Ilyes
HAMMOUDA

ENSAE 2ème année
Stage d'application
Année scolaire 2023-2024

Étude de l'optimisation à l'ordre zéro en grande dimension

ENSAE-CREST
France-Paris

Encadrants : Alexandre TSYBAKOV
Mohamed NDAOUD

Table des matières

1 Remerciements

Je souhaite tout d'abord exprimer ma profonde gratitude envers mes deux encadrants, Monsieur Alexandre Tsybakov et Monsieur Mohamed Ndaoud. Monsieur Tsybakov m'a généreusement prodigué son expertise à travers des explications détaillées sur les enjeux cruciaux du projet, tandis que Monsieur Ndaoud m'a accompagné de manière hebdomadaire tout au long du projet. Il a su m'orientant avec constance vers les meilleures voies à suivre. Leurs conseils éclairés, leur vaste savoir-faire, et leur disponibilité sans faille ont joué un rôle déterminant dans les progrès réalisés et les résultats obtenus au cours de ce stage enrichissant.

De plus, je tiens à exprimer ma reconnaissance envers Monsieur Nicolas Chopin, qui m'a suivi tout au long de ce stage d'application durant ma deuxième année à l'ENSAE. Sa compréhension et son soutien ont grandement contribué à rendre cette expérience formatrice et fructueuse.

2 Introduction

Au cours de ce stage d'application, dans le cadre de ma scolarité à l'ENSAE, j'ai travaillé sur le problème d'optimisation d'ordre zéro. En effet, la croissance de l'apprentissage automatique ainsi que les méthodes basées sur l'analyse des données au cours des dernières années ont été grandement soutenues par les progrès substantiels réalisés dans le domaine de l'optimisation. Ces techniques ont, quant à elles, bénéficié d'avancées considérables en termes d'acquisition de données : efficacité et flexibilité. Ces avancées englobent également l'exploitation des données, notamment dans des contextes et applications caractérisés par des ressources restreintes.

Par exemple, l'entraînement des modèles basés sur de vastes bases de données a particulièrement bénéficié du développement de méthodes d'optimisation stochastique. En effet, pour résoudre des problèmes d'optimisation en grande dimension, des méthodes d'optimisation stochastiques basées sur les informations de premier et de second ordre de la fonction objective ont été élaborées et appliquées à des cas concrets, comme on peut le trouver dans les études suivantes : [1], [2].

Il est vrai que les méthodes mentionnées ci-dessus offrent des taux de convergence et des performances intéressantes. Cependant, lorsque la forme explicite de la fonction à minimiser est inconnue ou lorsque le calcul du gradient ou de la hessienne est impossible ou coûteux, il est inévitable d'utiliser les méthodes d'optimisation d'ordre zéro.

Ainsi, les méthodes d'optimisation d'ordre zéro, ou optimisation sans gradient, sont utilisées dans les situations où la forme analytique de la fonction objective est inconnue ou lorsqu'il est inefficace d'utiliser les conditions de premier ou de second ordre. De plus, ces techniques s'appliquent dans divers domaines, tels que la médecine ([3]), la chimie ([4]), l'optimisation des réseaux cellulaires ([5]), la prédiction structurée, l'apprentissage par renforcement, l'optimisation d'algorithmes spécifiques tels que les algorithmes de bandit, et l'ajustement des hyperparamètres.

Plus récemment, ces méthodes ont trouvé des applications dans le domaine de l'apprentissage automatique contradictoire (Adversarial Machine Learning), notamment pour les attaques adverses dirigées contre les réseaux neuronaux ([6], [7]).

De manière générale, les algorithmes d'optimisation d'ordre zéro, également appelés algorithmes d'optimisation sans gradient, visent à optimiser des fonctions de type boîte noire [1], ainsi que des fonctions pour lesquelles le calcul des gradients est impossible ou coûteux, tout en minimisant le budget, c'est-à-dire le nombre d'évaluations de la fonction

1. Une fonction boîte noire est une fonction dont on connaît seulement les entrées et les sorties, mais dont la structure interne et la logique sont inconnues. Les sorties d'une telle fonction peuvent éventuellement contenir un certain bruit.

objective auxquelles on peut avoir accès.

Dans le cadre de ce stage, nous nous sommes interrogés sur la possibilité de développer une méthode adaptative pour résoudre ce problème d'optimisation. En d'autres termes, nous cherchions une méthode capable d'obtenir des résultats similaires à ceux de l'état de l'art, tout en nécessitant moins d'informations sur la structure des données. Plus précisément, nous souhaitions résoudre ce problème en l'absence d'informations précises sur la sparsité de la fonction objective à minimiser.

Pour répondre à cette problématique, nous avons tout d'abord entrepris une revue de la littérature récente portant sur le problème d'optimisation à l'ordre zéro. Dans le cadre de cette étude, nous avons effectué une analyse théorique de deux algorithmes en particulier : la Sélection de Composants Successifs [8] et la Méthode d'Optimisation Régularisée d'Ordre Zéro (ZORO) [9]. De plus, nous avons mis en place des implémentations de ces algorithmes afin d'étudier leurs comportements empiriques. Dans un deuxième temps, nous avons cherché à proposer une méthode adaptative et à étudier ses caractéristiques et son comportement empirique.

2.1 Notation

Dans ce rapport, nous adopterons les mêmes notations utilisées dans la plupart des papiers scientifiques abordant ce problème.

Pour un entier naturel $n \in \mathbb{N}$, nous définissons $[n] := 1, 2, \ldots, n$. La norme zéro, notée $\|\cdot\|_0$, est l'opérateur qui compte le nombre d'éléments non nuls du vecteur ou de la matrice. Nous noterons 's' la sparsité d'une matrice, c'est-à-dire que pour une matrice $\mathbf{M} \in \mathbb{R}^{n\times r}$, où $(m, n) \in \mathbb{N}^2$, nous définissons $s := \|\mathbf{M}\|_0$. Une matrice $\mathbf{M}$ est dite '$s - sparse$' si et seulement si elle vérifie la condition suivante : $\|\mathbf{M}\|_0 \leq s$. L'opérateur $\|\cdot\|_{2,\infty}$ est donné par : $\|\mathbf{M}\|_{2,\infty} := max_{j\in[r]}\|\mathbf{M}\|_2$.La norme infinie est définie comme suit : $\|\beta\|_\infty := \max_{i\in[n]} |\beta_i|$ où $\beta \in R^r$.

Pour simplifier les notations, nous noterons $g(x)$ (ou simplement g) le gradient de la fonction objective, c'est-à-dire $g(x) := \nabla f(x)$, et $g_k(x_k)$ (ou simplement g_k) sera le gradient évalué au point x_k, soit $g_k := \nabla f(x_k)$. La notation $[.]s$ correspond à la meilleure approximation sparse d'un vecteur. En d'autres termes, pour $\beta \in \mathbb{R}^r$,
$[\beta]_0 := \text{argmin}\{\|\beta - x\|_2, \|x\|_0 \leq s\}$.

Pour une fonction objective F, nous noterons également $F^* := \min_{x\in\mathbb{R}^d} F(x)$. Sauf indication contraire, nous adopterons la notation suivante : $e_{x_k} := F(x_k) - F$. De manière générale, un point $(\tilde{x}^*)$ est une solution ϵ-optimale pour un problème d'optimisation si et seulement si $e_{\tilde{x}^*} := F(\tilde{x}^*) - F^* \leq \epsilon$. Enfin, nous noterons $\chi^* = \{x^* : F(x^*) = F^*\}$, et

pour une fonction convexe avec χ^* non vide, nous définirons $P_* := argmin_{y \in \chi^*} \|y - x\|_2$.

3 Le problème d'optimisation à l'ordre zéro

3.1 Motivation

Les algorithmes d'optimisation à l'ordre zéro sont conçus pour minimiser une fonction objective en utilisant uniquement des informations obtenues par son évaluation en un nombre fini de points. Cependant, les algorithmes classiques d'optimisation sans gradient sont généralement peu efficaces lorsqu'il s'agit d'optimiser des fonctions en grande dimension, puisqu'ils doivent explorer un espace de variables étendu sans les indications fournies par le calcul exact des dérivées. Par conséquent, pour ces algorithmes classiques, le nombre d'évaluations de la fonction utilisées à chaque itération peut augmenter de manière linéaire avec la dimension du problème. Cela limite leurs utilisations aux problèmes où le nombre de variables ne dépasse généralement pas la centaine.

Plus récemment, des algorithmes ont fait le choix d'utiliser des directions aléatoires afin d'estimer le gradient et surmonter ce problème fondamental. Les avantages de cette approche sont expliqués ci-dessus ([10],[11]).

3.2 La représentation formelle du problème

Formellement, en adoptant les mêmes notations que [9], le problème peut s'écrire comme suit :
Soit $f : \chi \to R$ où $\chi \subseteq R^d$. f est accessible seulement par le biais d'observations bruitées, c'est-à-dire on observe $y_t := f(x_t) + \xi$. Où ξ est la partie bruit de l'observation, que l'on modélise avec une variable gaussienne. Dans notre étude empirique nous avons fait le choix de prendre $\xi \sim \mathcal{N}(0, \sigma^2)$.
Afin de simplifier les notations on pose : $E_f(x_t) := f(x_t) + \xi$ et on propose de résoudre le problème suivant :

$$min_{x \in \chi} E_f(x) = min_{x \in \chi} f(x) + \xi \tag{1}$$

3.3 Hyptohèses

Dans la littérature se penchant sur cette thématique d'optimisation, on observe fréquemment la formulation des hypothèses suivantes, dont on se servira également pour la démarche adaptative. Par ailleurs ces hypothèses sont essentielles du point de vue tant théorique que pratique :

A1 : La convexité de la fonction objective : $\forall(x, x') \in \chi^2$ et $\lambda \in [0, 1]$, $f(\lambda x + (\lambda - 1)x') \leq \lambda f(x) + (1 - \lambda)f(x')$.

A2 : Le minimiseur est borné en norme l_1 : On suppose $\exists x^* \in \chi, f(x^*) = f^* = inf_{x\in\chi} f(x)$ et $\|x\|_1 \leq B$.

A3 : La sparsité du gradient : $\exists H > 0, s \ll d$, $\|\nabla f(x)\|_0 \leq s, \|\nabla f(x)\|_1 \leq H$, $\forall x \in \chi$.

A4 : La sparsité faible de la Hessienne : On suppose que f est deux fois différentiable et $\exists H > 0 \|\nabla^2 f(x)\|_1 \leq H$.

A5 :La compressibilité du gradient : *On dit que le gradient de la fonction f est compressible si et seulement si :*

$$\exists p \in]0, 1[, \forall x \in R^d |\nabla f(x)|_{(i)} \leq i^{-\frac{1}{p}} \|\nabla f(x)\|_2. \tag{2}$$

L'hypothèse (2) implique d'après [12] : $\forall s \in [|0, d|]$

$$\|\nabla f(x) - [\nabla f(x)_{(s)}]\|_1 \leq (\frac{1}{p} - 1)^2 \|\nabla f(x)\|_2 s^{1-\frac{1}{p}}. \tag{3}$$

$$\|\nabla f(x) - [\nabla f(x)]_{(s)}\|_2 \leq (\frac{2}{p} - 1)^{-\frac{1}{2}} \|\nabla f(x)\|_2 s^{\frac{1}{2}-\frac{1}{p}}. \tag{4}$$

3.4 Prérequis

Défintion 1 (La Propriété d'Isométrie Restreinte (PIR)) *:*

On dit qu'une matrice $Z \in R^{m\times d}$ vérifie la propriété d'Isométrie Restreinte à l'ordre k.s qu'on notera (k.s-PIR) si et seulement si : $\forall v \in R^d$, $\|v\|_0 \leq k.s$, $\exists \delta_{k.s}(Z) \in]0, 1[$

$$(1 - \delta_{k.s}(Z))\|v\|_2^2 \leq \|Zv\|_2^2 \leq (1 + \delta_{k.s}(Z))\|v\|_2^2. \tag{5}$$

Théorème 1 *:*

Soit la suite de vecteurs de Rademacher $(z_i)_{i\in[m]}$ de dimension d. C'est-à-dire que pour tout $i \in [m]$, $z_i \in \{-1, 1\}^d$ et pour tout $j \in [d]$, $\mathcal{P}([z_i]_j = \pm 1) = \frac{1}{2}$. On construit la matrice $Z \in \mathbb{R}^{m\times d}$ de la manière suivante : pour tout $l \in [m]$, $[Z]_l = \frac{1}{\sqrt{m}} z_l^T$, où par abus d'écriture $[Z]_l$ représente la l-ième ligne de la matrice Z. Alors, si m est proportionnel à $s\log(d)$, la matrice Z, ainsi construite, vérifie la 3-s-PIR et la 4-s-PIR presque sûrement.

En effet, soient b_1 et b_2 deux réels indépendants de s, d et m. Si $m = b_1 s \log(d/s)$, alors Z vérifie la 4s-PIR avec $\delta_{4s}(Z) < \sqrt{\frac{2}{5+\sqrt{73}}}$ avec une probabilité de $1 - 2(\frac{d}{s})^{b_2 s}$.

Soient maintenant $\tilde{b}1$ et $\tilde{b}2$ deux réels indépendants de s, d et m. Si $m = \tilde{b}1 s \log(d/s)$, alors Z vérifie la 3s-PIR avec $\delta_{3s}(Z) < \frac{1}{2}$ avec une probabilité de $1 - 2(\frac{d}{s})^{\tilde{b}_2 s}$.

Preuve :

La preuve de ce théorème s'obtient par une application directe du théorème (**5.2**) de [13]. Une preuve détaillée est donnée dans [14].

Théorème 2 *: **Une borne supérieure pour une matrice vérifiant la k.s-PIR [12] :***

Si une matrice Z vérifie la $k.s - PIR$ alors :

$$\forall \nu \in R^d : \|Z\nu\|_2 \leq \sqrt{1+\delta_{k.s}(Z)}(\|x\|_2 + \frac{1}{\sqrt{k.s}}\|x\|_1).$$

La méthode la plus naïve pour estimer le gradient a été introduite dans le schéma d'approximation stochastique de Kiefer-Wolfowitz. Elle consiste à estimer le gradient en décomposant la fonction objective le long de la base canonique, c'est-à-dire [15] :

$$g_k = \sum_{i=0}^{d} \frac{F(x_t + c_t e_i) - F(x_t)}{c_t} e_i.$$

La séquence $(e_i)_{i\in[d]}$ représente la base canonique, c_t est une séquence bien choisie décroissante dans le temps, et d représente la dimension de la fonction objective. Cependant, plutôt que de décomposer la fonction objective en d directions, on propose de la décomposer en utilisant des directions aléatoires pour estimer le gradient.

Ainsi, pour estimer le gradient, nous commençons par fixer un nombre m de requêtes que nous utiliserons pour cette estimation lors d'une seule itération de l'algorithme. Nous choisissons également un réel $\delta > 0$ qui nous permettra d'estimer un voisinage d'un point $x \in \mathbb{R}^d$. Ensuite, nous introduisons une suite de vecteurs aléatoires de type Rademacher $[z_i]_{i\in[|1,m|]} \in \mathbb{R}^d$, c'est-à-dire que pour tout $(i,j) \in [|1,m|] \times [|1,d|]$, nous avons $\mathcal{P}(z_{i,j} = 1) = \mathcal{P}(z_{i,j} = -1) = \frac{1}{2}$.

On pose par la suite :

$$y_i = \frac{E_f(x + \delta z_i) - E_f(x - \delta z_i)}{2\sqrt{m}\delta}. \tag{6}$$

Lemme 1 *Estimation du Gradient :*

On pose :

$$\tilde{y}_i = \frac{E_f(x + \delta z_i) - E_f(x - \delta z_i)}{2\sqrt{m}.\delta}.$$

*Alors si l'hypothèse (**A4**) est vérifiée on a :*

$$\tilde{y}_i = \frac{1}{\sqrt{m}} z_i^T g + \frac{\mu_i}{\delta} + \delta\nu_i. \tag{7}$$

Où : $g = \nabla f(x)$, $\mu_i \leq \frac{\sigma}{\sqrt{m}}$ et $|\nu_i| \leq \frac{H}{2\sqrt{m}}$ et ce $\forall i \in [m]$.

Preuve : Avec un développement limité à l'ordre 2 on obtient en fixant un $x \in R^d$: $\exists t \in]0,1[$ tel que :

$$f(x+\delta z_i) = f(x) + \delta z_i^T \nabla f(x) + \frac{\delta^2}{2} z_i^T \nabla^2 f(x+tz_i) z_i.$$

Ainsi en posant $\nabla f(x) = g$, $E_f(x+\delta z_i) = f(x+\delta z_i) + \xi_+$, $E_f(x-\delta z_i) = f(x-\delta z_i) + \xi_-$ et en appliquant un développement limité à l'ordre 2 pour $f(x-\delta z_i)$, on obtient :

$$\tilde{y}_i = \frac{z_i^T g}{\sqrt{m}} + \frac{\xi_+ - \xi_-}{2\delta\sqrt{m}} + \frac{\delta}{4\sqrt{m}} z_i^T (\nabla^2 f(x+tz_i) - \nabla^2 f(x-tz_i)) z_i.$$

Soit $\mu_i = \frac{\xi_+ - \xi_-}{2\sqrt{m}}$, alors $|\mu_i| \leq \frac{\sigma}{\sqrt{m}}$ et $\nu_i = \frac{1}{4\sqrt{m}} z_i^T (\nabla^2 f(x+tz_i) - \nabla^2 f(x-tz_i)) z_i$, alors $|\nu_i| \leq \frac{H}{2\sqrt{m}}$.

$|z_i^T (\nabla^2 f(x+tz_i) - \nabla^2 f(x-tz_i)) z_i| \leq |z_i^T (\nabla^2 f(x+tz_i) z_i| + |z_i^T \nabla^2 f(x-tz_i)) z_i|$
$= |(\sum_{j,k} \nabla^2_{j,k} f(x+tz_i)(z_i)_k (z_i)_j)| + |\sum_{j,k} \nabla^2_{j,k} f(x+tz_i)(z_i)_k (z_i)_j)|$
$\leq (\|\nabla^2 f(x+tz_i)\|_1 + \|\nabla^2 f(x-tz_i)\|_1) \|z_i\|^2_\infty$
$\leq 2H$ (Car par définition de z_i, $\|z_i\|_\infty = 1$ et en utilisant (**A4**)). Ainsi : $|\nu_i| \leq \frac{H}{2\sqrt{m}}$.

Il devient donc intéressant d'estimer le gradient en résolvant le problème de régression suivant :

$$\hat{g} = argmin_{x \in R^d} \|Zx - Y\|_2^2 + \lambda \|x\|_0. \tag{8}$$

Ce modèle correspond au cadre théorique des méthodes itératives comme l'IHT, ISTA et FISTA. Une convexification de ce problème est possible et donne le cadre théorique de l'estimation Lasso (9).

$$\hat{g} = argmin_{x \in R^d} \|Zx - Y\|_2^2 + \lambda \|x\|_1. \tag{9}$$

Cependant, si **s** est relativement petit et d relativement grand, il est plus intéressant de résoudre l'équation suivante, avec le COSAMP, que (9),car ceci pourrait être plus rapide :

$$\hat{g} = argmin_{\|x\|_0 \leq s} \|Zx - Y\|_2^2. \tag{10}$$

La solution du problème (9), obtenue grâce au **Lasso** classique, présente un biais, contrairement aux solutions des problèmes (10 8). Ce biais constitue une source supplémentaire d'erreur. Dans la suite, nous verrons comment corriger le biais de la solution obtenue par le Lasso classique et comparerons les résultats avec ceux du CoSaMP.

4 L'algorithme de sélection de composantes successives

4.1 Présentation

4.1.1 Motivations

Dans un premier temps, nous avons effectué une étude empirique de l'algorithme de sélection de composants successifs proposé dans [16]. Cet algorithme fait partie des premiers à traiter le problème d'optimisation à l'ordre zéro en grande dimension.

Les articles classiques sur l'optimisation sans gradient ([17], [18]) utilisent des méthodes d'acquisition comprimée (compressed sensing) ou de récupération de la sparsité (sparse recovery) pour estimer le gradient et la Hessienne de la fonction objective à minimiser. Bien que l'algorithme proposé dans [16] repose sur les mêmes principes que ceux des articles précédents ([17], [18]), il se concentre exclusivement sur l'estimation du gradient, car l'estimation de la Hessienne peut s'avérer très complexe et coûteuse, surtout en grande dimension.

4.1.2 Estimation du gradient avec le lasso

Avant de définir l'algorithme de sélection de composants successifs, nous présentons tout d'abord l'algorithme basé sur la régression Lasso, qui sera utilisé dans un premier temps pour estimer le gradient.

Algorithme 1 : Estimation du gradient avec le lasso

Données d'entrée : x_t , n , δ , λ

Simulation de variables aléatoires suivant de type Rademacher : $z_1, z_2, \cdots z_n \in \{1, -1\}^d$

Soit : $\tilde{y}_i := \frac{y_i}{\delta}$, où $y_i := E_f(y_i + \delta z_i)$

On résoud finalement le problème du Lasso classique donné par :

$(\hat{g}_t, \hat{\mu}_t) = argmin_{(g \in R^d, \mu \in R)} \frac{1}{n} \sum_{i=1}^{n} (\tilde{y}_i - g^T z_i - \mu)^2 + \lambda \|g\|_1 + \lambda |\mu|$

return $\underline{\hat{g}_t \ , \ \hat{\mu}_t.}$

Afin de résoudre le problème du Lasso, nous avons initialement utilisé la méthode **sklearn.linear_model.Lasso** de la bibliothèque **sklearn**, ce qui nous a permis d'obtenir nos premiers résultats empiriques. Par la suite, nous avons exploré d'autres méthodes qui suivent le même comportement que l'estimateur du Lasso, tout en permettant d'éviter ou limiter le biais qui peut être associé à cet estimateur. Parmi ces méthodes, nous avons utilisé l'estimateur du **Lasso-débiaisé** ainsi que l'algorithme **Iterative Hard Thresholding (IHT)**, dont nous expliquerons l'utilité plus tard.

4.1.3 L'algorithme de sélection de composants successifs

Ainsi on peut désormais définir l'algorithme principal présenté dans [16] :

Algorithme 2 : L'algorithme de sélection de composants successifs

Données d'entrée : T , η , δ , λ , s , B , ν

1 **Initialisation :** $x_0 = 0$, $\hat{S}_0 = \emptyset$, $\hat{S}_{-1} \neq \emptyset$, $t = 0$, $\tilde{\chi} = \{x \in \chi : \|x\|_1 \leq B\}$, $T^{'} = \lfloor \frac{T}{2s} \rfloor$

2 **while** $|S_t| < s$ et $t < s$ et $\hat{S}_t \neq \hat{S}_{t-1}$ **do**

3 $\quad t \leftarrow t + 1$

4 $\quad$ **Estimation du Gradient :**

5 $\quad (\hat{g}_t, \hat{\mu}_t) \leftarrow$ **Estimation du gradient avec le lasso**$(x_{t-1}$, $T^{'}$, δ , $\lambda)$

6 $\quad$ **seillage :** $\hat{S}_t \leftarrow \hat{S}_{t-1} \cup \{i \in [d] : |[\hat{g}]_i| \geq \eta\}$

7 $\quad$ **On utilise l'algorithme BGD, introduit en annexe :**

8 $\quad x_t \leftarrow$ **BGD**$(\chi, \hat{S}_t, \nu$, $T^{'}$, $x_{t-1})$

9 **return** x_{T+1} **Si** $|\hat{S}_t| = s$**. Sinon** x_T.

4.1.4 Explications

Dans l'algorithme (3), les symboles **B**, **T**, **s**, et ν ont des significations spécifiques. **B** représente la borne supérieure dans l'hypothèse (**A2**) que la fonction objective doit respecter. **T** correspond au budget, c'est-à-dire au nombre d'évaluations de la fonction objective. Pour les algorithmes d'optimisation à l'ordre zéro comme celui-ci, il est crucial d'avoir un budget limité pour atteindre la convergence. **s** représente la sparsité du gradient de la fonction objective, ce qui signifie que $\|\nabla f(x)\|_0 \leq s$ pour tout $x \in \chi$. En pratique, la validation croisée est souvent utilisée pour déterminer le paramètre λ.

En outre, cet algorithme utilise un budget total égal à $\mathbf{2}T^{'}$ à chaque itération. La moitié de ce budget est consacrée à l'estimation du gradient à l'aide du Lasso. Une fois cet estimateur du gradient obtenu, il est utilisé pour créer une séquence de sous-ensembles $[S_t]_{t\in[T+1]}$.L'étape de seuillage (étape 6), est introduite car les observations de la fonction auxquelles on a accès sont bruitées. Elle permet donc de réduire ce bruit. Enfin, lors de chaque itération, une descente de gradient est effectuée en utilisant la restriction de la fonction objective à l'ensemble $[S_t]_{t\in[T+1]}$, d'où l'utilité de l'utilisation de l'algorithme **BGD** défini en annexe.

4.2 Propriétés

Dans cette partie, nous présenterons les différentes propriétés de l'algorithme (3). Pour l'étude de cet algorithme, en plus des hypothèses énoncées au début du rapport, nous posons une hypothèse supplémentaire, plus restrictive, concernant la sparsité de la fonction :

A6 : La sparsité de la fonction objective : $\exists S \subseteq [d]$, $|S| \leq s$ et $f_S : R^{|S|} \to R$ tel que $f(x) \equiv f_S(x_S)$ où x_S représente la restriction de x sur $R^{|S|}$. Comme mentionné plus haut $x \in R^d$.

Comme mentionné dans la partie expliquant le fonctionnement de l'algorithme, l'étape 6 de seuillage vise à réduire l'effet du bruit présent dans les observations auxquelles nous avons accès. Le corollaire suivant montre que les composantes du gradient restreint au support S, ayant une valeur absolue suffisamment élevée, peuvent être détectées à l'aide de cette procédure de seuillage.

Corollaire 1 *: On suppose que les hypothèses **A1** et **A4** sont vérifiées. Soit $\eta := \omega\lambda$, où λ est le paramètre de pénalisation dans le problème du Lasso et ω est une constante assez large c'est-à-dire $\omega > 1$. Soit $\hat{S}(\eta) := \{i \in [d] : |[\hat{g}]_i| > \eta\}$. $\hat{g}_t$ est l'estimateur du Lasso obtenu avec la procédure (1). Alors avec probabilité $1 - \mathcal{O}(d^{-2})$:*

$$\{i \in S : |[\nabla f(x_t)]_i| > 2\eta\} \subseteq \hat{S}(\eta) \subseteq S.$$

Ce corollaire implique qu'avec une grande probabilité, l'estimateur $\hat{S}(\eta)$ n'inclura pas d'éléments qui n'appartiennent pas à S, ce qui signifie que nous ne ferons pas de fausses découvertes avec une grande probabilité. De plus, ce corollaire indique que les éléments de S ayant des dérivées partielles suffisamment larges seront également détectés par $\hat{S}(\eta)$. Le théorème suivant traite la convergence de l'algorithme (3) :

Théorème 3 *: On suupose que les hypothèses **A5** et **A1** sont vérifiées. Supposons également que $T := \Omega(s^3 log\ d)$ et $T \leq d$. Posons les paramètres δ,λ et η de la manière suivante : $\delta \asymp (\frac{\sigma^2 slog(d)}{H^2 T})^{\frac{1}{4}}$, $\lambda \asymp \frac{\sigma}{\delta}\sqrt{\frac{slog(d)}{T}} + \delta H$ et $\eta := \omega\delta$ avec $\omega > 1$, comme dans le corollaire (1). Alors avec probabilité **0.9** :*

$$R_{\mathcal{A}}^{S}(T) \lesssim B(\frac{\sigma^2 H^2 slog(d)}{T})^{\frac{1}{4}} + \tilde{\mathcal{O}}(T^{-\frac{1}{3}}).$$

Où l'opérateur $R_{\mathcal{A}}^{S}(\cdot)$ représente le regret simple c'est-à-dire : $R_{\mathcal{A}}^{S}(T) := f(x_{T+1}) - f^*$. Par ailleurs les paramètres **H** et **B** sont ceux de l'hypothèses (**A2**) et (**A3**). Enfin la notation $\tilde{\mathcal{O}}(.)$ indique une dépendance polynomiale.

4.3 Critiques

L'article [16] représente,certes, l'un des premiers travaux à traiter la problématique de l'optimisation d'ordre zéro en haute dimension. Il se distingue également en tant qu'une des premières études à explorer les taux de convergence associés à cette approche. Cependant, des limitations significatives ont été identifiées dans la démarche.

Pour commencer, il convient de noter que l'étape d'estimation du gradient est exclusivement utilisée lors de la phase de seuillage, alors qu'il aurait été envisageable de l'exploiter pour la descente du gradient. Par ailleurs, dans le contexte de la descente de gradient effectuée par la fonction **BGD**, une alternative plus pertinente aurait été de considérer $c_t := \frac{E_{f_{\hat{S}_i}}(x_t+\delta u_t)-E_{f_{\hat{S}_i}}(x_t-+\delta u_t)}{2\delta}$ plutôt que $c_t := \frac{E_{f_{\hat{S}_i}}(x_t+\delta u_t)-E_{f_{\hat{S}_i}}(x_t)}{\delta}$. Notons qu'une telle modification pourrait potentiellement améliorer les performances, vu que l'article annonce que le taux de regret $T^{-1/3}$ est déduit de l'analyse de l'algorithme **BGD**. En revoyant la valeur de c_t de cette manière, des résultats plus favorables pourraient être obtenus.

En outre, on a pu observer que les résultats théoriques exposés dans l'article susmentionné sont sous-optimaux, notamment les démonstrations sont parfois ambiguës. À titre d'exemple, la démonstration du théorème (3) fait référence à l'application de l'analyse de [19], pour établir une inégalité servant de fondement au résultat du théorème (3), après l'utilisation de l'inégalité de Markov. Cependant, le respect des hypothèses requises par [19] pour la fonction à laquelle cette analyse est appliquée n'a pas été démontré.

Sur le plan pratique, l'approche proposée par [16] s'avère difficile à mettre en œuvre. En effet, elle repose sur la connaissance de la sparsité inhérente à la fonction objectif, un paramètre qui n'est pas toujours accessible dans le contexte d'optimisation considéré. Même si une estimation de ce paramètre peut être obtenue, cela engendre un coût additionnel en termes d'évaluations de la fonction objectif, ce qui contredit l'objectif de minimiser ces évaluations. Ceci pourrait engendrer également une source d'erreur supplémentaire. Par ailleurs, les hypothèses formulées sont souvent restrictives et ne correspondent pas nécessairement à la réalité pratique.

Une amélioration potentielle pourrait découler de l'utilisation d'un estimateur différent du Lasso pour l'estimation du gradient. Étant donné le biais inévitable, il peut engendrer des erreurs supplémentaires. Bien que l'article suggère l'utilisation de la version dé-biaisée du Lasso pour pallier partiellement ce problème, celle-ci n'élimine pas totalement le biais. Par ailleurs, l'article propose l'application de la méthode "Mirror Descent" à la place de la descente de gradient conventionnelle, mais cela requiert la connaissance de l'ensemble contenant le point optimal, ce qui n'est généralement pas réalisable en pratique. En témoigne la figure 7, lorsque le voisinage du point optimal n'est pas connu, le recours à la

méthode "Mirror Descent" se révèle inefficace.

5 ZORO

5.1 Présentation

5.1.1 Motivations

Comme mentionné dans la section précédente, la première approche présentait de nombreuses limites. C'est pourquoi, dans un second temps, nous avons décidé de nous pencher sur une alternative qui permet de réduire certains de ces défauts. Notre objectif est d'étudier cette nouvelle approche afin d'identifier des points d'amélioration possible, ceux-ci seront traités par la suite.

À titre d'exemple l'approche suivie par [9], remplace l'hypothèse,très restrictive portant sur **la sparsité de la fonction objective** (hypothèse A6) par une autre plus réaliste, l'hypothèse sur la **La compressibilité du gradient** (hypothèse A5). Par ailleurs, cette approche suit le schéma d'optimisation plus classique que celui suivit par [1], elle a donc l'avantage d'être plus simple mais tout aussi efficace.

5.1.2 Estimation du gradient avec le CoSaMP

Avant de définir l'algorithme ZORO, nous présentons l'algorithme de Poursuite d'appariement échantillonnage compressive (Compressive Sampling Matching Pursuit - CoSaMP) [20], que nous utiliserons pour estimer le gradient. La principale différence entre l'estimateur du Lasso et CoSaMP est que le premier résout le problème (9), tandis que le second résout le problème (10).

Algorithme 3 : Poursuite d'appariement échantillonnage compressif(compressive sampling matching pursuit CoSaMP)

Données d'entrée : Z , Y , s

$a_0 \leftarrow 0$

$\nu \leftarrow Y$

$k \leftarrow 0$ **while** Le critère d'arrêt n'est pas satisfait **do**

  $k \leftarrow k + 1$

  $x \leftarrow Z^T \cdot \nu$

  $\Omega \leftarrow supp(y_{2s})$

  $T \leftarrow \Omega \cup supp(a_{k-1})$

  $b_{|T} \leftarrow Z_T^\dagger \cdot Y$ ($Z_T^\dagger$ représente la restriction du pseudo-inverse de Z sur ses T composantes de plus grande magnitude)

  $b_{|T^c} \leftarrow 0$

  $a_k = b_s$

  $\nu \leftarrow Y - Z \cdot a_k$

return a_K Où a_K représente le vecteur a obtenu à la dernière itteration.

5.1.3 L'algorithme ZORO

Ainsi, on peut désormais définir l'algorithme principal présenté dans [9] :

Algorithme 4 : Méthode d'optimisation régularisée d'ordre zéro (ZORO)

Données d'entrée : x_0 , δ , λ , K

$m \leftarrow b_1 s \cdot log(d/s)$ On peut choisir $b_1 = 1$

$z_1, z_2, ..., z_m \leftarrow$ des vecteurs aléatoires qui suivent la lois de Rademacher

for k=0 **To** K **do**

  for i=0 **To** m **do**

    $\tilde{y}_i = \frac{E_f(x_t+\delta z_i) - E_f(x_t)}{\sqrt{m}\delta}$

  $\tilde{Y}_k \leftarrow \frac{1}{\sqrt{m}} \cdot [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_m]^T$

  $Z \leftarrow \frac{1}{\sqrt{m}} \cdot [z_1, z_2, ..., z_m]$

  $\hat{g}_k \leftarrow$**Estimation du gradient avec le CoSaMP**(Z , $\tilde{Y}_k$, s)

  $x_{k+1} \leftarrow x_k - \alpha \cdot \hat{g}_k$

return x_K

Sur les figures quand on mentionnera l'algorithme **ZORO with DLasso**, ceci correspond à l'algorithme (4), mais on utilise l'estimateur du lasso dé-biaisé à l'étape (**8**) à la place du CoSaMP.

5.1.4 Explications

Dans l'algorithme (4), α représente le pas d'apprentissage pour la descente de gradient. Le paramètre δ est utilisé pour l'estimation du gradient et doit être choisi assez petit. K représente le budget. Pour trouver une solution ϵ-optimale [2],si la fonction objective respecte les hypothèses (A4, A5, 3 et la coercivité), il suffit d'utiliser un budget égal à $\tilde{K}$ où :

$$\tilde{K} := \frac{4b_1 s log(d)/LR^2}{\epsilon(1-8\psi)^2}.$$

Avec $\psi := b_4 s^{1/2-1/p}$. Ceci est vrais pour tous ϵ qui vérifie la condition suivante :
$\epsilon \geq b_3 R\sqrt{2\sigma H/(1-8\psi^2)}$.$(b_1, b_3, b_4)$ sont des constantes qui seront précisées dans la section suivante. Il est également possible de simplifier cette complexité sous d'autres hypothèses plus fortes, ce point sera également abordé dans la partie suivante.

L'algorithme ZORO suit le schéma traditionnel de l'optimisation sous contrainte. En effet, la première étape lors de chaque itération consiste à estimer le gradient en utilisant l'expression de $\tilde{y}$ de l'étape 5. Par un raisonnement similaire à celui présenté dans la

2. Une solution x_K est dite ϵ-optimale dans le contexte considéré si et seulement si : $E_f(x_K) - E_f^* \leq \epsilon$.

section des prérequis, il est possible de montrer que :

$$\tilde{y}_i = \frac{E_f(x + \delta z_i) - E_f(x)}{\sqrt{m}\delta} = \frac{1}{\sqrt{m}} z_i^T g + \frac{\mu_i}{\delta} + \delta\nu_i.$$

Avec $g = \nabla f(x)$, $|\mu_i| \leq \frac{2\sigma}{\sqrt{m}}$ et $|\nu_i| \leq \frac{H}{2\sqrt{m}}$.

Une fois que l'estimateur du gradient obtenu, nous passons ensuite à l'étape de descente de gradient, correspondant à l'étape (9). Dans le but d'améliorer les performances de cet algorithme, nous avons cherché à améliorer la précision de l'estimation du gradient et à rendre cette partie adaptive, c'est-à-dire, à réduire la dépendance par rapport à s.

5.2 Propriétés

Avant d'explorer les différentes propriétés de l'algorithme (4), nous allons d'abord introduire quelques hypothèses supplémentaires qui nous seront utiles pour élaborer les résultats ultérieurs.

A7 : La coercivité :

• On dit qu'une fonction f est coercive si :

$$\forall (x_k)_{k\in\mathcal{N}^*}, lim_{k\to\infty}\|X_k - P_*(x_k)\|_2 = +\infty \implies lim_{k\to\infty} f(x_k) = +\infty. \tag{11}$$

• On dit que le gradient de la fonction f, $\nabla f(.)$, est coercive par rapport à f si :

$$\forall (x_k)_{k\in\mathcal{N}^*}, lim_{k\to\infty} f(x_k) = +\infty \text{ et également },lim_{k\to\infty}\|\nabla f(x_k)\|_2 = +\infty. \tag{12}$$

• On dit que la dérivée partielle de la fonction f, $\partial f(.)$, est coercive par rapport à f si :

$$\forall (x_k)_{k\in\mathcal{N}^*}, lim_{k\to\infty} f(x_k) = +\infty \text{ et également },lim_{k\to\infty} inf_{u\in\partial f(x_k)}\|u\|_2 = +\infty. \tag{13}$$

Le premier théorème important présenté dans l'article [9] porte sur le comportement du gradient estimé par le CoSaMP, ce qui correspond à la ligne (8) de l'algorithme (4). Ce théorème permet de borner l'erreur de l'estimation du gradient. Nous démontrerons un résultat similaire dans la section suivante.

Théorème 4 *: Soit f la fonction objective à minimiser. Supposons que l'ensemble des minimiseurs de cette fonction, notons le* $\chi^* := \{x^* \in \chi, f(x^*) = f^*\}$*, est non vide. Autrement dit* $\chi^* \neq \emptyset$*. Supposons également que le gradiend de f est lipschitz, sparse(hypothèse* ***A3****) et que la fonction est convexe. Soit* $(\hat{g}_k)$ *la suite obtenue avec la ligne (****8****).Alors avec*

probabilité au moins égale à $1-2(s/d)^{b_2 s}$ *:*

$$\|\hat{g}_k - g_k\|_2 \leq (\psi + \rho^n)\|g_k\|_2 + 2\frac{2\sigma\tau}{\delta} + \frac{\tau\delta H}{2}. \tag{14}$$

les constantes : *ρ vérifie $\rho < 1$ et $\tau \approx 10$. Ces deux constantes sont fixes. b_1 et b_2 sont choisies de façon à ce que* ***Z*** *vérifie le théorème 1. Finalement ψ vérifie : $\psi := b_4 s^{1/2-1/p}$ où :*

$$b_4 := (1+\tau\sqrt{1+\delta_{4s}(Z)}(\frac{2}{p}-1)^{-1/2} + \tau\sqrt{1+\delta_{4s}(Z)}(\frac{1}{p}-1)^{-1}.$$

Comme toute inégalité de concentration, il est possible d'optimiser le résultat de ce théorème en optimisant l'expression de la droite en fonction de δ. Ainsi, en choisissant $\delta := 2\sqrt{\frac{\sigma}{H}}$:

Corollaire 2 *: Sous les mêmes hypothèses du* ***théorème 4****. L'inégalité suivante est vraie avec une probabilité d'au moins égale à* $1-2(s/d)^{b_2 s}$ *:*

$$\|\hat{g} - g\|_2 \leq (\psi + \rho^n)\|g\|_2 + 2\tau\sqrt{\sigma H}. \tag{15}$$

Le théorème suivant établit l'ordre de grandeur du budget nécessaire pour trouver la solution optimale avec l'algorithme (4). Comme précisé précédemment, il est essentiel de minimiser ce coût dans le problème que nous considérons.

Théorème 5 *: Soit f la fonction objective à minimiser. Supposons que f vérifie les mêmes hypothèses que celles précisées au* ***théorème 4****. Supposons en plus la sparsité faible de la Hessienne de f (hypothèse* ***A4****). En choisissant s assez grand de manière à avoir $\psi \leq 0.35$ et $\alpha = \frac{1}{L}$. Alors l'algorithme 4 trouve une solution ϵ-optimile avec un budget égal à :*

$$\frac{4b_1 log(d) L R^2}{\epsilon(1-8\psi^2)}, \forall \epsilon > b_3 R\sqrt{2\sigma H/(1-8\psi^2)}. \tag{16}$$

Par ailleurs si à la place de supposer la coércivité de la fonction objective on on suppose la convexité forte restreinte (Restricted strong convexity) c'est-à-dire : $\forall x \in R^d, \exists \alpha \in R$ tel que :
$f(x) - minf \geq \alpha/2\|x - P_*(x)\|_2^2$. Alors dans ce cas le budget nécessaire devient :

$$\frac{b_1 s log(d) log(\frac{\epsilon}{\mathcal{E}_0} - \frac{2b_3^2\sigma\upsilon H}{\upsilon\mathcal{E}_0(1-8\psi)^2})}{log(1-\frac{(1-8\psi^2)\psi}{4L})} = \mathcal{O}(s log(d) log(\frac{1}{\epsilon})). \tag{17}$$

Avec $\mathcal{E}_k := f(x_k) - f^*$.

5.3 Critiques

Les résultats présentés dans l'article [9] sont bien plus précis que ceux de l'article [1], comme détaillé dans la section 3 de cet article. Les résultats de l'algorithme 4 atteignent,certes, l'état de l'art. Cependant, cette approche dépend du paramètre s. Notre objectif est donc de trouver une démarche équivalente, aussi optimale, mais adaptative. Bien qu'il ait été possible d'utiliser le Lasso ou sa version débiaisée pour atteindre cet objectif, la présence inévitable de biais dans cette démarche rend les résultats sous-optimaux. C'est pourquoi nous avons opté pour l'algorithme IHT. Une brève explication du biais associé à l'estimateur du Lasso débiaisé est présentée en annexe.

5.3.1 Utilisation du Lasso dé-biaisé

Comme mentionné dans la section 4.3, l'une des principales limites de l'estimateur du Lasso est qu'il souffre d'un biais qu'il est impossible d'éliminer. Ce biais pourrait donc constituer une source d'erreur supplémentaire. Comme le montrent les figures (3,4) il n'est pas possible d'avoir d'aussi bons résultats d'optimisation avec le lasso dé-biaisé que ceux avec le CoSaMP. Ainsi, nous avons décidé d'étudier le comportement de l'algorithme en utilisant d'abord l'estimateur du Lasso débiaisé, puis de le comparer à la version originale. En effet, pour rendre l'algorithme adaptatif et éliminer la dépendance au paramètre de sparsité 's', nous allons utiliser la version adaptative de l'algorithme IHT (Itérative Hard Thresholding) ([21]). Étant donné que l'IHT résout le problème (8), nous estimons qu'il devrait avoir un comportement similaire à la version de l'algorithme avec l'estimateur du Lasso débiaisé. L'avantage de l'algorithme IHT est que sa solution n'est pas biaisée.

Algorithme 5 : Estimation du gradient avec le lasso dé-biaisé

Données d'entrée : x_t , n , δ , λ , Z , $\tilde{Y}_t$

$(\hat{g}_t, \hat{\mu}_t) = argmi_{(g\in R^d, \mu\in R)} \frac{1}{n}\sum_{i=1}^{n}(\tilde{y}_i - g^T z_i - \mu)^2 + \lambda\|g\|_1 + \lambda|\mu|$

$\tilde{g}_t \leftarrow \hat{g}_t + \frac{1}{n} Z_t(\tilde{Y}_t - Z_t\hat{g}_t - \hat{\mu}_t \cdot \mathbf{1}_n)$

return $\underline{\tilde{g}_t}$

6 L'alternative adaptative de l'algorithme ZORO

6.1 L'utilisation de l'algorithme IHT classique

6.1.1 Motivation

Comme indiqué précédemment dans le rapport, l'objectif de ce stage est de proposer une alternative adaptative aux algorithmes présentés, avec des performances se rapprochant de celles de l'état de l'art. Pour atteindre cet objectif, nous utilisons la version adaptative de l'algorithme IHT (Itérative Hard Thresholding), telle qu'elle est présentée dans l'article [21]. Pour obtenir une première idée de cette approche, nous avons choisi de commencer par utiliser l'approche classique de l'IHT, car elle est plus simple à étudier.

6.1.2 Présentation de l'IHT classique

Rappelons dans un premier temps que la régression par LASSO produit un biais qui est inévitable, c'est aussi le cas de certains estimateurs convexes par exemple l'estimateur SLOPE. Or l'algorithme IHT permet d'enlever ce biais dans certains cas [21], d'où notre intérêt à utiliser cette procédure. On se place dans le cadre du modèle suivant.

$$Y = X\beta + \sigma\xi. \tag{18}$$

où Y représente l'observable, le vecteur signal β un signal s-sparse, $\|\beta\|_0 \leq s$ et ξ un vecteur centré de bruit. On note $\|\beta_0\|$ le nombre de coordonnées non nuls de β.

On pose l'algorithme suivant :

$$\hat{\beta}_0 = 0, \forall n \geq 0, \hat{\beta}_{n+1} = H_{\lambda_{n+1}}\left(\hat{\beta}_n + \frac{1}{n+1}X^T(Y - X\hat{\beta}_n)\right), \tag{19}$$

où $(\lambda_n)_n$ est une suite positive, et H_λ est l'opérateur de Hard Thresholding défini comme suit : $H_\lambda : R^d \to R^d$

$$\forall u \in R^d, \forall j = 1, \ldots, d, \quad H_\lambda(u)j = u_j \mathbf{1}\{|u_j| \geq \lambda\}. \tag{20}$$

Dans [22] on montre que si $\|\beta\|_0 \leq s$ cet algorithme converge vers un minimum local du problème d'optimisation suivant :

$$\min_\beta \frac{1}{2n}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_0. \tag{21}$$

Cette procédure peut être interprétée comme une descente de gradient projetée sur un ensemble non convexe.

6.1.3 ZORO avec l'IHT classique

On commence par étudier l'algorithme suivant et essaye de montrer des résultats similaires à ceux présentés dans l'article [16].

Algorithme 6 : Méthode d'optimisation régularisée d'ordre zéro avec IHT (ZORO avec IHT)

Données d'entrée : x_0 , δ , λ , K

$m \leftarrow b_1 s \cdot log(d/s)$ On peut choisir $b_1 = 1$

$z_1, z_2, ..., z_m \leftarrow$ des vecteurs aléatoires qui suivent la lois de Rademacher

for k=0 **To** K **do**

  for i=0 **To** m **do**

    $\tilde{y}_i = \frac{E_f(x_t+\delta z_i) - E_f(x_t - \delta z_i)}{2\sqrt{m}\delta}$

  $\tilde{Y}_k \leftarrow \frac{1}{\sqrt{m}} \cdot [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_m]^T$

  $Z \leftarrow \frac{1}{\sqrt{m}} \cdot [z_1, z_2, ..., z_m]$

  $\hat{g}_k \leftarrow$**Estimation du gradient avec l'IHT**(Z , $\tilde{Y}_k$, s)

  $x_{k+1} \leftarrow x_k - \alpha \cdot \hat{g}_k$

return x_K

6.1.4 Résultats Théoriques

Théorème 6 *:*

Soit $(g_n)_{n\in N}$*, la suite obtenue par l'algorithme IHT. Si* $m = \tilde{b}_1.s.log(d/s)$ *et* $g_0 = 0$*.Soit* $[g]_s$ *la meilleure approximation s-sparse du vecteur* g*. Alors* $\forall g \in R^d$ *avec probabilité* $1 - 2(\frac{d}{s})^{\tilde{b}_2 s}$ *:*

$$\|g_n - g\|_2 \leq \|g - [g]_s\|_2 + \tau\|Z(g - [g]_s)\|_2 + \frac{\tau}{\delta}\|\mu\|_2 + \tau\delta\|\nu\|_2 + \rho^n\|[g]_s\|_2. \tag{22}$$

Avec $\tau = \frac{2\sqrt{1+\delta_{2s}}}{1-2\delta_{3s}}$ *avec* $\delta_{2s} < \frac{3}{4+\sqrt{6}}$ *et* $\rho \leq 1$.

Preuve :

Soit $(g_n)_{n\in N}$, la suite obtenue par l'algorithme IHT. Si $m \geq \tilde{b}_1.s.log(d/s)$, alors d'après le **Théorème 1** Z vérifie la la (3s-PRI) avec $\delta_{3.s}(Z) < \frac{1}{2}$ avec probabilité $1 - 2(\frac{d}{s})^{\tilde{b}_2 s}$. On applique le **Théorème 3** de [14], alors on a avec la même probabilité :

$\|g_n - g\|_2 \leq \|g - [g]_{(s)}\|_2 + \|g_n - [g]_s\|_2$ (Inégalité triangulaire).

$\leq \|g - [g]_{(s)}\|_2 + \rho^n\|[g]_{(s)}\|_2 + \tau\|Z(g - [g]_{(s)}) + \frac{1}{\delta}\mu + \delta\nu\|_2$ (Théorème 3 de [14])

$\leq \|g - [g]_{(s)}\|_2 + \tau\|Z(g - [g]_{(s)})\|_2 + \frac{\tau}{\delta}\|\mu\|_2 + \tau\delta\|\nu\|_2 + \rho^n\|[g]_{(s)}\|_2$. (Inégalité triangulaire)

Théorème 7 *: Si la matrice Z vérifie les conditions du théorème (1) alors avec probabilité $1-2(\frac{d}{s})^{\tilde{b}_2 s}$, on a : $\forall s \in [|0,d|], \forall \tau \in R$:*

$$\|g-[g]_{(s)}\|_2+\tau\|Z(g-[g]_{(s)}\|_2 \leq \psi\|g\|_2. \tag{23}$$

Avec

$$\psi = (1+\tau\sqrt{1+\delta_{3.s}(Z)})(\frac{2}{p}-1)^{-\frac{1}{2}}) + \tau\sqrt{1+\delta_{3.s}(Z)}(\frac{1}{p}-1)^{-1}s^{\frac{1}{2}-\frac{1}{p}}.$$

Preuve :

Si Z vérifie les conditions du théorème (1) alors avec probabilité $1-2(\frac{d}{s})^{\tilde{b}_2 s}$, Z vérifie la $3s-PIR$, d'après (1).
D'après (2) :

$\|Z(g-[g]_{(s)})\|_2 \leq \sqrt{1+\delta_{3.s}(Z)}(\|g-[g]_{(s)}\|_2+\frac{1}{\sqrt{s}}\|g-[g]_{(s)}\|_1)$
$\leq \sqrt{1+\delta_{3.s}}(\frac{2}{p}-1)^{-\frac{1}{2}}\|g\|_2 s^{\frac{1}{2}-\frac{1}{p}} + \frac{1}{\sqrt{s}}(\frac{1}{p}-1)^{-1}\|g\|_2 s^{1-\frac{1}{p}}$ (On utilise (3) et (4))
$= \sqrt{1+\delta_{3.s}(Z)}(\frac{2}{p}-1)^{-\frac{1}{2}} + (\frac{1}{p}-1)^{-1})s^{\frac{1}{2}-\frac{1}{p}}\|g\|_2$.
On utilise ensuite la condition (4) pour borner $\|g-[g]_{(s)}\|_2$ également et conclure.

Théorème 8 *: Sous l'hypothèse d'existence d'un minimum de la fonction objective, du fait que le gradient est lipschitz,* ***A3*** *et* ***A5****. Soit $(\hat{g_n})$ l'estimateur du gradient produit avec l'IHT, alors on peut borner l'erreur d'estimation de la façon suivante :*

$$\forall k, \|\hat{g}_k - g\|_2 \leq (\psi+\rho^n)\|g\|_2+2\tau\sqrt{\sigma}H.$$

Où ψ est celui précisé dans le théorème 7 et ρ τ sont ceux précisés dans le théorème 6. Ce résultat est vrai avec probabilité $1-2(s/d)^{\tilde{b}_2 s}$.

Preuve :

On montre dans un premier temps que le résultat suivant est vrai avec probabilité $1-2(s/d)^{\tilde{b}_2 s}$: $\forall k, \|\hat{g}_k - g\|_2 \leq (\psi+\rho^n)\|g\|_2+2\frac{\tau\sigma}{\delta}+\frac{\tau\delta H}{2}$. On obtient ce résultat en combinant le théorème 6 et 7. On utilise également : $\|\mu\|_2 \leq 2\sigma$ et $\|\nu\|_2 \leq \frac{H}{2}$. Comme ce résultat est vrai pour tout δ, alors en minimisant la borne de droite par rapport à δ, on obtient le résultat pour $\delta = 2\sqrt{\frac{\sigma}{H}}$.

Rappelons qu'on utilise la descente de gradient classique, c'est-à-dire $x_{k+1} \leftarrow x_k + \alpha \hat{g}_k$, avec l'estimateur du gradient qui vérifie l'équation suivante :

$$\|g - \hat{g}_k\|_2^2 \leq \epsilon_{rel} \|\hat{g}_k\|_2^2 + \epsilon_{abs}. \tag{24}$$

Pour montrer la suite des résultats on utilisera le lemme suivant [16].

Lemme 2 *Supposons que la fonction objective à minimiser f vérifie les hypothèses de coércivité, existence du minimum et son gradient est Lipschitz. Supposons en plus que la suite d'estimateur du gradient vérifie l'équation 24 pour tout k avec $\epsilon_{rel} < 1$. En posant $\alpha = \frac{1}{L}$ on obtient :*

$$e_k \leq max \left[\frac{4LR^2 e_0}{(1-\epsilon_{rel})e_0 k + 4LR^2}, R\sqrt{\frac{2\epsilon_{abs}}{1-\epsilon_{rel}}} \right]$$

.

R est une constante qui vérifie : $\forall k, \|x_k - P_*(x_k)\|_2 \leq R$.Si par ailleurs l'hypothèse de la convexité fortement restreinte , est vérifiée au lieu des hypothèses sur la coércivité alors cette borne peut être améliorer pour obtenir :

$$e_k \leq (1 - \frac{(1-\epsilon_{rel})\nu}{4L}) e_0 + \frac{2\epsilon_{abs}}{\nu(1-\epsilon_{rel})}.$$

Théorème 9 *Si la fonction objective à minimiser : f, est convexe et vérifie les différentes hypothèses suivantes : Compressibilité du gradient, sparsité faible de la Hessienne, l'existence du minimum, si en plus le gradient est lipschitz et la fonction répond aux conditions de coércivité, en fixant le pas de la descente de gradient tel que : $\alpha = 1/L$. Soit n le nombre d'itérations de l'algorithme IHT alors pour n assez grand, $\rho^n < \psi$. Ainsi si $\psi < 1/36$ $\epsilon_{rel} := 2(\psi + \rho^n)^2 < 8\psi^2 < 1$: avec probabilité égale à $1 - 2(\frac{s}{d})^{b_2 s}$ l'algorithme trouve une solution ϵ-opptimale en utilisant $\frac{4b_1 s log(d) LR^2}{\epsilon(1-8\psi^2)}$, $\forall \epsilon > b_3 R\sqrt{\frac{2\sigma H}{1-8\psi^2}}$.*

6.2 L'utilisation de l'algorithme IHT adaptatif

6.2.1 Présentation de l'algorithme IHT adaptatif

On se place toujours dans le cadre du même modèle(18). Dans cette démarche, on adopte une procédure de seuillage qui interpole, dans un sens, deux seuils classiques à savoir le plus grand s dans l'algorithme IHT et $\sigma\sqrt{\frac{2log(d)}{n}}$ dans la procédure de régression linéaire par Lasso [21].Par ailleurs, cette procédure garantie, implicitement, à chaque itération la sparsité de l'estimateur sans avoir à choisir exactement s composantes. Afin de garantir un résultat statistiquement optimal et une convergence rapide, on définit la suite λ_m de la manière suivante :

$$\lambda_0 \in R, \forall m \in N, \lambda_m = max(\kappa^{m/2}\lambda_0, \lambda_\infty), \text{où } \kappa \text{ est une constante fixée.} \tag{25}$$

Il reste ainsi que λ_0 et λ_∞ à définir. En ce qui concerne le premier paramètre λ_0, le choix de celui-ci n'a pas vraiment d'importance. Cependant pour réduire le budget utilisé par notre algorithme il convient de choisir le λ_0 optimal proposé dans [21]. Pour λ_∞, qu'on pourrait interpréter comme étant une règle de sortie de l'algorithme, pratiquement on utilise la cross validation pour le déterminer. Une valeur théorique de ce dernier peut être donnée par, $\lambda_\infty = \sqrt{\frac{2\sigma^2 log(ed/s)}{\|X\|^2_{2,\infty}}}$ [21].

Dans [21], on suppose que le bruit ζ est indépendant de la matrice d'échantillonnage [3] X. Or, dans notre situation ce n'est pas le cas. En effet, rappelons que le modèle que nous considérons s'écrit de la manière suivante :

$$\tilde{y_i} = \frac{1}{\sqrt{m}} z_i^t g + \frac{\mu_i}{\delta} + \delta\nu_i.$$

Avec $\tilde{y_i} := \frac{E_f(x+\delta z_i) - E_f(x-\delta z_i)}{2\sqrt{m}\delta}$, $g := \nabla f(x)$, $\mu_i := \frac{\xi_+ - \xi_-}{2\sqrt{m}}$ et $\nu_i := \frac{\delta}{4\sqrt{m}} z_i^t(\nabla^2 f(x + tz_i) - \nabla^2 f(x - tz_i))$.

Il est évident que le niveau de bruit dans notre modèle dépend de la matrice d'échantillonnage. Par conséquent, il est nécessaire de modifier la valeur limite théorique de λ afin d'adapter la procédure à cette contrainte. Ce changement correspond au prix à payer pour cette dépendance. En pratique, il n'est pas nécessaire de modifier l'algorithme pour prendre en compte ce changement, car nous pouvons obtenir cette valeur limite grâce à la validation croisée.

Ainsi, on pose désormais, $\lambda_\infty := 2\sqrt{\frac{1+3\delta}{sm}}(2\sigma + \frac{H}{2})$. On posera également $\Xi := \frac{\sigma Z^T(\frac{\mu}{\delta} + \delta\nu)}{\|Z\|^2_{2,\infty}}$ et $\Phi := \frac{1}{\|Z\|^2_{2,\infty}} Z^T Z - I_p$. En reprenant le cadre du modèle générale 18 pour des raisons de simplicité d'écriture, on peut désormais définir l'algorithme IHT adaptatif de la manière

3. Sensing Matrix

suivante :

$$\forall n \geq 1, \tilde{\beta}_{n+1} = T_{\lambda_{n+1}} \left(\tilde{\beta}_n + \frac{1}{\|X\|^2_{2,\infty}} X^T (Y - X\tilde{\beta}_n) \right). \tag{26}$$

$$\lambda_n = \max(\kappa^{n/2}\lambda_0, \lambda_\infty), m = 1, ...$$

κ est un paramètre fixe, tel que $0 < \kappa < 1$. Il est recommandé de choisir κ très proche de 1. L'opérateur de Hard Thresholding :T_λ est définit de la manière suivante :

$$H_\lambda : R^p \to R^p$$

$$\forall u \in R^p, \forall j = 1, \ldots, p, \quad H_\lambda(u)j = u_j \mathbf{1}\{|u_j| \geq \lambda\}.$$

6.2.2 ZORO avec l'IHT adaptatif

On étudie le ZORO avec la procédure adaptative, en prenant en considération les changements dûs à la dépendance entre le bruit et la matrice d'échantillonnage.

Algorithme 7 : Méthode d'optimisation régularisée d'ordre zéro avec IHT (ZORO avec IHT)

Données d'entrée : x_0 , δ , λ , K

$m \leftarrow b_1 s \cdot log(d/s)$ On peut choisir $b_1 = 1$

$z_1, z_2, ..., z_m \leftarrow$ des vecteurs aléatoires qui suivent la lois de Rademacher

for k=0 **To** K **do**

  for i=0 **To** m **do**

    $\tilde{y}_i = \frac{E_f(x_t + \delta z_i) - E_f(x_t - \delta z_i)}{2\sqrt{m}\delta}$

  $\tilde{Y}_k \leftarrow \frac{1}{\sqrt{m}} \cdot [\tilde{y}_1, \tilde{y}_2, ..., \tilde{y}_m]^T$

  $Z \leftarrow \frac{1}{\sqrt{m}} \cdot [z_1, z_2, ..., z_m]$

  $\hat{g}_k \leftarrow$**Estimation du gradient avec l'IHT adaptatif**(Z , $\tilde{Y}_k$, s)

  $x_{k+1} \leftarrow x_k - \alpha \cdot \hat{g}_k$

return x_K

6.2.3 Résultats théorique

Nous commençons par démontrer un résultat similaire au "Théorème 1" de l'article [21], en tenant compte de la dépendance entre le bruit et la matrice d'échantillonnage. Pour des raisons de simplicité d'écriture, nous reprenons le cadre du modèle général 18. On pose l'événement $\mathcal{O} := \{\sum_{i\in[|0,s|]} \Xi_i^2 \leq (1+3\delta)\|\xi\|^2/m\}$.

Théorème 10 *:*

Soit $(\hat{\beta}_m)$ *la suite obtenue avec l'IHT adaptatif. Si* β *est s-sparse (c'est à dire* $\|\beta\|_0 \leq$ *s), si X satisfait la propriété* $RIP(3s, \delta/2)$*. Soient* λ_0 *et* λ_∞*>0. Soient* $0 < \kappa < 1$*,* $\delta < 1/36 \wedge \kappa$*,* $2\sqrt{\frac{1+3\delta}{sn}}(2\sigma + \frac{H}{2}) \leq \lambda_\infty$*. On notera cette dernière hypothèse* ***B1****. Si on plus l'événement* $\mathcal{O}$*est vérifié alors :*

$$\|\hat{\beta}^m_{S^C}\| \leq s, \tag{27}$$

$$\|\beta - \hat{\beta}_m\|^2 \leq 9s\lambda_m^2. \tag{28}$$

Preuve :

On prouve le théorème avec un raisonnement par récurrence. Pour m= 0 le résultat est vrai. Soit $m \in \mathcal{N}$, supposons que la propriété est vérifiée pour m et montrons qu'elle reste vraie pour $m+1$.

Soit $H^{m+1} := \hat{\beta}_m + \frac{1}{\|X\|^2_{2,\infty}} X^T(Y - X\hat{\beta}_m)$.

$H^{m+1} = \beta + \Phi(\beta - \hat{\beta}_m) + \Xi$.

Ainsi : $\hat{\beta}_{m+1} = T_{\lambda_{m+1}}(H^{m+1}) = T_{\lambda_{m+1}}(\beta + \Phi(\beta - \hat{\beta}_m) + \Xi)$. On prouve la première partie du théorème par un raisonnement par l'absurde. Supposons que $\|\hat{\beta}^{m+1}_{S^C}\| > s$, alors il existe un sous ensemble de S^C, $\tilde{S}$ tel que $\|\tilde{S}\|_0 = s$ et :

$$s\lambda^2_{m+1} \leq \sum_{i\in\tilde{S}} (H_i^{m+1})^2 \mathbf{1}\{|H_i^{m+1}| \geq \lambda_{m+1}\}.$$

En utilisant la définition de H^{m+1} on obtient :

$$\sqrt{s}\lambda_{m+1} \leq \sqrt{\sum_{i\in\tilde{S}} \Xi_i^2} + \sqrt{\sum_{i\in\tilde{S}} < \Phi_i^T, \beta - \hat{\beta}_m >^2}.$$

Comme β est s-sparse et $\|\hat{\beta}^m_{S^C}\| \leq s$ (par l'hypothèse de la récurrence) alors le vecteur $\beta - \hat{\beta}_m$ est au plus 2s-sparse. Avec le lemme 1 de l'article [21] on obtient :

$$\sqrt{s}\lambda_{m+1} \leq \sqrt{\sum_{i\in\tilde{S}} \Xi_i^2} + \delta\|\beta - \hat{\beta}_m\|.$$

En utilisant l'événement $\mathcal{O}$, si $\delta < \frac{1}{36}$:

$$\sqrt{s}\lambda_{m+1} \leq \frac{\sqrt{(1+3\delta)}\|\xi\|}{\sqrt{n}} + 3\sqrt{s}\delta\lambda_m.$$

$$\sqrt{s}\lambda_{m+1} \leq (\frac{1}{2} + 3\sqrt{\delta})\sqrt{s}\lambda_{m+1} < \sqrt{s}\lambda_{m+1}.$$

Pour avoir l'avant dérnière inégalité on utilise le fait que le bruit se décompose de la façon suivante : $\xi = \mu + \nu$, tel que $\|\mu\| \leq 2\sigma$ et $\|\nu\| \leq \frac{H}{2}$ et l'hypothèse : $2\sqrt{\frac{1+3\delta}{n}}(2\sigma + \frac{H}{2}) \leq \lambda_\infty$. On aboutit ainsi à une contradiction, d'où $\|\hat{\beta}^m_{S^C}\| \leq s$.

Par ailleurs remarquons que $\forall i \in S$:
$(\hat{\beta}_m)_i - \beta_i = (H_i^{m+1})\mathbf{1}\{|H_i^{m+1}| \geq \lambda_{m+1}\} - H_i^{m+1} + \Xi_i + < \Phi_i^T, \beta - \hat{\beta}_m >$
$= -(H_i^{m+1})\mathbf{1}\{|H_i^{m+1}| \leq \lambda_{m+1}\} + \Xi_i + < \Phi_i^T, \beta - \hat{\beta}_m >$.

Ainsi en utilisant les mêmes arguments que dans la première partie de la démonstration on obtient :

$$\|\beta_S^{m+1} - \beta\| \leq \sqrt{s}\lambda_{m+1} + \sqrt{\frac{1+3\delta}{n}}\|\xi\| + \delta\|\hat{\beta}_m - \beta\|.$$

Et

$$\|\beta_{S^C}^{m+1}\| \leq \sqrt{\frac{1+3\delta}{n}}\|\xi\| + \delta\|\hat{\beta}_m - \beta\|.$$

D'où

$$\|\beta^{m+1} - \beta\| = \|\beta_S^{m+1} + \beta_{S^C}^{m+1} - \beta\| \leq \sqrt{s}\lambda_{m+1} + 2\sqrt{\frac{1+3\delta}{n}}\|\xi\| + 2\delta\|\hat{\beta}_m - \beta\|.$$

En utilisant l'hypothèse **B1**, l'hypothèse de récurrence ainsi que la définition de λ_m on obtient finalement :

$$\|\beta^{m+1} - \beta\| \leq \sqrt{s} + \sqrt{s}\lambda_{m+1} + 6\delta\sqrt{s}\lambda_m \leq \sqrt{s}\lambda_{m+1}(2 + 6\sqrt{\delta}) \leq 3\sqrt{s}\lambda_{m+1}.$$

Théorème 11 *:*

Sous les hypothèses du théorème 10, si en plus la fonction objective à minimiser : f, est convexe et vérifie les différentes hypothèses suivantes : Compressibilité du gradient, sparsité faible de la Hessienne, l'existence du minimum, si en plus le gradient est lipschitz et la fonction répond aux conditions de coércivité, en fixant le pas de la descente de gradient tel que : $\alpha = 1/L$ alors : avec probabilité égale à $1 - 2(\frac{s}{d})^{\tilde{b}_2}$ l'algorithme trouve une solution ϵ-opptimale en utilisant $\frac{4\tilde{b}_1 slog(d)LR^2}{\epsilon(1-\epsilon_{rel})}$, $\forall \epsilon > R\sqrt{9s\lambda_0^2/(1-\epsilon_{rel})}$ avec $0 < \epsilon_{rel} < 1$. Comme ce résultat dépend du choix de λ_0, il est essentiel de sélectionner cet hyperparamètre de manière optimale, comme décrit dans [21].

Preuve :
Sous les hypothèses du théorème 10, d'après le théorème 1 l'estimateur du gradient produit par la procédure de l'IHT adaptatif $(\hat{g}_k)$ vérifie :

$$\forall k, \|g - \hat{g}_k\|_2^2 \leq \epsilon_{rel}\|g_k\|_2^2 + \epsilon_{abs}.$$

Avec $0 < \epsilon_{rel} < 1$ et $\epsilon_{abs} := 9s\lambda_0^2$, par définition de la suite (λ_k). Alors $\forall \epsilon > R\sqrt{9s\lambda_0^2/(1-\epsilon_{rel})}$, on peut résoudre l'inéquation $e_k \leq \epsilon$. On utilise le lemme 2 :

$$k \leq \frac{4LR^2}{\epsilon(1-\epsilon_{rel})} \leq \frac{4LR^2}{\epsilon(1-\epsilon_{rel})} - \frac{4LR^2}{e_0(1-\epsilon_{rel})}.$$

Pour obtenir le résultat final on multiplie cette inégalité par le nombre de reqêtes nécessaires pour le fonctionnement de l'algorithme ZORO à chaque itération, à savoir $4b_1 log(d)$.

7 Autres pistes de réflexion

Pendant ce stage, en plus d'essayer d'améliorer la procédure d'estimation du gradient en proposant une alternative adaptative, nous avons également tenté d'améliorer la procédure d'optimisation. Pour ce faire, nous avons envisagé de remplacer la descente de gradient classique par une descente de gradient miroir, également connue sous le nom de Miror Gradient Descent.

Comme le montre la figure(7), cette alternative s'avère particulièrement efficace lorsque l'espace dans lequel se trouve le minimum est connu. Comme illustré dans la figure (7), dans ce cas, la descente de gradient miroir permet de trouver une solution de meilleure qualité en moins de temps que la descente de gradient classique. Cependant, dans le problème considéré, il n'est pas possible de connaître cette information a priori. Dans ce scénario, la descente de gradient miroir présente un inconvénient majeur : elle peut diverger, contrairement à la descente de gradient classique.

Cette divergence s'explique par le fait que la descente de gradient miroir combine une procédure de descente de gradient classique avec une projection du point trouvé sur un ensemble prédéfini. Ainsi, lorsque l'espace du minimum est précisé, la convergence est plus rapide et plus précise. Cependant, en l'absence de cette information, la divergence peut se produire.

8 Conclusion

En conclusion, nous avons réussi à apporter une réponse positive à la problématique qui a guidé ce stage depuis son début comme le montrent les figures (5,6). Nous avons développé une procédure, présentée dans la section 7, qui nous permet d'obtenir un algorithme adaptatif capable de résoudre efficacement les problèmes d'optimisation d'ordre zéro. Les résultats théoriques de cette approche se rapprochent de ceux obtenus dans l'article de référence [9], sur lequel notre étude est basée, avec une légère sous-optimalité due à notre volonté d'obtenir une approche adaptative.

Nos études empiriques ont également montré que lorsque toutes les hypothèses sont vérifiées, notre approche adaptative peut rivaliser avec le ZORO classique. Cependant, nous reconnaissons que des améliorations supplémentaires sont possibles, en particulier dans la partie de l'optimisation car notre étude a grandement porté sur l'estimation du gradient. Il serait envisageable de poursuivre nos efforts pour améliorer davantage ces procédures, en explorant des méthodes d'optimisation plus complexes, telles que la méthode de Broyden-Fletcher-Goldfarb-Shanno (BFGS) ou L-BFGS. Nous avons choisi de laisser ces perspectives d'amélioration pour de futures études.

Références

[1] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Mike Rabbat. Stochastic gradient push for distributed deep learning. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, Proceedings of the 36th International Conference on Machine Learning, volume 97 of Proceedings of Machine Learning Research, pages 344–353. PMLR, 09–15 Jun 2019.

[2] Ran Xin, Anit Kumar Sahu, Usman A. Khan, and Soummya Kar. Distributed stochastic optimization with gradient tracking over strongly-connected networks. In 2019 IEEE 58th Conference on Decision and Control (CDC), pages 8353–8358, 2019.

[3] Alison L. Marsden, Jeffrey A. Feinstein, and Charles A. Taylor. A computational framework for derivative-free optimization of cardiovascular geometries. Computer Methods in Applied Mechanics and Engineering, 197(21) :1890–1905, 2008.

[4] Genetha Gray, Tamara Kolda, Kenneth Sale, and Malin Young. Optimizing an empirical scoring function for transmembrane protein structure determination. INFORMS Journal on Computing, 16 :406–418, 11 2004.

[5] Pengcheng He, Siyuan Lu, Xin Guan, Yibin Kang, and Qingjiang Shi. A zeroth-order block coordinate gradient descent method for cellular network optimization. In 2022 International Symposium on Wireless Communication Systems (ISWCS), pages 1–6, 2022.

[6] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks : Reliable attacks against black-box machine learning models, 2017.

[7] Pin-Yu Chen, Huan Zhang, Yash Sharma, Jinfeng Yi, and Cho-Jui Hsieh. Zoo. Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, Nov 2017.

[8] Yining Wang, Simon Du, Sivaraman Balakrishnan, and Aarti Singh. Stochastic zeroth-order optimization in high dimensions, 2018.

[9] HanQin Cai, Daniel McKenzie, Wotao Yin, and Zhenliang Zhang. Zeroth-order regularized optimization (zoro) : Approximately sparse gradients and adaptive sampling. SIAM Journal on Optimization, 32(2) :687–714, Apr 2022.

[10] John C. Duchi, Michael I. Jordan, Martin J. Wainwright, and Andre Wibisono. Optimal rates for zero-order convex optimization : the power of two function evaluations, 2013.

[11] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. Foundations of Computational Mathematics, 17(2) :527–566, Apr 2017.

[12] D. Needell and J. A. Tropp. Cosamp : Iterative signal recovery from incomplete and inaccurate samples, 2008.

[13] Richard Baraniuk, Mark Davenport, Ronald DeVore, and Michael Wakin. A simple proof of the restricted isometry property for random matrices. Constructive Approximation, 28 :253–263, 12 2008.

[14] Simon Foucart. Sparse recovery algorithms : Sufficient conditions in terms of restrictedisometry constants. In Marian Neamtu and Larry Schumaker, editors, Approximation Theory XIII : San Antonio 2010, pages 65–77, New York, NY, 2012. Springer New York.

[15] J. Kiefer and J. Wolfowitz. Stochastic Estimation of the Maximum of a Regression Function. The Annals of Mathematical Statistics, 23(3) :462 – 466, 1952.

[16] Yining Wang, Simon Du, Sivaraman Balakrishnan, and Aarti Singh. Stochastic zeroth-order optimization in high dimensions, 2017.

[17] Afonso S. Bandeira, Katya Scheinberg, and Luis Nunes Vicente. Computation of sparse low degree interpolating polynomials and their application to derivative-free optimization, 2013.

[18] Afonso S. Bandeira, Katya Scheinberg, and Luis Nunes Vicente. Convergence of trust-region methods based on probabilistic models, 2013.

[19] Alekh Agarwal, Ofer Dekel, and Lin Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. pages 28–40, 12 2010.

[20] D. Needell and J. A. Tropp. Cosamp : Iterative signal recovery from incomplete and inaccurate samples, 2008.

[21] Mohamed Ndaoud. Scaled minimax optimality in high-dimensional linear regression : A non-convex algorithmic regularization approach, 2020.

[22] Thomas Blumensath and Mike E. Davies. Iterative thresholding for sparse approximations. Journal of Fourier Analysis and Applications, 14(5) :629–654, Dec 2008.

[23] Akira Shinkyu and Naoya Sueishi. Small tuning parameter selection for the debiased lasso, 2022.

A Le biais du lasso dé-biaisé

Consiédrons le modèle suivant :

$$Y = X\beta_0 + \epsilon \tag{29}$$

Le principal problème associé à l'estimateur Lasso réside dans la présence d'un biais, qui peut certes être réduit, mais dont l'élimination totale demeure hors de portée. L'estimateur du Lasso classique cherche à résoudre le problème suivant :

$$\hat{\beta} := argmin_{\beta \in R^d} \frac{1}{n} \|Y - X\beta\|_2^2 + 2\lambda\|\beta\|_1 \tag{30}$$

Comme mentionné plus haut il est possible de limiter le biais de l'estimateur Lasso, on définit ainsi l'estiamteur suivant qui est la version dé-biasié d'un estimateur donné, disons β_{01} :

$$\tilde{\beta} := \hat{\beta}_1 + \frac{1}{n}\hat{\Theta}_1^T X^T (Y - X\hat{\beta}) \tag{31}$$

Où β_1 représente la première composante de β, donc l'estimateur associé à la constante dans la régression. Par ailleurs, on n note $\Sigma := E(X^T X/n)$ et Θ_1 représente la première colonne de Σ^{-1}. Ainsi $\hat{\Theta}_1$ est un estimateur de Θ_1. Généralement on utilise le node-wise-Lasso pour construire $\hat{\Theta}_1$.

En effet la regression node-wise est définit de la manière suivante :

$$\hat{\gamma} := argmin_{\gamma \in R^{d-1}} \frac{1}{n} \|X_1 - X_{-1}\gamma\|_2^2 + 2\mu\|\gamma\|_1 \tag{32}$$

Où X_{-1} représente la sous-matrice obtenue en éliminant la colonne X_1 de X. Soit τ la variance de l'erreur dans la régression de X_1 sur X_{-1}. Soient : $\tilde{\tau} := \frac{1}{n}\|X_1 - X{-1}\hat{\gamma}\|_2^2$ et $\hat{\tau} := \tilde{\tau}^2 + \mu\|\hat{\gamma}\|_1$. Les conditions du Karush-Kuhn–Tucker(KKT) dans le problème de la regression node-wise, impliquent [23] : $\hat{\tau} = X_1^T(X_1 - X_1\hat{\gamma})/n$. Ainsi on peut estimer Θ_1 par :[23]

$$\hat{\Theta}_1 := \frac{1}{\hat{\tau}^2} \begin{bmatrix} 1 \\ -\hat{\gamma}^T \end{bmatrix} \tag{33}$$

Par ailleurs, $\sqrt{n}(\tilde{\beta} - \beta_{01}) := \frac{1}{\sqrt{n}}\hat{\Theta}_1^T X^T \epsilon + \Delta_1$, où $\Delta_1 := \sqrt{n}(\hat{\Sigma}\Theta_1 - e_1)^T(\beta_0 - \hat{\beta})$. D'après l'inégalité $l_1 - l_\infty$ de Hölder : [23]

$$|\Delta_1| \leq \sqrt{n}\|\hat{\Sigma}\hat{\Theta}_1 - e_1\|_\infty \|\hat{\beta} - \beta_0\|_1 \leq \sqrt{n}\frac{\mu}{\hat{\tau}^2}\|\hat{\beta} - \beta_0\|_1 \tag{34}$$

Sous certaines conditions, l'estimateur classique du Lasso vérifie $\|\hat{\beta} - \beta_0\| = \mathcal{O}_p(s\sqrt{log\ \mathrm{p/n}}$,où $s := \|\beta_0\|_0$ et le taux de convergence ne peut pas être améliorer. Ainsi le biais de cet estimateur dépend largement de $\frac{\mu}{\hat{\tau}^2}$.

B Bandit gradient descent algorithm

Algorithme 8 : BGD

Données d'entrée : S, S_i, ν, T, y_0

for t=1 **To** T **do**

- $u_t \leftarrow$ **Realisation d'une variable uniforme sur la sphère** S_i;
- $x_t \leftarrow y_t + \delta u_t$;
- $c_t := \frac{E_{f_{\hat{S}_i}}(x_t+\delta u_t) - E_{f_{\hat{S}_i}}(x_t)}{\delta}$;
- $y_{t+1} := P_S(y_t - \nu c_t | \hat{S}_i | u_t)$;

return y_T.

C Figures

C.1 Comaraison entre le CoSaMP et le lasso Dé-biaisé

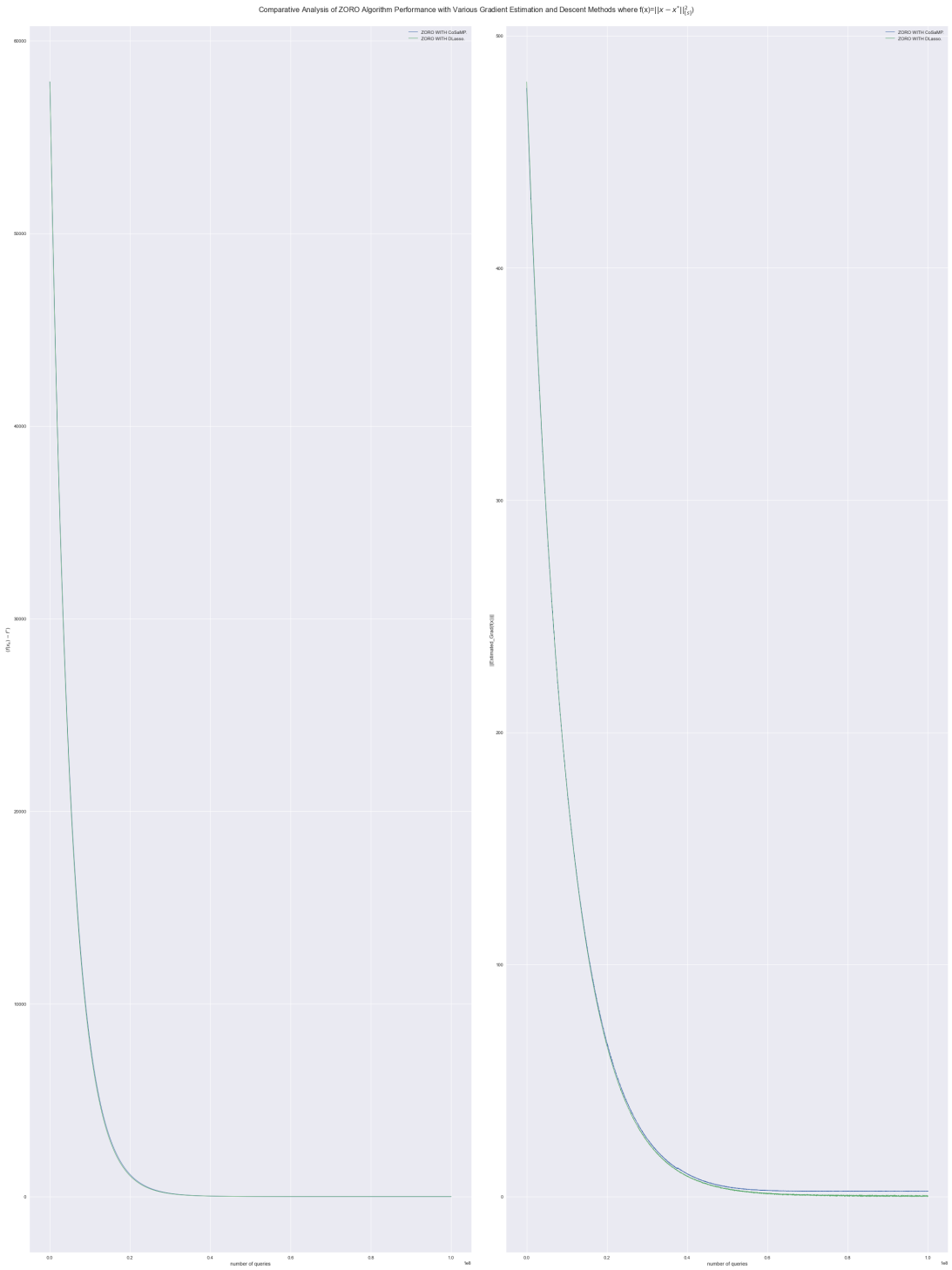

FIGURE 1 – Comportement général des deux algorithmes CoSaMP et DLasso pour un budget égal à 1e8.

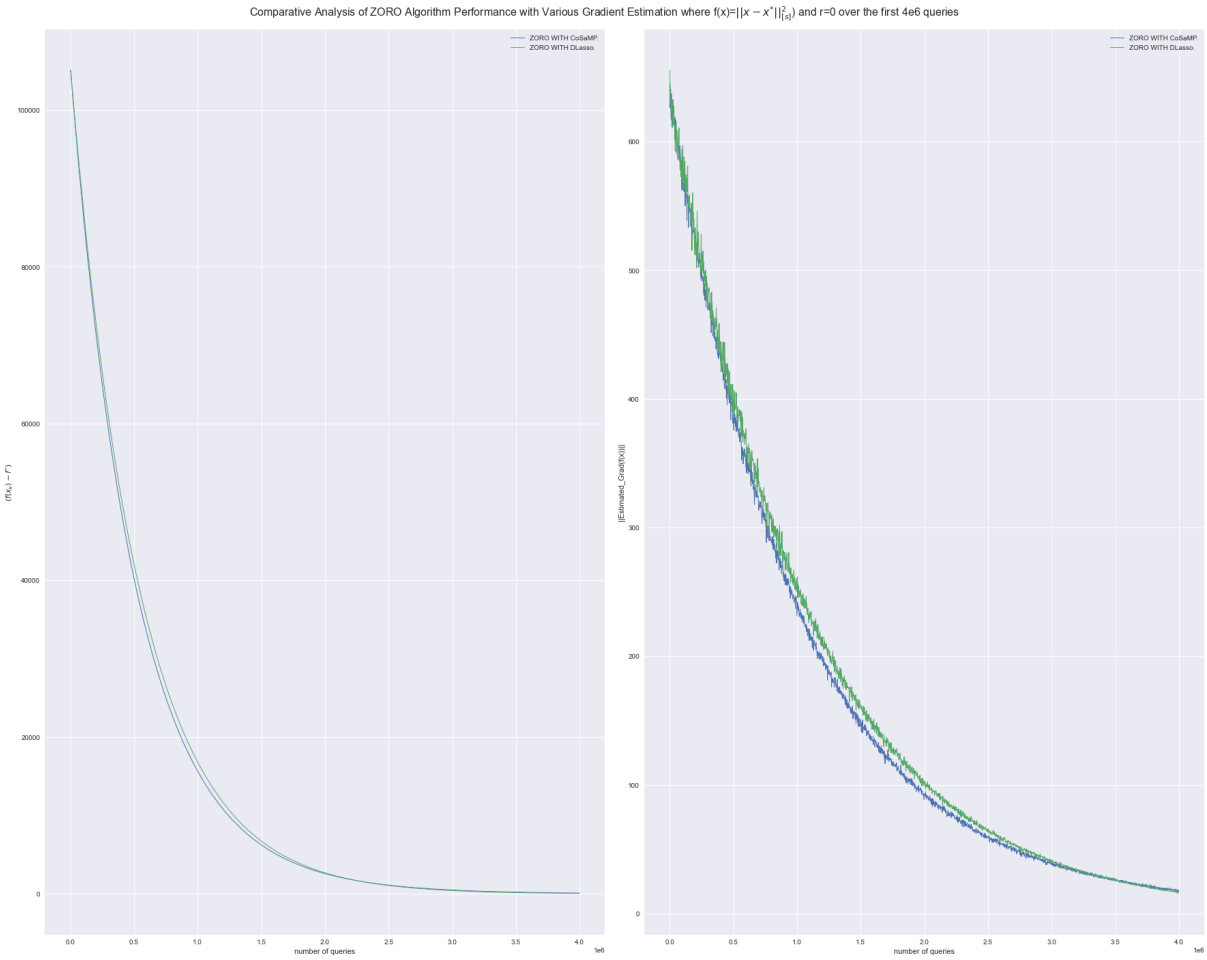

FIGURE 2 – Comportement général des deux algorithmes CoSaMP et DLasso sur la première partie du budget.

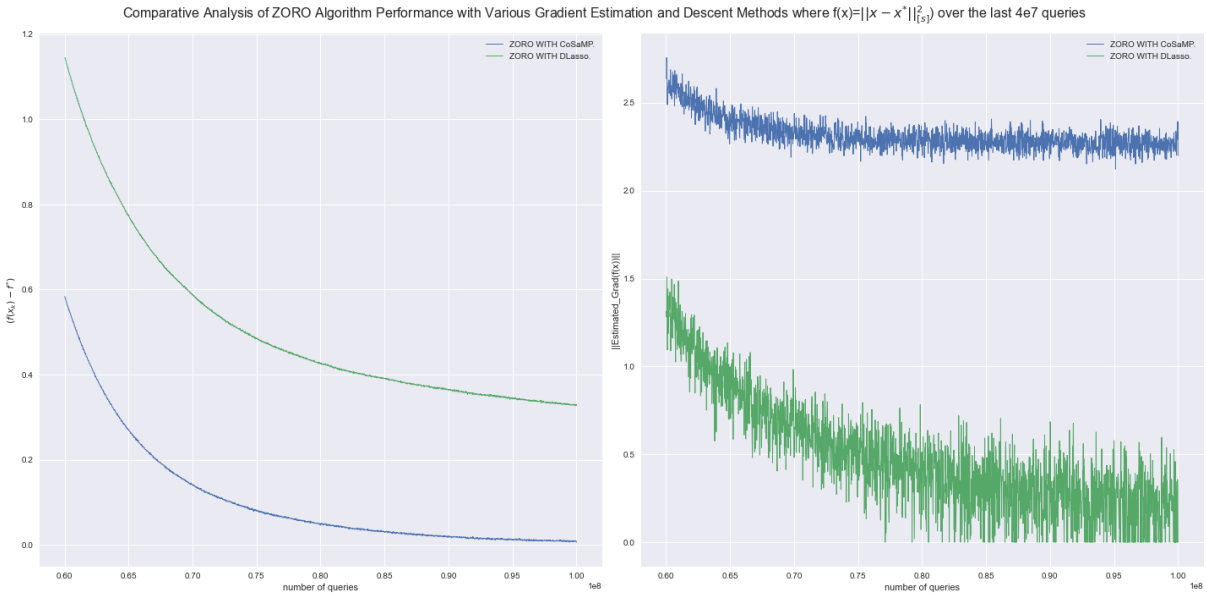

FIGURE 3 – Comportement général des deux algorithmes CoSaMP et DLasso sur la dernière partie du budget.

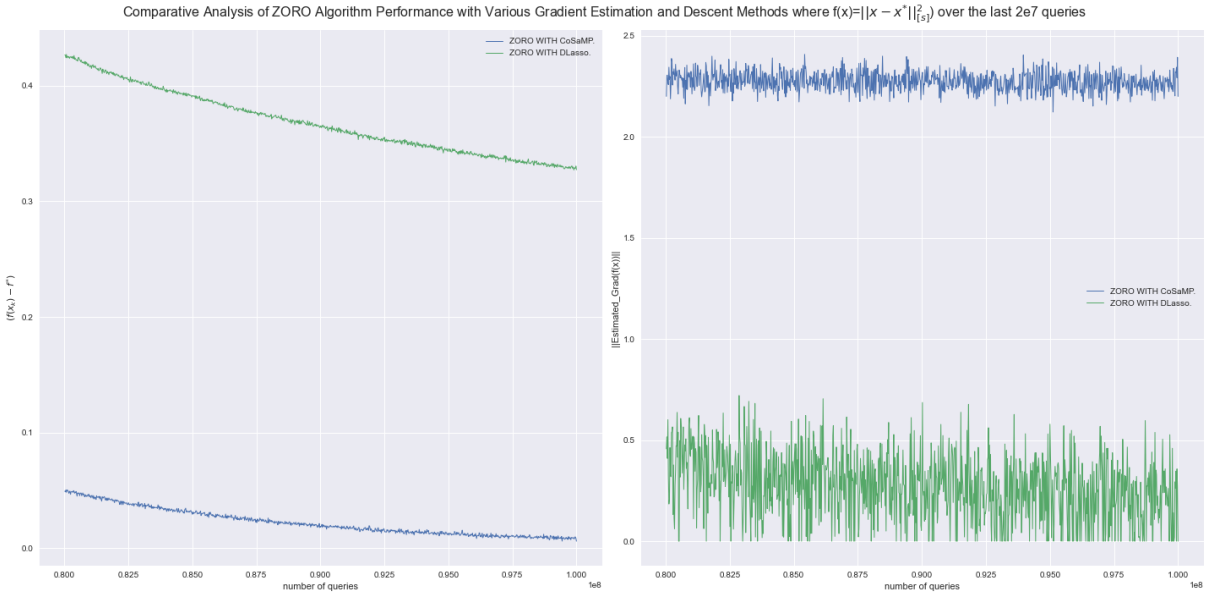

FIGURE 4 – Effet du biais du Lasso dé-biaisé par rapport au CoSaMP.

C.2 Comportement de la version adaptative de l'IHT

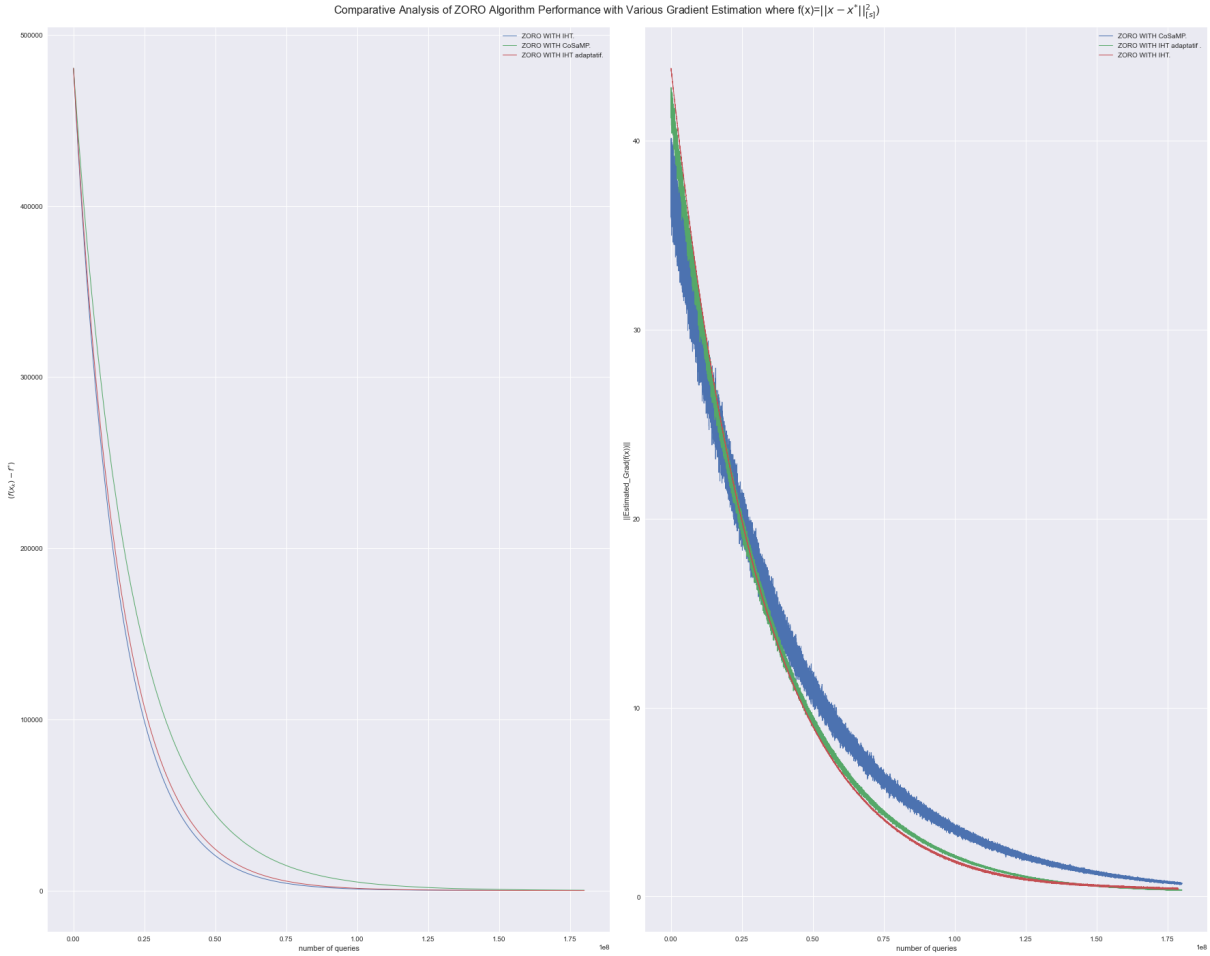

FIGURE 5 – Comportement général de la version adaptatif vis-à-vis l'IHT et le CoSaMP.

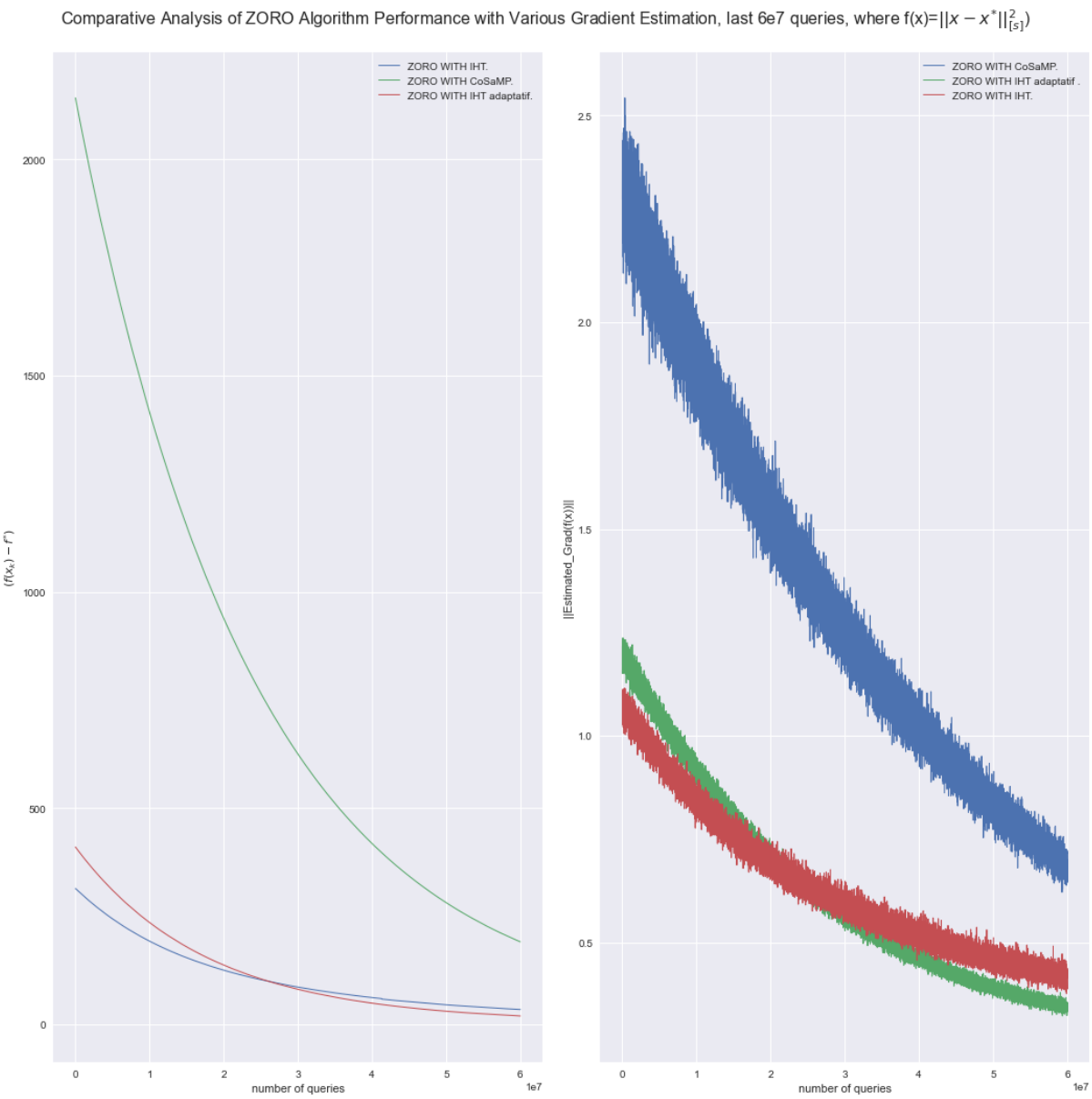

FIGURE 6 – Comportement général de la version adaptatif vis-à-vis l'IHT et le CoSaMP sur la dernière partie du budget.

C.3 Tentative d'améliorer la procédure d'optimisation

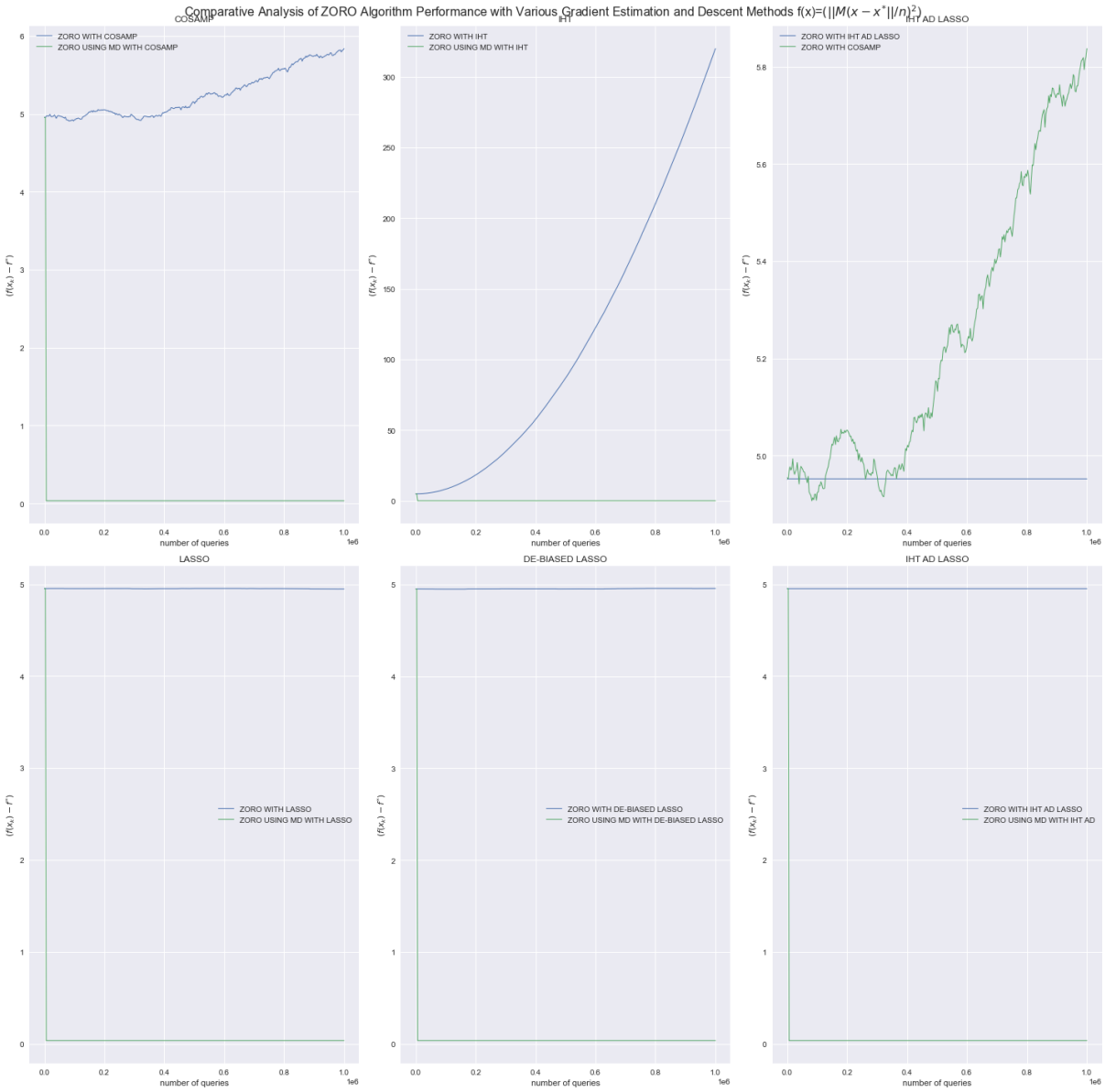

FIGURE 7 – Le comportement de la descente de gradient miroir