COCO Denoiser: Using Co-Coercivity for Variance Reduction in Stochastic Convex Optimization

Manuel Madeira
Instituto Superior Técnico
Lisbon, Portugal

ABSTRACT

First-order methods for stochastic optimization gained undeniable relevance, in particular due to their pivotal role in machine learning. These methods blindly accept noisy gradients provided by an oracle, which may be inconsistent with structural properties of the underlying objective function. We exploit gradient co-coercivity of L-smooth convex objective functions to obtain more accurate gradient estimates. The method introduced in this thesis is then coined as the co-coercivity (COCO) denoiser.

Our denoiser is a joint Maximum Likelihood (ML) estimator for the gradients, constrained by pairwise co-coercivity conditions. Although ML leads to a Quadratically Constrained Quadratic Problem, we introduce an efficient first-order algorithm for COCO, which is based on the Fast Dual Proximal Gradient method. For the denoiser that deals with a single pair of gradients, we derive the closed-form solution for the ML problem, which is relevant in practice, since our experiments have shown that even this simple scenario leads to variance reduction gains in stochastic optimization.

We carry out a theoretical analysis that provides insight into parameter tuning and estimator results, showing in particular why COCO necessarily improves with respect to the noisy oracle. The experimental analysis corroborates these results and shows that the COCO estimator, although not unbiased, leads to a reduction of the mean squared error of the function gradients. To illustrate the impact in stochastic optimization, we use both synthetic data and a real online learning task. Our experiments show that COCO leads to improvements in variance reduction with respect to baseline algorithms.

KEYWORDS

Stochastic Optimization; First-Order Algorithms; Convex Optimization; Co-coercivity; COCO Denoiser; Variance Reduction; Quadratically Constrained Quadratic Problem; Machine Learning.

1 INTRODUCTION

Nowadays, *mathematical optimization* [1] is recognized as a pivotal tool not only for phenomena description in science but also when rationalizing the process of decision making. In this thesis, we focus on convex optimization problems, given their strong theoretical guarantees and wide range of applications. A common approach to solve these problems are the first-order algorithms, which can be defined as iterative methods which only use first-order derivatives (in the multivariate case, gradients), requiring then that the objective function must be differentiable. In spite of exhibiting slower convergence rate to the optimal solution, these algorithms present a much cheaper cost per iteration when compared to alternative approaches. For this reason, the first-order algorithms are the usual choice when addressing high-dimensional convex optimization problems, scenario which is common in, for example, machine learning or signal and image processing.

Typically, there are two different settings through which the gradient estimates are provided: the deterministic (*e.g.*, in the Gradient Descent (GD) algorithm), where that estimate corresponds to the true gradient of the objective function at that point, and the stochastic (*e.g.*, in the Stochastic Gradient Descent (SGD) algorithm), where we only have access to a noisy version of the true gradient. Our goal in this thesis is to evaluate the possibility of further improving the performance of the state-of-the-art stochastic optimization algorithms by providing them cleaner gradient estimates. These gradient estimates will be obtained by exploiting properties of convex functions that, although widely used for convex optimization algorithm analysis, have been left out of algorithm design.

Two typical structural properties of many convex functions are the L-smoothness and (μ-)strong convexity. The relevance of strong convexity and L-smoothness is clear: while strong convexity imposes that in every point there is at least some curvature of the function, L-smoothness grants that that curvature can not be arbitrarily high. These properties have shown to be very useful for the analysis of convex optimization algorithms without excessively restricting the convex settings in which they can be verified, as they are still very general [2]. However, to the best of our knowledge, existing algorithms for stochastic convex optimization do not make explicit use of this kind of restrictions to the function curvature, naively accepting the noisy gradients provided by the oracle. This motivated us to exploit L-smoothness (usually considered an even weaker assumption than strong convexity [3]) and mere convexity (which can be seen as a relaxation of strong-convexity to $\mu = 0$) to constrain function gradients. It has been shown that L-smoothness and convexity can be merged into one single condition, usually referred to as gradient *co-coercivity* [4], which suggested coining our method as the *COCO denoiser*.

We formulate the denoising problem as the joint ML estimation of a set of function gradients, constrained by the co-coercivity conditions, from their noisy observations provided by the oracle, where, as often done, the noise is assumed to be zero mean white Gaussian. Our workflow for stochastic optimization, schematically represented in Figure 1, consists in using COCO as a "plug-in" denoiser, *i.e.*, in feeding the denoised gradients to a baseline algorithm rather that letting it consult directly the oracle.

The ML estimation in the COCO denoiser leads to a particular convex optimization problem known as Quadratically Constrained Quadratic Problem (QCQP). This problem can be solved by using available convex optimization packages, such as the popular

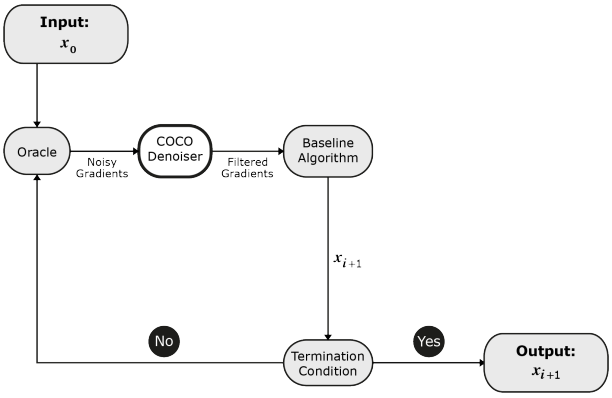

Figure 1: Our workflow for stochastic optimization with first-order algorithms. The approach can be interpreted as using a "new oracle", composed by the original one coupled with the proposed COCO denoiser.

CVX [5], but its complexity may turn out prohibitive. For this reason, we exploit the particular structure of the ML estimation problem to derive an original first-order algorithm, based on the so-called Fast Dual Proximal Gradient (FDPG) method [6], which succeeds in providing an approximate solution in reasonable time.

In spite of the efficiency of the proposed algorithm, the number of co-coercivity constraints grows quadratically with the number of points simultaneously processed by COCO. Thus, although cleaner gradient estimates are obtained by processing simultaneously the ever growing number of points visited, *i.e.*, points $x_0, \ldots, x_i$, in Figure 1, we also consider performing the denoising using just a fixed number K of last visited points, *i.e.*, $x_{i-K+1}, \ldots, x_i$. For shortness of reference, we call this denoiser COCO_K (naturally, the globally optimal denoiser is then COCO_{i+1}). This point deserves particular attention because our experiments have shown that convergence gains in stochastic optimization are obtained with values of K as small as 2 and we are able to find the *closed-form solution* for COCO_2.

By theoretically analysing COCO, we are able to provide insight regarding the estimator results, showing in particular why COCO necessarily improves with respect to the noisy oracle. We also evaluate the probability of the co-coercive constraints being "active", which enables interpreting the impact of the chosen Lipschitz constant L in our approach. Our experimental analysis corroborates these results and show that the COCO estimator, although not unbiased, leads to a reduction of the MSE of the function gradients, as desired.

In the context of stochastic optimization, to evaluate the impact of using the proposed COCO denoiser, we consider two scenarios. Firstly, using synthetic data that follow the assumptions underlying the design of the denoiser. Secondly, using an online learning task (logistic regression [3]), in which the noise affecting the gradients falls out of those assumptions. Our experiments have shown that COCO leads to improvements in variance reduction with respect to baseline algorithms such as SGD and Adam.

We emphasize the following aspects as original contributions of the work in the thesis:

- Exploration of L-smoothness and convexity (*i.e.*, co-coercivity) in the context of the Maximum Likelihood estimation of function gradients from noisy observations, leading to the COCO denoiser;
- Efficient first-order solution method for COCO (FDPG);
- Closed-form solution for COCO_2;
- Analytic study of COCO, providing insights into parameter tuning and expected error reduction;
- Experimental analysis of COCO, regarding estimator bias and mean squared error;
- Framework for stochastic optimization using first-order algorithms with the COCO denoiser;
- Experiments illustrating variance reduction in convex stochastic optimization using COCO.

The bibliographic review presented in the thesis also deserves mention, in particular due to the summary of asymptotically optimal convergence rates, which is not conveyed in such condensed form even in recent surveys. The main original results of this work will also appear in [7, 8].

2 OVERVIEW OF CURRENT APPROACHES

The problem we motivated in the previous section lives in the field of stochastic optimization. Although the machine learning frenzy of the last few years has strongly contributed to the development and enhancement of algorithms like SGD, the first approaches to stochastic optimization date back to the fifties of the last century [9]. Here, we single out results that inspire the methods proposed in the thesis. The scenario is simply described: the objective function f is unknown but can be accessed through queries to a first-order *oracle*, *i.e.*, an unit that takes as input a vector x and outputs the gradient $\nabla f(x)$[10]. While GD uses an exact oracle, SGD has to deal with an inexact one.

Unlike with GD, a fixed step size for SGD does not make it converge to the optimal solution, even in the convex setting. In fact, the convergence of SGD can be analyzed considering two terms: (i) the bias term, which represents the dependence of the convergence on the initial distance to the optimum (*e.g.*, $\|f(x_0) - f(x^*)\|$), and (ii) the variance term, which represents the dependence of the convergence on the noise of the oracle itself. Although the bias term vanishes under a convenient selection of a fixed step size, that does not happen to the variance one. For this reason, the algorithm gets stuck when the bias term has vanished, originating random iterates within a certain "ball of uncertainty".

This problem in the stochastic setting can be solved by using a diminishing step size, *i.e,*, by selecting at iteration i a step size $\gamma_i = C/i$, where C is constant. Basically, this strategy progressively reduces the referred ball of uncertainty, enabling convergence. It can be shown that the convergence rates obtained using this approach are asymptotically optimal when the expected value of the oracle equals the true gradient (*e.g.*, whenever the noise affecting the gradients is additive and zero mean) [11, 12]. In spite of this, further significant improvements were achieved through online averaging (the so-called Polyak-Ruppert averaging [13]) and adaptive step size techniques, *e.g.*, Adam [14]. Naturally, these improvements in constants on the convergence rate are of utmost importance in

practice, where one necessarily deals with a finite horizon in terms of number of iterations.

In the last decade, a new direction has driven the research community to a paradigm that enhances the aforementioned asymptotic bounds. For objectives that can be decomposed as a finite sum of functions, which is precisely the case of many machine learning problems, the so-called variance reduction (VR) techniques significantly improve the performance of the optimization algorithms. In fact, the rise of algorithms such as the Stochastic Average Gradient (SAG) [15] made it possible to close the gap that existed between the asymptotic convergence rate gap of the deterministic GD and the ones of all the stochastic oracle counterparts.

Nevertheless, in the deterministic scenario, it is also known that GD is sub-optimal among the methods that perform each step still using only gradient information but obtained at more than one point [2]. In fact, considering the class of algorithms that generate each iterate using a linear combination of the gradients of all previously visited locations, *i.e.*, such that $x_i \in x_0 + \text{span}\{\nabla f(x_0), \ldots, \nabla f(x_{i-1})\}$, the work developed by Polyak, with its heavy-ball method [16], then refined by Nesterov [17], lead to the so-called Nesterov Accelerated Gradient (NAG) descent algorithm, which achieves the optimal rate. In the stochastic scenario, the optimal rates were naturally shown to be worse [18, 19] and are attained by a family of stochastic accelerated algorithms, first introduced by the reference [20].

Table 1 summarizes the different asymptotically optimal convergence rates for each family of algorithms mentioned throughout this section.

Table 1: Asymptotic optimal convergence rates ($\mathbb{E}[f(x_i) - f(x^*)]$) for algorithms GD, NAG, SGD (which includes averaging and adaptive methods), VR (only for finite sums) and ACC (only for finite sums), under different assumptions on the objective function, f. See thesis for the meaning of constants k, $k_{\max}$, and n.

	Deterministic		**Stochastic**		
Assumption(s)	GD	NAG	SGD	VR	ACC
Convexity	$O\left(\frac{1}{\sqrt{i}}\right)$	$O\left(\frac{1}{\sqrt{i}}\right)$	$O\left(\frac{1}{\sqrt{i}}\right)$	$O\left(\frac{1}{\sqrt{i}}\right)$	$O\left(\frac{1}{\sqrt{i}}\right)$
+ L-Smoothness	$O\left(\frac{1}{i}\right)$	$O\left(\frac{1}{i^2}\right)$	$O\left(\frac{1}{\sqrt{i}}\right)$	$O\left(\frac{1}{i}\right)$	$O\left(\frac{1}{i^2}\right)$
+ Strong Convexity	$O\left(e^{-\frac{i}{k}}\right)$	$O\left(e^{-\frac{i}{\sqrt{k}}}\right)$	$O\left(\frac{1}{i}\right)$	$O\left(e^{-\frac{i}{k_{\max}+n}}\right)$	$O\left(e^{-\frac{i}{\sqrt{nk_{\max}}+n}}\right)$

3 COCO DENOISER

In this section, we describe our approach. First, we formulate COCO as a Maximum Likelihood estimator constrained by co-coercivity conditions; then, we propose efficient methods to compute its solution. Finally, we study theoretical properties of the estimator.

3.1 Maximum Likelihood Estimation

Let $f : \mathbb{R}^d \to \mathbb{R}$ be a convex and L-smooth function (the reader might remind that the definition of L-smoothness does not assume convexity). A standard result in convex analysis is that the gradient of f is co-coercive [1], *i.e.*,:

$$\begin{aligned} &\forall x, y \in \mathbb{R}^n,\ L \in \mathbb{R}^+ : \\ &\frac{1}{L}\|\nabla f(y) - \nabla f(x)\|^2 \leq \langle \nabla f(y) - \nabla f(x),\ y - x \rangle . \end{aligned} \tag{1}$$

Note that Equation (1) is stronger than the inequality in L-smoothness definition, since the inequality in that definition follows from Equation (1). This fact can be easily observed by applying the Cauchy-Schwartz inequality on the right-hand side of Equation (1).

In our approach, despite not knowing the objective function f, we assume the following:

Assumption 3.1. *We know a Lipschitz constant of its gradient, L.*

This assumption is commonly adopted in stochastic optimization methods as a result of its usefulness in the analysis of those algorithms, without narrowing excessively the universe of possible applications. This imposition prevents the gradients from changing arbitrarily fast from one point to another. Moreover, L-smoothness is usually considered a weaker assumption than strong convexity [3]. In fact, as a consequence of their high-dimensionality, typical machine learning problems have correlated variables, yielding non-strongly convex objective functions (*i.e.*, $\mu \approx 0$) [21]. By additionally considering that estimating L is often easier than estimating μ, we emphasize the pertinence of this assumption.

Assumption 3.2. *We have access to an oracle which, given an input $x \in \mathbb{R}^d$, outputs a noisy version of the gradient of f at x; specifically, we assume that the oracle outputs $g(x; w) = \nabla f(x) + w$, where $w \in \mathbb{R}^d$ is a sample of a Gaussian distribution with zero mean, i.e., $w \sim \mathcal{N}(0, \Sigma)$, where the covariance matrix Σ is assumed to be known. Moreover, we assume the noise samples are independent across the oracle consultations.*

The motivation for the noise model comes, naturally, from the simplicity that it provides to our method. Moreover, the Central Limit Theorem states that the sum (or mean) of a given number of independent and identically distributed random variables with finite variances will tend to a normal distribution as the number of variables grows. Note that in machine learning, the mini-batch scheme is a common procedure to obtain gradient estimates at a point and its gradient estimator is defined as a mean of independent random variables. This reasoning reinforces the relevance of this assumption.

We assume that the oracle was consulted at the input points $x_1, \ldots, x_K$, and returned the outputs $g_1, \ldots, g_K$. This is our available data, which we arrange in the vector $g = [g_1,\ \ldots,\ g_K]^T \in \mathbb{R}^{Kd}$. We address the problem of estimating the gradients $\nabla f(x_1),\ \ldots,\ \nabla f(x_K)$ from the available data. Thus, the parameter we're interested in estimating is $\theta = [\theta_1,\ \ldots,\ \theta_K]^T \in \mathbb{R}^{Kd}$, where $\theta_k = \nabla f(x_k) \in \mathbb{R}^d$.

From Assumption 3.2, the available data g is related to the parameter of interest θ by the observation model

$$g = \theta + w, \tag{2}$$

where $w = [w_1,\ \ldots,\ w_K]^T \in \mathbb{R}^{Kd}$ is distributed as $w \sim \mathcal{N}(0, \Sigma_w)$. Note that Σ_w is a block-diagonal matrix, each block being Σ.

We also have the following information about the parameter θ, which comes from the co-coercivity condition (Equation (1)):

$$\theta \in \Theta = \{(\theta_1, \ldots, \theta_K) : \frac{1}{L}\|\theta_m - \theta_l\|^2 \leq \langle \theta_m - \theta_l,\ x_m - x_l \rangle, \ 1 \leq m < l \leq K\}\,.$$

The Maximum Likelihood (ML) estimate of θ is

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \quad p(g|\theta)\,,$$

where, in accordance to our observation model in Equation (2),

$$p(g|\theta) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(g-\theta)^T \Sigma_w^{-1}(g-\theta)}\,.$$

Consequently, it is immediate that computing our solution $\hat{\theta}$ corresponds to solving the following optimization problem:

$$\begin{aligned} \underset{\theta_1,\ldots,\theta_K}{\text{minimize}} \quad & \sum_{k=1}^{K} (g_k - \theta_k)^T \Sigma^{-1} (g_k - \theta_k) \\ \text{subject to} \quad & \frac{1}{L}\|\theta_m - \theta_l\|^2 \leq \langle \theta_m - \theta_l, x_m - x_l \rangle, \\ & 1 \leq m < l \leq K. \end{aligned} \tag{3}$$

In fact, this instantiation of the convex optimization problem can be classified as a QCQP [1], since both the objective and the constraints are quadratic functions. In this type of problem, the objective function (quadratic) is minimized over a feasible region that results from the intersection of ellipsoids. Despite their ubiquity in many engineering and scientific applications, solving a generic nonconvex QCQP problem is NP-hard [22]. Nevertheless, for convex instances of those problems, it is possible to explore the structure of the problem and attain a tractable solution method. This is the case of the problem considered and the methods proposed are discussed below.

The main idea supporting our approach is the observation that all the methods mentioned in Section 2, even though often assuming the L-smoothness of the objective function for their analyses, do not take advantage of it in their methodology. In fact, after taking this assumption into consideration, it is unsatisfactory to blindly accept the gradients that the oracle is outputting. It is on the demand of making those observations coherent with this assumption that our approach is based on. Since the co-coercivity constraints play the pivotal role of merging two important conditions (convexity and L-smoothness of the objective function) into only one expression, we call our method the COCO denoiser.

The number of constraints in this approach scales quadratically with K, the number of consulted points. More precisely, the number of constraints is given by $K(K-1)/2$, as each constraint is imposed between every two points from the ones considered. This drawback motivates a simplification of the original COCO denoiser: instead of considering all the consulted points across all the i iterations, we fix a given number of points, K ($1 \leq K \leq i$), and only consider the information belonging to the last K points to denoise the consulted gradients. For example, if we fix $K = 2$, the denoise is performed only considering x_i, x_{i-1}, g_i and g_{i-1}. To this denoiser using a fixed window of length K, we call COCO_K.

3.2 Efficient Solutions for COCO_K

In this section, a solution method is proposed for the QCQP raised by COCO_K. We start by providing closed-form solutions for $K = 1$ and $K = 2$ and then propose an iterative algorithm which yields an approximate solution for arbitrary K.

The closed-form solutions for $K = 1$ and $K = 2$ can be achieved by instantiating the well-known KKT conditions for the QCQP we formulated. These results are provided in Theorem 3.1 and Theorem 3.2.

Theorem 3.1. *The solution to Equation* (3) *for $K = 1$ is given by:*

$$\hat{\theta}_1 = g_1\,. \tag{4}$$

Note that the result for Theorem 3.1 is for a generic Σ, while for Theorem 3.2 the result is specified under Assumption 3.3:

Assumption 3.3. *The covariance matrix of the multivariate Gaussian distribution of noise is multiple of the identity matrix, i.e., $\Sigma = \sigma^2 I$.*

Theorem 3.2. *Under Assumption 3.3, the solution to Equation* (3) *for $K = 1$ and $K = 2$ is given by:*

If $\|g_1 - g_2\| \leq L\ \langle g_1 - g_2,\ x_1 - x_2 \rangle$*:*

$$\begin{cases} \hat{\theta}_1 = g_1 \\ \hat{\theta}_2 = g_2\,. \end{cases}$$

If $\|g_1 - g_2\| > L\ \langle g_1 - g_2,\ x_1 - x_2 \rangle$*:*

$$\begin{cases} \hat{\theta}_1 = \frac{g_1 + g_2 + \frac{L}{2}(x_1 - x_2)}{2} + \|\frac{L}{4}(x_1 - x_2)\| \frac{g_1 - g_2 - \frac{L}{2}(x_1 - x_2)}{\|g_1 - g_2 - \frac{L}{2}(x_1 - x_2)\|} \\ \hat{\theta}_2 = \frac{g_1 + g_2 - \frac{L}{2}(x_1 - x_2)}{2} - \|\frac{L}{4}(x_1 - x_2)\| \frac{g_1 - g_2 - \frac{L}{2}(x_1 - x_2)}{\|g_1 - g_2 - \frac{L}{2}(x_1 - x_2)\|}\,. \end{cases}$$

The two cases in which we decompose the closed-form solution for COCO_2 have an intuitive explanation supporting them: when the two observed gradients are co-coercive ($\|g_1 - g_2\|^2 \leq L\langle g_1 - g_2,\ x_1 - x_2 \rangle$), they are on the feasible set of the problem, so, they are also the estimated gradients; when the two observed gradients are not co-coercive ($\|g_1 - g_2\|^2 > L\langle g_1 - g_2,\ x_1 - x_2 \rangle$), their difference is orthogonally projected onto its feasible set (which is a ball). This projection is achieved through the expression obtained for each of the estimated gradients in Theorem 3.2. This closed-form solution is of the utmost relevance, since the experiments described in the following section show that COCO leads to significant improvements in stochastic optimization, even for the simple case that considers only two gradients.

Since the closed-form solution for COCO_K for $K \geq 3$ could not be found (even with the help of symbolic manipulation packages of *Matlab* and *Mathematica*), a straightforward procedure to solve the QCQP in Equation (3) is its implementation onto *CVX* [5]. This tool is designed in such a way that it can be used as black-box, in the sense that the user does not need to understand how the problem can be solved, as far as that problem is presented in the format required by the software. As a consequence of its generality, this tool resorts to higher order methods (*e.g.*, second-order cone programming methods) which ensure high precision but necessarily end up being slower than methods which are specifically tailored for a given problem. This fact combined with the quadratic growth of the number of constraints with the K motivates an alternative approach. Therefore, a first-order algorithm which explores the particular structure of the QCQP is presented. In particular,

under Assumption 3.3 and considering the duality principle for optimization problems, this QCQP can be rewritten in the following form:

$$\underset{s}{\text{minimize}} \quad \underbrace{\frac{1}{2}\| - A^T s\|^2}_{p^*(-A^T s)} + \underbrace{\sum_{1\le m<l\le K} r_{ml}\|s_{ml}\| - s_{ml}^T c_{ml}}_{q^*(s)}, \tag{5}$$

where s is the dual variable of the original QCQP and consists of the stacked vector from the different s_{ml}, A is a structured matrix, and r_{ml} and c_{ml} can be obtained through simple computations from the constraints of the QCQP.

The first term from Equation (5), $p^*(-A^T s)$, is differentiable and a proximity operator can be efficiently computed for the second one, $q^*(s)$. Hence, we are in conditions of applying the Fast Dual Proximal Gradient (FDPG) method [6]. This approach consists of applying the Fast Iterative Shrinkage-Thresholding Algorithms (FISTA) to the a dual problem. Therefore, since FDPG is a first-order method (thus, with a very low cost per iteration), we obtain an efficient solution method for COCO. The FDPG instantiation for the COCO denoiser is represented below:

Algorithm .1: FDPG (applied to COCO Denoiser)

Input: Initial Point: s_0; Number of steps: T; L-Smoothness Constant: L; Momentum Auxiliary Iterate: $y_0 = s_0$; Initial Momentum Constant: $t_0 = 1$

for $i = 1, \dots, T$ **do**

$s_i = \text{prox}_{\frac{1}{L}q^*}(\ y_{i-1} - \frac{1}{L}\nabla p^*(-A^T y_{i-1}) \ )$

$t_i = \frac{1+\sqrt{1+4t_{i-1}^2}}{2}$

$y_i = s_i + \frac{t_{i-1}-1}{t_i}(s_i - s_{i-1})$

Output: Final Point: s_T

Through FDPG, it is possible to find an approximate solution of the dual problem, s^*, from which we easily recover the primal solution, $\hat{\theta}$, to the QCQP.

3.3 Properties of the Estimator

Regarding properties of the COCO estimator, we can attain a easily interpretable relation that relates the sum of the noisy gradients (COCO input) to the sum of the denoised ones (COCO output). It is expressed by the following theorem.

Theorem 3.3. *The gradients estimated by the $COCO_K$ denoiser, $\hat{\theta}_1, \dots, \hat{\theta}_K$, verify the following relation with its raw inputs, $g_1, \dots, g_K$:*

$$\sum_{i=1}^{K} \hat{\theta}_i = \sum_{i=1}^{K} g_i \, .$$

Note that this property holds for generic Σ and not only for $\Sigma = \sigma^2 I$, contrarily to the proof of Theorem 3.2. Moreover, by multiplying both sides of the equality from Theorem 3.3 by $1/K$, we obtain:

$$\frac{1}{K}\sum_{i=1}^{K} \hat{\theta}_i = \frac{1}{K}\sum_{i=1}^{K} g_i \, ,$$

showing that the centroid of the estimated and consulted gradients is the same, reinforcing the interpretability of this relation.

We are also able to ensure that the COCO estimator outperforms the oracle in terms of gradient estimation, through Theorem 3.4. Note that the Mean Squared Error (MSE) is a well-known performance metric for estimators (the smaller, the better).

Theorem 3.4. *From Equation* (3) *and under Assumption 3.3, the following inequality holds:*

$$MSE(\hat{\theta}) \le MSE(g), \tag{6}$$

where $\hat{\theta}$ denotes the stacked vector of the different $\hat{\theta}_k$ outputted from the $COCO_K$ denoiser and g denotes the stacked vector of the different g_k outputted from the oracle.

Every constraint included in the $COCO_K$ denoiser problem involves a pair of estimated gradients. Every time that two gradient observations, g_i and g_j are not co-coercive between them (note that to conclude that, it is required to also know x_i and x_j), the solution will have to output two estimated gradients, $\hat{\theta}_i$ and $\hat{\theta}_j$ which respect that condition and, thus, necessarily different from g_i and g_j. Noting that g_i and g_j are random variables, an important question to answer is: how often are the observed gradients incoherent with the co-coercivity constraint?

In order to find a reasonable answer to this problem, the following setup is proposed: for the sake of simplicity, our focus remains on the one-dimensional situation ($d = 1$) where we have access to two different points, x_1 and x_2. Without loss of generality, let us assume $x_1 > x_2$. The true gradients on those points are $\nabla f(x_1)$ and $\nabla f(x_2)$, whose noisy versions (provided by the oracle) are g_1 and g_2. Therefore, $g_1 \perp\!\!\!\perp g_2$[1] and $\Sigma = \sigma^2$, which is as general as possible for the one-dimensional case. We obtain the following result for the probability of g_1 and g_2 being co-coercive, $p_{inactive}$:

$$p_{inactive} = \Phi\left(\frac{L\Delta_x - \Delta_{\nabla f}}{\sqrt{2}\sigma}\right) - \Phi\left(\frac{-\Delta_{\nabla f}}{\sqrt{2}\sigma}\right), \tag{7}$$

where $\Delta_x = x_1 - x_2$ and $\Delta_{\nabla f} = \nabla f(x_1) - \nabla f(x_2)$.

From this expression, it is possible to analyse how the co-coercivity constraint becomes "looser" with the increase of the distance between x_1 and x_2, with the overestimation of the Lipschitz constant L and with the decrease in the variance of the oracle, σ^2.

4 EXPERIMENTS

4.1 Properties of the Estimator

Theorem 3.4 provides an upper bound ensuring that the gradient estimation using COCO is preferable than the raw oracle. Given this, an interesting problem is to study to what extent the former outperforms the latter. Moreover, from Section 3 it was possible to obtain for the one-dimensional case, an expression which relates the constraint tightness probability as a function of the distance between the two points from which we sample the noisy gradient estimates. Therefore, noting that, in the case in which the constraint

[1] The notation $\perp\!\!\!\perp$ denotes independence between random variables.

is not active, the COCO denoiser outputs the result provided by the oracle, and, in the case in which the constraint is active, the denoiser filters the output of the oracle, $\text{MSE}(\hat{\theta})$ is expected to be a function of the constraint tightness and, as a consequence, implicitly a function of the distance between points.

Taking this reasoning into consideration, the experiment represented in Figure 2 recovers experimentally the theoretical results obtained in Section 3. Therefore, an one-dimensional quadratic function, $f(x) = 1/2\ x^2$ was considered, thus with $L_{\text{real}} = 1$, where two points were considered: one fixed at $x_1 = 0$ and a variable point at $x_2 = \Delta_x$. The oracle consultations provided gradient estimates with additive Gaussian noise with $\Sigma = \sigma^2 = 100$. In this conditions, the probability of the constraint between the considered gradients being active, p_{active} $(= 1 - p_{inactive})$, and the $\text{MSE}(\hat{\theta})$ for each Δ_x were estimated through Monte Carlo simulations. It is possible to obtain a closed-form result for the $\text{MSE}(g)$ for a general number of points considered, K, a general dimension d and $\Sigma = \sigma^2 I$: $\text{MSE}(g) = Kd\sigma^2$. Hence, in this case, we have $\text{MSE}(g) = 200$.

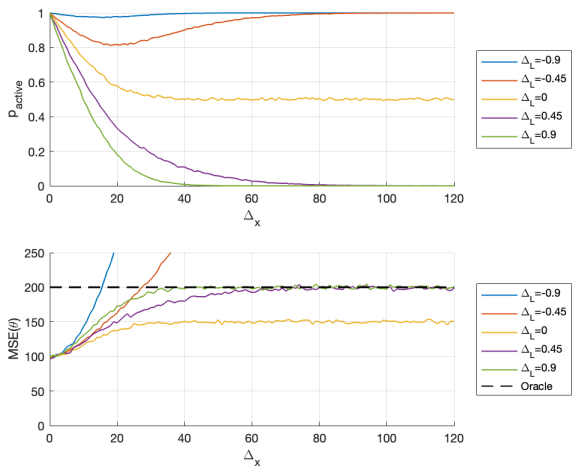

Figure 2: Top: Experimental plot for p_{active} as a function of Δ_x for different values of Δ_L. Bottom: Computed $\text{MSE}(\hat{\theta})$; the dashed line denotes the (theoretical) value for the oracle. Number of Monte-Carlo simulations (for both plots): $N = 10000$.

Regarding the $\text{MSE}(\hat{\theta})$, it can be observed that (i) for the cases in which the L is underestimated ($\Delta_L < 0$), the $\text{MSE}(\hat{\theta})$ is not guaranteed to be lower than $\text{MSE}(g)$. Nevertheless, note that there still is a range of Δ_x where $\text{MSE}(\hat{\theta}) \leq \text{MSE}(g)$. The more underestimated L is, the smaller this region becomes. This observation not only recalls that the result from Theorem 3.4 only holds for $\Delta_L \geq 0$, but also reinforces the importance of ensuring that the L considered for COCO denoiser is an upper bound for L_{real}; (ii) For the case in which the L is perfectly estimated ($\Delta_L = 0$), just as the p_{active} tends to an intermediate value, so it happens with $\text{MSE}(\hat{\theta})$. This is the ideal situation, as $\text{MSE}(\hat{\theta})$ is minimal for every Δ_L. Moreover, note that when the p_{active} curve stabilizes, the $\text{MSE}(\hat{\theta})$ also stabilizes, reinforcing the expected relation between those curves. (iii) For the cases in which the L is overestimated ($\Delta_L > 0$), just as p_{active} tends to 0, the $\text{MSE}(\hat{\theta})$ also tend to the $\text{MSE}(g)$ reference curve. Moreover, it is possible to see that when p_{active} stabilizes around 0, so it happens to $\text{MSE}(\hat{\theta})$ around the oracle's curve. This is easily explained, again, by the fact that when the constraints are loose, the COCO denoiser outputs the oracle results without any "filtering".

This analysis was performed for a metric which combines the information from every point considered. In particular, in order to have a better insight about what happens at each point, it is preferable to look at the MSE at each point, *i.e.*, for x_k, $\text{MSE}(\hat{\theta}_k) = E[\|\hat{\theta}_k - \nabla f(x_k)\|^2]$ and $\text{MSE}(g_k) = E[\|g_k - \nabla f(x_k)\|^2] = \sigma^2 d = 100$.

These results are represented in Figure 3 and they suggest that the $\text{MSE}(\hat{\theta})$ divides itself equally by the two points. It also presents empirical evidence on the following result: if $\Delta_L > 0$, then $E[\|\hat{\theta}_k - \nabla f(x_k)\|^2] \leq E[\|g_k - \nabla f(x_k)\|^2]$. Note that this inequality is stronger than the one from Theorem 3.4, as the former imposes each term on the left-hand side from the latter to be smaller or equal than the respective term on its right-hand side.

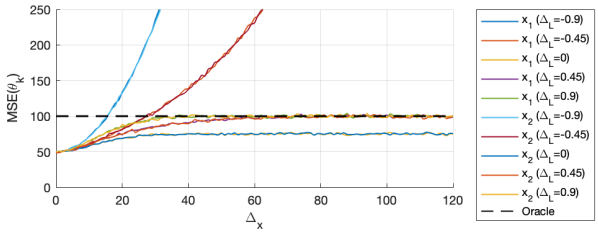

Figure 3: Experimental plot for $\text{MSE}(\hat{\theta}_k)$ as a function of Δ_x, for different Δ_L; the dashed line denotes the (theoretical) value for the oracle. Number of Monte-Carlo simulations: $N = 10000$.

In order to further reinforce this result, we investigate the possibility of generalizing the statement $E[\|\hat{\theta}_k - \nabla f(x_k)\|^2] \leq E[\|g_k - \nabla f(x_k)\|^2]$ for arbitrary dimension, d, and number of points considered, K. We find empirical evidence that, as conjectured with only two points, closer iterates allow lower MSE, even for higher number of points and dimension. Moreover, when the distance is sufficiently high, the COCO denoiser approximates its behaviour from the oracle; in particular, we observe that $E[\|\hat{\theta}_k - \nabla f(x_k)\|^2] = \text{MSE}(\hat{\theta}_k) \leq E[\|g_k - \nabla f(x_k)\|^2] = \text{MSE}(g_k)$ for every point in every tested setting, reinforcing the empirical evidence on that sense; this result allows us to conclude that $\text{Var}(\hat{\theta}_k) \leq \text{Var}(g_k)$, making the intended variance reduction via COCO explicit. Moreover, contrarily to what was supposed for the case in which $K = 2$, we verify that the $\text{MSE}(\hat{\theta})$ does not distribute evenly among the different points (*i.e.*, $\text{MSE}(\hat{\theta}_k)$ varies from point to point when $K > 2$). In fact, that is a consequence of the symmetry from the COCO_2 closed-form solution, property which does not hold for higher K solutions. In particular, points which have other points closer, present lower $\text{MSE}(\hat{\theta}_k)$.

We also study the $\text{MSE}(\hat{\theta}_k)$ for different numbers of points considered, K. In particular, in this case, $1 \leq K \leq 10$ and the different iterates are generated randomly following an uniform distribution inside a cube centered at the origin with edge length of 10

($x_k \in [-5,\ 5] \times [-5,\ 5] \times [-5,\ 5]$). Moreover, we consider a three-dimensional space ($d = 3$), an anisotropic Hessian with eigenvalues linearly spaced between 1 and 1/3 and $N = 1000$. Note that this choice for the size of the cube is not inadequate having in mind that the first-order algorithms in which this scheme will be applied, as, for example, for GD, its optimal step size is $\gamma = 1/L = 1$. The results obtained are shown in Figure 4. We recover the known result for the oracle: $\text{MSE}(g_k) = d\sigma^2$, which in this plot is represented by a line of zero intercept and slope d. Surprisingly, the correspondent results for COCO_K suggested that those results could as well be approximated by a line with zero intercept (when there is no noise, the noisy gradients correspond to the real ones, then no error is expected for both estimators).

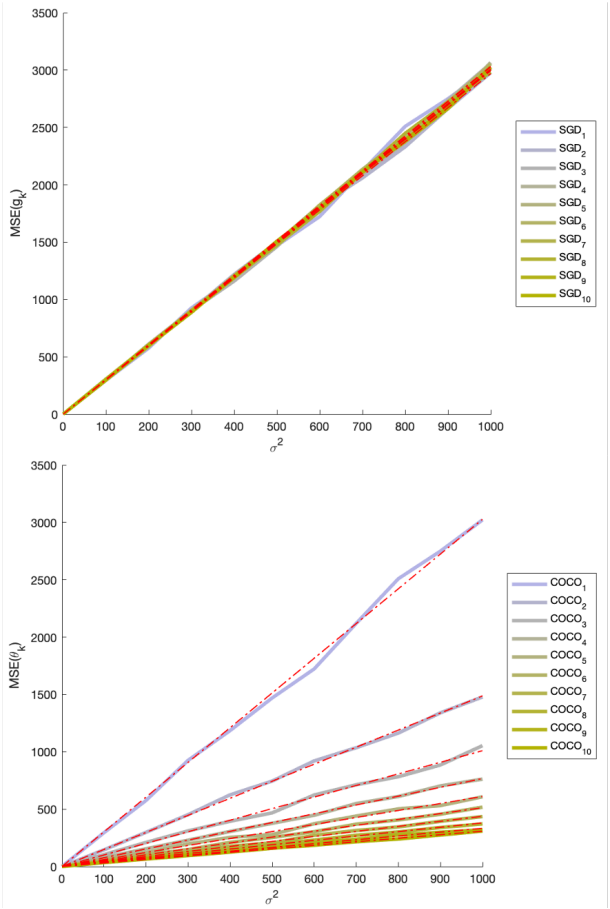

Figure 4: $\text{MSE}(g_k)$ (top) and $\text{MSE}(\hat{\theta}_k)$ (bottom), estimated via Monte-Carlo method, as functions of the noise variance σ^2, for several numbers K of points considered. The dashed-dotted red lines result from linear regressions with intercept fixed at 0. Number of Monte-Carlo simulations: $N = 1000$. For each simulation, a different set of points is randomly generated from an uniform distribution in a cube centered at the origin with edge length 10.

These results suggest the slope of $\text{MSE}(\hat{\theta}_k)$ to be $O(1/K)$, while, for $\text{MSE}(g_k)$, it remains constant (independently of K). This result for $\text{MSE}(\hat{\theta}_k)$ is remarkable, as this is the usual result for the common averaging of normally distributed variables. In this case, that averaging would require that at each iterate x_k, K gradient estimates of the oracle were required. With COCO_K, it is possible to achieve the same $\text{MSE}(\hat{\theta}_k)$ without having to be stuck on the same position. Therefore, $COCO_K$ can be interpreted as an extension to that procedure in the sense that it allows to integrate more information for more precise gradient estimates without having to stop the iterate progression.

Regarding the bias of the these gradient estimators, the oracle whose noise follows the additive and normally distributed model is unbiased. We are interested in also having some characterization of $\hat{\theta}_k$ at this respect. In order to test the bias of the COCO denoiser estimator, we estimate $\|Bias(\hat{\theta}_k)\|$ (note that if it is an unbiased estimator, then $\|Bias(\hat{\theta}_k)\| = 0$) via Monte-Carlo simulations. Therefore, considering this estimator in the same setup used to obtain Figure 2 and Figure 3, the results in Figure 5 were achieved.

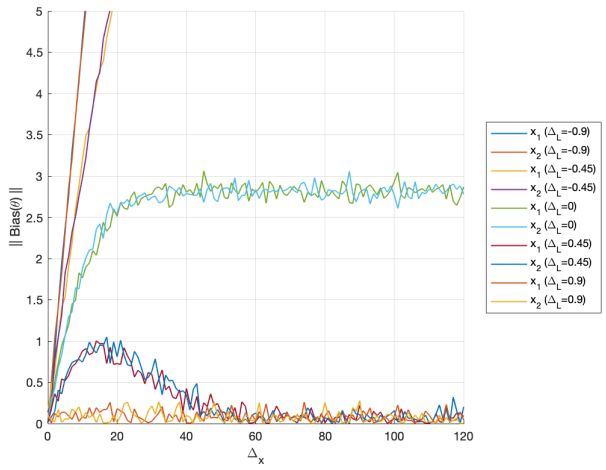

Figure 5: Bias as a function of the distance between the points considered, for the setup of Figure 2.

The conclusions to take from Figure 5 are consistent to the ones in the previous section: for $\Delta_L < 0$, the bias of this estimator seems to grow linearly with Δ_x; the smaller the Δ_L, the higher the slope of that linear relation. For $\Delta_L = 0$, the estimator is biased as well; that bias grows until a stabilization which happens at the Δ_x that it happened with p_{active}. For $\Delta_L > 0$, the estimator is also biased. As the COCO estimator outputs become more similar to the ones of the oracle, its bias decreases. Moreover, the higher the Δ_L, the lower the bias (as the constraints are less restrictive). In all cases, for $\Delta_x = 0$, the COCO estimator is unbiased since it consists of the averaging estimator (see Theorem 3.2). Generically, these observations suggest that if the constraint between two gradient estimates is active (except for $\Delta_x = 0$), then it imposes bias on the COCO estimator. On the other hand, when the constraint is inactive, the COCO estimator outputs the oracle consultations, which are known to be unbiased.

4.2 Stochastic Optimization with COCO

By coupling a baseline algorithm with COCO_K, note that, at iteration i, only the oldest gradient (g_{i-K}) is forgotten and a new one (g_i) is kept in memory. Thus, it is reasonable to think of taking advantage from the COCO_K solution obtained for the previous iterate to obtain a new solution faster. We suggest a warm-starting procedure for the COCO_K solution method (FDPG) which enables this utilization of past information. In particular, we achieve it by a careful initialization of the dual variable, s. In fact, s is the vector that results from stacking the different s_{ml}, where each s_{ml} addresses the co-coercivity constraint between the COCO estimates for gradient m, $\hat{\theta}_m$, and for gradient l, $\hat{\theta}_l$. Since we expect the estimates for old gradients to only have small relative variations among them on the new iterate as they have been "filtered" at least once, we initialize these s_{ml} to the values obtained for the correspondent dual variables in the previous COCO_K solution. For the multiple s_{ml} concerning the new gradient, we do not have any information yet, thereby being initialized to the default value (zero). This warm-starting procedure allows the iterative method to start with a much better guess of s^*, thereby achieving satisfactory approximate solutions faster.

We assess the usefulness of COCO_K in a scenario whose setup perfectly matches the assumptions under which the denoiser was proposed (*synthetic dataset*). In this case, the objective function is a 10-dimensional ($d = 10$) quadratic function, $f(x) = 1/2\ x^T Ax$, where the matrix A is the Hessian of the objective function. We consider an anisotropic Hessian, with eigenvalues linearly separated between 1 and 1/3, with the minimum at $x^* = (0, 0, \ldots, 0)^T$. Moreover, the first-order oracle provides a gradient estimate whose noise is additive and normally distributed, with $\Sigma = 100\ I$. The initial iterate was kept the same through all the simulations, $x_0 = (100, 100, \ldots, 100)^T$.

Considering the setup described above, the COCO_K denoiser is coupled both to SGD and Adam, where the latter is picked as a representative of the class in which it is inserted (adaptive step size algorithms). Note that algorithms that address finite sum objectives are not here tested, as the setup considered falls out of their scope. In both cases, the COCO_K hyperparameters are correctly set: $L = 1$, $\Sigma = 100\ I$. Given that we are in a stochastic setting, the quantity that we are interested in following across iterations is $E[\|x_i - x^*\|]$, which is again estimated via Monte-Carlo method. We depict these results in Figure 6.

From this figure, it is possible to recall that there is an initial *bias regime*, where all the algorithms seem to converge linearly (see Table 1; we are in an L-smooth and strongly convex setting, where GD is known to converge linearly and the stochastic algorithms are able to keep up with it initially); across iterations that convergence is successively slowed down and eventually leads to a stagnation to which we call *variance regime*. In fact, from Figure 6, we can observe that a higher K in COCO_K leads to improved performance at least in terms of the variance regime (without compromising the bias one). Moreover, it can be shown that the "level" at which SGD stops converging is directly dependent on the (uncentered) variance of the oracle. This reinforces the variance reduction achieved by coupling COCO_K to a baseline algorithm.

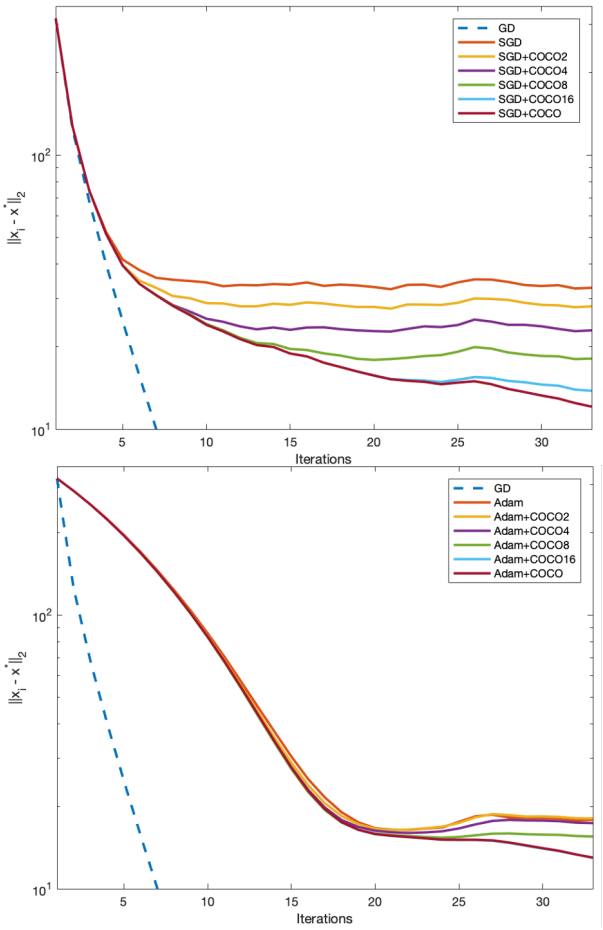

Figure 6: Results of stochastic optimization with the proposed COCO denoiser, when used with baselines SGD (top) and Adam (down). The performance is measured in terms of $E[\|x_i - x^*\|]$ (the performance of GD is also depicted for reference; the lines for "Adam + COCO_{16}" and "Adam + COCO" are superimposed). Number of Monte-Carlo simulations: $N = 100$.

Further analysis of the COCO plug-in in this setup is carried out for an observation of its gains in variance, as well as the expected performance deterioration caused by the wrong choice of the L considered by COCO. Another interesting property from typical averaging algorithms is also explored in this context: the COCO plug-in is able to stabilize the baseline algorithm for larger step sizes. This result reinforces the interpretation of COCO estimator as an extension of the averaging estimator, but which allows the integration of (gradient) information coming from different points.

We also test this coupling in a real context by applying it to readily available datasets. In particular, we analyse the performance of coupling COCO_K to SGD, Adam, and SAG in Tikhonov regularized logistic regression problems based on the "fourclass" (smaller dataset, with $n = 862$ and $d = 2$) and "mushrooms" (bigger dataset, with $n = 8124$ and $d = 112$) datasets [23].

From the first two plots in Figure 7, it is possible to observe the improvements brought by the COCO plug-in in the "fourclass"

dataset, namely on the variance regime of these algorithms. Contrarily to the SGD case, there is no delay in the bias regime for the $COCO_K$ coupled Adam cases, since it adapts the step size accordingly to the magnitude of the gradient provided. On the other hand, from the last plot in Figure 7, it is possible to conclude that the naive $COCO_K$ coupling to SAG does not bring any benefit. It halts the linear convergence which characterizes this method. Nevertheless, taking into consideration that SAG is specifically designed for finite sum objectives, this failure is understandable. In fact, this setup falls out of the assumptions of $COCO_K$ since the different gradients sampled are not completely independent samples (at least the ones coming from the same example in the dataset) and the oracle noise does not follow an additive and normally distributed model (even though, by resorting to mini-batches to compute the gradient estimate, its distribution approximates itself from a Gaussian by the Central Limit Theorem).

These results are transposable to the “mushrooms" dataset.

5 CONCLUSION

This chapter summarizes our work and suggests possible extensions and future research directions.

5.1 Summary

This thesis introduced the COCO denoiser, which exploits co-coercivity of convex and L-smooth objective functions to denoise gradient estimates provided by a stochastic oracle. Our denoiser is based on the joint ML estimation of the gradients, constrained by the co-coercivity conditions. Our theoretical analysis enables finding an interpretable relation between the observations and COCO estimates.

By assuming a noise model with covariance proportional to the identity, we proved that the estimates provided by COCO are necessarily more accurate than the oracle, in what respects to MSE. For this case, we introduced an efficient first-order solution to COCO, based on the FDPG method. By considering the simpler scenario of optimizing a function of a single variable, we conclude that the MSE deteriorates with the distance between the points where the gradients are observed and with the model mismatch in what regards to the Lipschitz constant L.

Our computational experiments corroborate the theoretical results above and have also shown that the elementwise MSE decreases with the rate of $O(1/K)$, where K is the number of gradients simultaneously estimated from sufficiently close points. This is the same rate obtained for a gradient averaging estimator that had access to K observations of the gradient at the same point, which supports interpreting the COCO denoiser as an extension that allows incorporating information from different points.

In stochastic optimization, our experiments with synthetic data have shown that current first-order methods coupled with $COCO_K$ lead to variance reduction, an increase in performance that is noticed even for the case in which only two points are considered. This is particularly relevant because we derived the closed-form solution for $COCO_2$.

To illustrate the usefulness of COCO in a real online learning task, we solve a logistic regression problem. Although algorithms such as SAG, which exploits the finite sum decomposition of the objective function, do not gain by using COCO estimates, our experiments show that more general baseline algorithms, such as SGD or Adam, clearly exhibit variance reduction.

5.2 Future Work

A simple task that deserves attention in the immediate future is the experimental exploration of the limits of COCO in what respects to dealing with situations that do not fully match the design assumptions. For example, even for problems requiring the (local) minimization of a non-convex function (*e.g.*, deep learning), there is hope for improvement of baseline first-orders methods when coupled with COCO, since the objective function is often locally convex.

First-order algorithms for stochastic optimization can be considered to also estimate (in a non-explicit way) the function gradient (SGD estimates it as the noisy observation itself, while others, *e.g.*, Adam or SAG, have their own operations on the noisy gradient). This observation motivates the possibility of using COCO for stochastic optimization in a slightly different way that the one explored in the thesis: instead of feeding baseline algorithms with the output of COCO, why not feed COCO with the gradient estimates provided by the baseline algorithms? The standard gradient descent steps would then use directly the output of COCO.

Naturally, our theoretical analysis of COCO can be extended. It would be interesting to demonstrate the universality of the evidence provided by our experiments, namely in what respects to the estimator bias and variance (at least for $COCO_2$, for which there is closed-form solution) and the decrease with K of the elementwise MSE of $COCO_K$. Regarding the usage of COCO as a plug-in for stochastic optimization, it would be important to study convergence guarantees (which, naturally, also depend on the baseline algorithm) and to quantify the gains in variance reduction.

Aspects of computational efficiency can also motivate future work. For example, the extension of the proposed FDPG efficient method for COCO to deal with more general noise covariance matrices. This would certainly bring robustness to the denoiser, which, despite the predictable higher computational cost, could widen the range of application scenarios. Another interesting line of thought concerns dealing with the quadratic scaling of the number of constraints with the number of points simultaneously considered. In fact, our analysis showed that the larger gains in denoising come from close-by points, which could motivate strategies to reduce (maybe to a linear dependence) the number of constraints that could effectively be considered without compromising the results.

More exploratory lines of research would address the possibility of denoising gradients using different assumptions on the underlying objective function. For example, strong convexity, which has lead to better convergence rates for stochastic optimization algorithms, or the finite sum decomposition that is omnipresent in machine learning applications. Even in the non-convex setting, it could be interesting to consider the single assumption of L-smoothness, since, just as in the convex case, it would prevent arbitrarily fast changes of the gradient, thereby promising denoising capabilities.

Finally, our insights relative to the influence of the location of the query points may motivate strategies for active learning, *i.e.*,

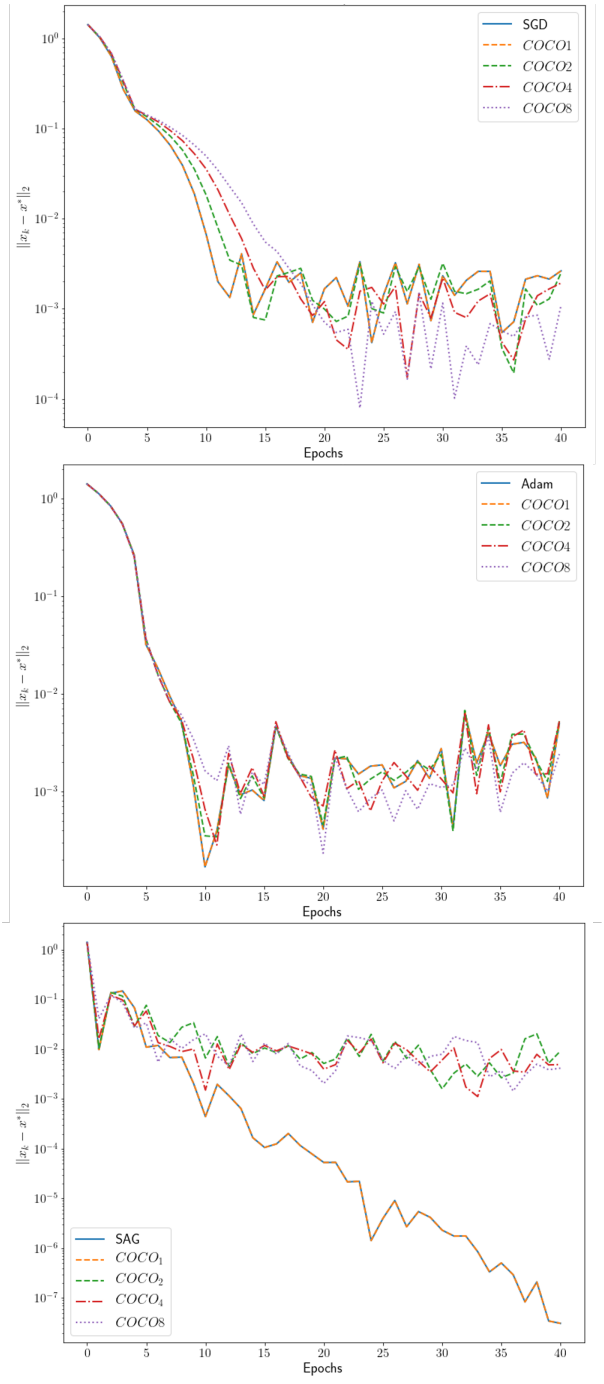

Figure 7: Top: Performance results obtained for SGD and its coupling with COCO_K in the "fourclass" dataset [23]. Note the performance improvement with the increase of K. Middle: Same as the plot above, now for baseline algorithm Adam. Bottom: Same as the plots above, now for baseline algorithm SAG. In this case, no improvements are observed with the increase in K and the linear convergence of SAG is even compromised by COCO.

for actively selecting those points, rather than passively using only the past iterates.

REFERENCES

[1] S. P. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge, UK ; New York: Cambridge University Press, 2004.

[2] Y. E. Nesterov, *Introductory lectures on convex optimization: a basic course*, ser. Applied optimization. Boston: Kluwer Academic Publishers, 2004, no. v. 87.

[3] R. M. Gower, M. Schmidt, F. Bach, and P. Richtarik, "Variance-Reduced Methods for Machine Learning," *arXiv:2010.00892 [cs, math, stat]*, Oct. 2020, arXiv: 2010.00892. [Online]. Available: http://arxiv.org/abs/2010.00892

[4] X. Zhou, "On the Fenchel Duality between Strong Convexity and Lipschitz Continuous Gradient," *arXiv:1803.06573 [math]*, Mar. 2018, arXiv: 1803.06573. [Online]. Available: http://arxiv.org/abs/1803.06573

[5] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming, version 2.1," Mar. 2014. [Online]. Available: http://cvxr.com/cvx/

[6] A. Beck and M. Teboulle, "A fast dual proximal gradient algorithm for convex minimization and applications," *Operations Research Letters*, vol. 42, pp. 1–6, Jan. 2014.

[7] M. Madeira, R. Negrinho, J. Xavier, and P. Aguiar, "COCO - Exploring co-coercivity to filter noisy gradients," *To be submitted*, 2021.

[8] ——, "Variance Reduction in Stochastic Convex Optimization using Using Co-Coercivity," *To be submitted*, 2021.

[9] H. Robbins and S. Monro, "A Stochastic Approximation Method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, Sep. 1951. [Online]. Available: http://projecteuclid.org/euclid.aoms/1177729586

[10] S. Bubeck, "Convex Optimization: Algorithms and Complexity," *arXiv:1405.4980 [cs, math, stat]*, Nov. 2015, arXiv: 1405.4980. [Online]. Available: http://arxiv.org/abs/1405.4980

[11] A. S. Nemirovsky and D. B. Yudin, *Problem complexity and method efficiency in optimization*, ser. Wiley-Interscience series in discrete mathematics. Chichester ; New York: Wiley, 1983.

[12] A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright, "Information-Theoretic Lower Bounds on the Oracle Complexity of Stochastic Convex Optimization," *IEEE Transactions on Information Theory*, vol. 58, no. 5, pp. 3235–3249, May 2012. [Online]. Available: http://ieeexplore.ieee.org/document/6142067/

[13] B. T. Polyak and A. B. Juditsky, "Acceleration of Stochastic Approximation by Averaging," *SIAM Journal on Control and Optimization*, vol. 30, no. 4, pp. 838–855, Jul. 1992. [Online]. Available: http://epubs.siam.org/doi/10.1137/0330046

[14] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv:1412.6980 [cs]*, Jan. 2017, arXiv: 1412.6980. [Online]. Available: http://arxiv.org/abs/1412.6980

[15] M. Schmidt, N. L. Roux, and F. Bach, "Minimizing Finite Sums with the Stochastic Average Gradient," *arXiv:1309.2388 [cs, math, stat]*, May 2016, arXiv: 1309.2388 version: 2. [Online]. Available: http://arxiv.org/abs/1309.2388

[16] B. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, pp. 1–17, Jan. 1964. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0041555364901375

[17] Y. E. Nesterov, "A method for solving the convex programming problem with convergence rate O(1/k^2)," *Dokl. Akad. Nauk SSSR*, vol. 269, pp. 543–547, 1983. [Online]. Available: https://ci.nii.ac.jp/naid/10029946121/

[18] G. Lan and Y. Zhou, "An optimal randomized incremental gradient method," *arXiv:1507.02000 [cs, math, stat]*, Oct. 2015, arXiv: 1507.02000. [Online]. Available: http://arxiv.org/abs/1507.02000

[19] B. Woodworth and N. Srebro, "Tight Complexity Bounds for Optimizing Composite Objectives," *arXiv:1605.08003 [cs, math, stat]*, Apr. 2019, arXiv: 1605.08003. [Online]. Available: http://arxiv.org/abs/1605.08003

[20] S. Shalev-Shwartz and T. Zhang, "Accelerated Proximal Stochastic Dual Coordinate Ascent for Regularized Loss Minimization," in *International Conference on Machine Learning*. PMLR, Jan. 2014, pp. 64–72, iSSN: 1938-7228. [Online]. Available: http://proceedings.mlr.press/v32/shalev-shwartz14.html

[21] F. Bach and E. Moulines, "Non-strongly-convex smooth stochastic approximation with convergence rate O(1/n)," *arXiv:1306.2119 [cs, math, stat]*, Jun. 2013, arXiv: 1306.2119. [Online]. Available: http://arxiv.org/abs/1306.2119

[22] K. Basu, A. Saha, and S. Chatterjee, "Large-Scale Quadratically Constrained Quadratic Program via Low-Discrepancy Sequences," Oct. 2017. [Online]. Available: https://arxiv.org/abs/1710.01163v1

[23] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 27:1–27:27, May 2011. [Online]. Available: https://doi.org/10.1145/1961189.1961199