco Della Santa

Abstract

In this lesson, we implement the *Projected Gradient* method for *Constrained optimization problems* with respect to the *Steepest Descent* method.

1 Exercises

Exercise 1 (Sphere Projection)**.** Write a Matlab function *sphere_projection.m* that, for each $\boldsymbol{x} \in \mathbb{R}^n$ and any n-dimensional sphere $X = B_r(\boldsymbol{c})$, returns $\Pi_X(\boldsymbol{x})$ as defined in (3). In particular the inputs and outputs of the function are the following.

INPUTS:

`x`: the input column vector $\boldsymbol{x} \in \mathbb{R}^n$;

`c`: the column vector $\boldsymbol{c} \in \mathbb{R}^n$ representing the center of the sphere;

`r`: the radius $r > 0$ of the sphere.

OUTPUTS:

`xhat`: the value $\Pi_X(\boldsymbol{x})$.

Exercise 2 (Box Projection)**.** Write a Matlab function *box_projection.m* that, for each $\boldsymbol{x} \in \mathbb{R}^n$ and any n-dimensional "box" $X = [m_1, M_1] \times \cdots \times [m_n, M_n]$, returns $\Pi_X(\boldsymbol{x})$ as defined in (4). In particular the inputs and outputs of the function are the following.

INPUTS:

`x`: the input column vector $\boldsymbol{x} \in \mathbb{R}^n$;

`mins`: n-dimensional column vector $[m_1, \ldots, m_n]^\top$;

`maxs`: n-dimensional column vector $[M_1, \ldots, M_n]^\top$.

OUTPUTS:

`xhat`: the value $\Pi_X(\boldsymbol{x})$.

Exercise 3 (Steepest Descent with Projected Gradient Implementation)**.** Write a Matlab function *constr_steepest_desc_bcktrck.m* that implement the *constrained steepest descent* optimization method with *projected gradient* (see Appendix A); the function implements the case with fixed $\gamma_k \equiv \gamma > 0$ and the backtracking strategy with respect to α_k (see item 1 of Appendix A.1).

In particular, start from the code of the function the function *steepest_desc_bcktrck.m* and modify it, introducing/removing the following inputs and outputs:

Removed INPUTS:

`alpha0`**:** We do not need to specify $\alpha_k^{(0)}$ since it is equal to 1.

Additional INPUTS:

`gamma`**:** fixed factor $\gamma > 0$ that multiplies the descent direction befor the (possible) projection on ∂X, where X is the optimization domain;

`tolx`**:** a *real scalar value* characterizing the tolerance with respect to the norm of $\parallel \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \parallel$ to stop the method;

`Pi_X`**:** a *function handle* variable that characterizes the function $\mathit{\Pi}_X$ with respect to an arbitrary set $X \subset \mathbb{R}^n$. For each column vector $\boldsymbol{x} \in \mathbb{R}^n$, Π_X returns $\boldsymbol{x}$ if $\boldsymbol{x} \in X$, and its projection on ∂X otherwise. The function handle must take only $\boldsymbol{x}$ as input argument.

Additional OUTPUTS:

`deltaxk_norm`**:** the euclidean norm $\parallel \boldsymbol{x}_k - \boldsymbol{x}_{k-1} \parallel$ (where k is intended to be the last iteration).

Once you have written the function, test it using the data in *test_functions2.mat*. Concerning the other paramters: `c1` $= 10^{-4}$, `rho` $= 0.8$, `btmax` $= 50$, $\gamma = 0.1$, `tolx`$= 10^{-12}$, sphere center $\boldsymbol{c} = [-3, -3]^\top$, sphere radius $r = 1.5$, box domain $[-4.5, -1.5] \times [-4.5, -1.5]$.
Then, plot:

- the loss f_i (given in *test_functions2.mat*), $i = 1, 2, 3$, using the Matlab function `contour`, the sequence `xseq` and the boundary domain;
- the loss f_i (given in *test_functions2.mat*), $i = 1, 2, 3$, using the Matlab function `surf`, the sequence `xseq` and the boundary domain *projected on the surface defined by the function.*
- the barplot of the values in `btseq` using the function `bar`, with respect to the functions f_i.

A Steepest Descent and Projected Gradient

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a loss function that have to be minimized with respect to a set $X \subset \mathbb{R}^n$ of constrains, i.e. we want to find

$$\arg\min_{\boldsymbol{x}\in X} f(\boldsymbol{x})\,.$$

The *Projected Gradient* method applied to the steepest descent is characterized by the following main steps (at the k-th iteration):

1. compute the steepest descent direction $\boldsymbol{p}_k = -\nabla f(\boldsymbol{x}_k)$;
2. given an arbitrary sequence of $\gamma_k > 0$, we check if $\bar{\boldsymbol{x}}_k = (\boldsymbol{x}_k + \gamma_k \boldsymbol{p}_k)$ is in X or not; in particular, we compute

$$\widehat{\boldsymbol{x}}_k = \mathit{\Pi}_X(\bar{\boldsymbol{x}}_k) := \begin{cases} \bar{\boldsymbol{x}}_k\,, & \text{if } \bar{\boldsymbol{x}}_k \in X \\ \Pi_X(\bar{\boldsymbol{x}}_k)\,, & \text{otherwise} \end{cases}\,, \tag{1}$$

   where $\Pi_X(\bar{\boldsymbol{x}}_k)$ is the projection of $\bar{\boldsymbol{x}}_k$ on the boundary ∂X of X.

3. compute the *feasible* direction $\boldsymbol{\pi}_k := (\widehat{\boldsymbol{x}}_k - \boldsymbol{x}_k)$;
4. compute the next step of the optimization method: $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \alpha_k \boldsymbol{\pi}_k$, where $\alpha_k \in [0,1]$ is the sequence of step length factors.

Attention: Since the minimum not necessarily corresponds to a point with null gradient, a second stopping criterium with respect to the incrementation step $\| \boldsymbol{x}_{k+1} - \boldsymbol{x}_k \|$ is necessary.

A.1 Linea Search and values of γ_k and α_k

In a nutshell, two main line search methods exist for the Projected Gradient method:

1. **Line Search along the feasible direction $\boldsymbol{\pi}_k$:** we assume a constant sequence $\gamma_k \equiv \gamma > 0$, for each $k \in \mathbb{N}$, and we perform a line search method with respect to α_k along the direction $\boldsymbol{\pi}_k$ (starting from the value $\alpha_k^{(0)} = 1$).
2. **Linea Search along the projection arc:** we assume the sequence α_k constantly equal to 1; then, from (1), we have that

$$\begin{aligned} \boldsymbol{x}_{k+1} = \widehat{\boldsymbol{x}}_k &= \mathit{\Pi}_X\left(\boldsymbol{x}_k + \gamma_k \boldsymbol{p}_k\right) = \\ &= \mathit{\Pi}_X\left(\boldsymbol{x}_k - \gamma_k \nabla f(\boldsymbol{x}_k)\right)\,, \end{aligned} \tag{2}$$

   and we can perform a linea search method with respect to the parameter γ_k along the projection arc.

B Projection Functions

Here, some example of function $\mathit{\Pi}_X$ with X as a sphere or a box in $\mathbb{R}^n$.

B.1 n-dimesnional Sphere

Let $\boldsymbol{c} = [c_1, \ldots, c_n]^\top$ be the center of the sphere X and let $r \in \mathbb{R}^+$ be its radius. Then, it holds

$$\mathit{\Pi}_X(\boldsymbol{x}) = \begin{cases} \boldsymbol{x}\,, & \text{if } \| \boldsymbol{x} - \boldsymbol{c} \| \leq r \\ \boldsymbol{c} + r\frac{\boldsymbol{x}-\boldsymbol{c}}{\|\boldsymbol{x}-\boldsymbol{c}\|}\,, & \text{otherwise} \end{cases}\,. \tag{3}$$

B.2 n-dimesnional Box

Let X be the n-dimensional box $[m_1, M_1] \times \cdots \times [m_n, M_n] \subset \mathbb{R}^n$. Then, it holds

$$\mathit{\Pi}_X(\boldsymbol{x}) = \left[\mathit{\Pi}_{[m_1,M_1]}(x_1), \ldots, \mathit{\Pi}_{[m_n,M_n]}(x_n)\right]^\top , \tag{4}$$

where

$$\mathit{\Pi}_{[m_i,M_1]}(x_i) = \begin{cases} m_i\,, & \text{if } x_i < m_i \\ M_i\,, & \text{if } x_i > M_i \\ x_i\,, & \text{otherwise} \end{cases} . \tag{5}$$