# 7 Numerical Methods

I encounter in this paper different optimization problems that do not have closed forms: We have already seen that the lasso regression, the elastic net regression, the group lasso and the Huber loss function ( and its combination with other penalization method) do not have closed form solution. In this paper, I propose to solve all of the following methods using an Accelerated Proximal Gradient Descent method.

**Accelerated Proximal Gradient Descent method.** In this paper, I use the accelerated proximal gradient descent algorithm in order to solve the various proposed regularized and non regularized convex loss functions. This algorithm combines two different generic convex optimization methods, namely the accelerated gradient descent and the proximal gradient descent methods.

The proximal gradient descent is a method to optimize convex non smooth functions. There are different other methods that also minimize non smooth functions, but what makes proximal method interesting is its speed. Hence, while the ubiquitous sub gradient method[64], for instance, converges at a rate of $o(\frac{1}{\epsilon^2})$, the accelerated gradient descent has a speed of $o(\frac{1}{\epsilon})$ ( It converges as rapidly as the vanilla gradient descent). This optimization method relies principally on two pillars:

The first is the Moreau Proximal Point Algorithm (PPA) typically applied in the optimization of non-smooth functions. Formally, for some $\text{Min}_x f(x)$ problem, given a non-smooth function $f(.)$, the PPA algorithm is defined as such: $x^{t+1} = \text{prox}_{\gamma f}(x^t) = \text{Argmin}_y \gamma f(y) + \frac{1}{2}\|x^t - y\|_2^2$.[65] That is, the PPA defines some simple convex and differentiable function of $y$, $\gamma f(y) + \frac{1}{2}\|x^t - y\|_2^2$, which is tangent to $f(x)$ at $x^{(t)}$; such that its minimum is easily derivable. At each iteration, the new $x^{(t+1)}$ is the minimizer of the tangent function at $x^{(t)}$ , and this repeats until some convergence criterion is met. The *prox* operator simply refers to this minimization subproblem defined by this algorithm.

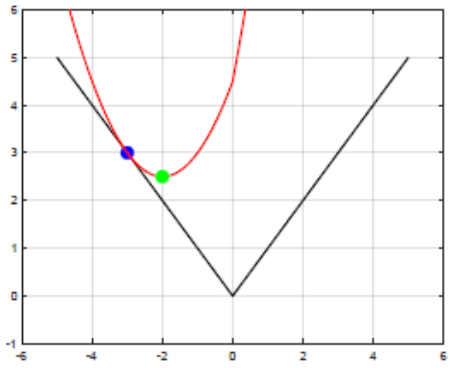


Figure 10: Illustration of Moreau Proximal Point Algorithm on some function $f(x) = |x|$. The proximal operator defines the red tangent simple smooth convex function and finds its minimum

[64]The subgradient method solves the problem of non differentiability of some objective function $f(.)$ by computing a subgradient of $f(.)$ at $x^t$, $g^t \in \partial f(x^t)$ and computing $x^{t+1} = x^t - \alpha^t g^{(t)}$ at each iteration t

[65]Here, the "y" is used to represent any variable.

Above, is an illustration of the Proximal Point Method: for $f(x) = |x|$, a non differential function. This is a representation of a single iteration of the algorithm where $x^{(t+1)}$ is the minimizer of a differentiable function defined by the proximal operator.

The second pillar of the proximal gradient descent method lies in the equivalence between the gradient descent algorithm, expressed as $x^{(t+1)} = x^{(t)} - \alpha \nabla f\left(x^{(t)}\right)$, and the minimization of the Taylor approximation of $f(.)$ around$x^{(t)}$ considering $\nabla^2 f(x) = \frac{1}{\alpha} I$.

In fact, this becomes clear when solving the *Argmin* of the Taylor expansion of $f(.)$ around $x^{(t)}$:

$$\underset{y}{\text{Argmin}} f\left(x^t\right) + \nabla f\left(x^{(t)}\right)^T \left(y - x^{(t)}\right) + \frac{1}{2\alpha} \left\| y - x^{(t)} \right\|_2^2$$

We find the minimum by equation the gradient of the objective function to zero $0 = \nabla f\left(x^{(t)} + \frac{1}{\alpha}\left(x^{(t+1)} - x^{(t)}\right)\right.$ $\implies x^{(t+1)} = x^{(t)} - \alpha \nabla f\left(x^{(t)}\right)$, which corresponds to the gradient descent method. Therefore, instead of employing the gradient descent algorithm, one can iteratively determine the minimum of the Taylor approximation of the objective function at a specific point until a convergence criterion is satisfied.

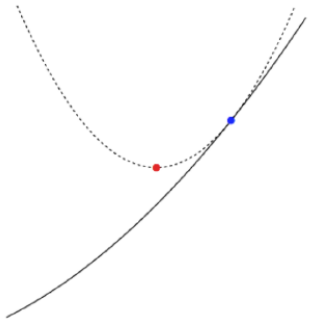

Figure 11: Graphical representation of the equivalence between Gradient descent and Taylor Approximation minimization. Instead of using Gradient Descent, one can minimize the Taylor approximation (Dotted curve) of the objected function (solid curve). This graph illustrates only one iteration.

To resume, the two pillars suggest: First, that non smooth functions can be optimized by using proximal operators at each iteration ; and Second that gradient descent on smooth differentiable functions is equivalent to minimizing the Taylor approximation of our function at each iteration.

Proximal Gradient Descent merges both methods: In fact,For some function $f(x) = g(x) + h(x)$ with $g(.)$ being convex and differentiable and $h(.)$ convex but non differentiable; Proximal gradient's approach involves iteratively minimizing "sub-problems" formed by the sum of the Taylor approximation of the convex and differentiable function $g(x)$ and the convex but non-differentiable function $h(x)$. Concretely; it consists of approaching the problem as if it were a gradient descent minimization

on some smooth function while keeping the non-smooth function untouched.

$$\begin{aligned} x^{(t+1)} &= \underset{y}{\operatorname{argmin}} \bar{g}_t(y) + h(y) \\ &= \underset{y}{\operatorname{argmin}} g(x^{(t)}) + \nabla g(x^{(t)})^T (y - x) + \frac{1}{2\alpha} \|y - x\|_2^2 + h(y) \\ &= \underset{y}{\operatorname{argmin}} \frac{1}{2\alpha} \|y - (x - \alpha \nabla g(x^{(t)})\|_2^2 + h(y) \end{aligned}$$

This is equivalent to the Proximal operator of $h(.)$ at $(x^{(t)} - \alpha \nabla g(x^{(t)}))$ with a proximal parameter $\alpha_{\mathrm{t}}$. Formally, the Proximal Gradient descent is defined as such:

First initialize $x^{(0)}$ , $x^{(t)} = \operatorname{prox}_{h,\alpha_{(t-1)}} \left( x^{(t-1)} - \alpha_{t-1} \nabla g \left( x^{(t-1)} \right) \right), \quad t = 1, 2, 3, \ldots$ , iterate until convergence

This hybrid optimization has a faster convergence rate than the standard proximal point method for non smooth functions (Tibshirani). Hence, whenever, the non-smooth objective function can be transformed into a composite optimization problem with smooth and non smooth components; it is preferable to use PGD than PPM for computational speed.

On the other hand, the accelerated proximal gradient descent, incorporates acceleration into the optimization. Introduced by Nesterov, the Accelerated Gradient Descent is a modification of the standard Gradient descent method designed to achieve a faster convergence rate. Gradient descent can exhibit very slow convergence depending on the shape of the objective function [66]; This is clearly the case when the convex objective function has a minimum in a "narrow valley" which causes the gradient descent to zigzag very slowly towards it [67]. In order to mitigate this problem, Nesterov incorporates memory into the Gradient descent method: For each iteration, the new direction incorporates the "momentum" of previous directions; this has the effect of tilting "degenerate" directions to "coherent" and "well behaved" ones. Formally, Nesterov is defined as such: For an unconstrained smooth and convex minimization problem: Initialize $y^{(0)}$, then compute $x^{(t)} = y^{(t-1)} - \alpha^{(t)} \nabla f \left( y^{(t-1)} \right)$ for $y^{(t)} = x^{(t)} + \frac{t-1}{t+2} \left( x^{(t)} - x^{(t-1)} \right)$ and iterate for $t = 1, 2, \ldots$ until convergence.[68] Ultimately, this method is faster than the Gradient descent method.

The Accelerated Proximal Gradient Descent Algorithm; simply incorporates the Nesterov Gradient descent to the Proximal Gradient descent. Formally, it is defined as such:

Initialize $x^{(0)}$ and $y^{(0)} = x^{(0)}$ ; then compute $x^{(t)} = \operatorname{prox}_{\alpha^{(t)} h} \left( y^{(t-1)} - \alpha^{(t)} \nabla g \left( y^{(t-1)} \right) \right)$ for $y^{(t)} = x^{(t)} + \frac{t-1}{t+2} \left( x^{(t)} - x^{(t-1)} \right)$ ; and iterate for $t = 1, 2, \ldots$ until convergence.

This optimization method is perfectly suited for our optimization problems as our objective func-

[66] And the set hyperparameters

[67] Or when a concave function has a maximum in a narrow space

[68] Geometrically, Nesterov Gradient descent simply extrapolates the previous direction by some "memorized" momentum ( depending on previous trajectories); then follows the negative gradient

tions can be decomposed as smooth and non smooth function; and , importantly each of the non-smooth functions used in this paper has its own closed form proximal operator. This fact is important, as Proximal Gradient Descent computes the proximal operator of $h(.)$ at each iteration. (See: Appendix for Closed form $prox(.)$ of each of the non-smooth functions).