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

Manuel Madeira[1] MANUEL.MADEIRA@TECNICO.ULISBOA.PT
Renato Negrinho[2] NEGRINHO@CS.CMU.EDU
João Xavier[1] JXAVIER@ISR.TECNICO.ULISBOA.PT
Pedro M. Q. Aguiar[1] AGUIAR@ISR.TECNICO.ULISBOA.PT
[1] *IST, Portugal* [2] *CMU, USA*

Abstract

First-order methods for stochastic optimization have undeniable relevance. Variance reduction for these algorithms has become an important research topic. We exploit convexity and L-smoothness to improve the noisy estimates outputted by the stochastic gradient oracle. Our method, named COCO denoiser, is the joint maximum likelihood estimator of multiple function gradients from their noisy observations, subject to co-coercivity constraints between them. The resulting estimate is the solution of a convex Quadratically Constrained Quadratic Problem. Although this problem is expensive to solve by interior point methods, we exploit its structure to apply an accelerated first-order algorithm, the Fast Dual Proximal Gradient method. Besides analytically characterizing the proposed estimator, we show empirically that increasing the number and proximity of the queried points leads to better gradient estimates. We also apply COCO in stochastic settings by plugging it in existing algorithms, such as SGD, Adam or STRSAGA, outperforming their vanilla versions, even in scenarios where our modelling assumptions are mismatched. [1]

1. Introduction

We study first-order solution methods to problems of the form:

$$\min_{x\in\mathbb{R}^d} \left\{ f(x) = \mathbb{E}\left[f_\xi(x)\right] \right\}, \tag{1}$$

where f is a convex and L-smooth function and the randomness denoted by the index ξ results from the selection of data points. This type of formulation commonly arises in cases where the exact gradient of the objective function f, $\nabla f(x)$, cannot be easily obtained, being preferable to consult a first-order stochastic oracle [3] that provides us with a noisy but unbiased gradient estimate, $\nabla f_\xi(x) = g_\xi(x)$. The difficulties in obtaining an exact gradient are typically caused by either the computational cost (*e.g.*, in large-scale machine learning problems, where we have a huge but finite number of indexes), or by the intrinsic nature of problem (*e.g.*, in *streaming applications*, where we have an infinite number of indexes [11]). We focus our attention on the latter, in applications where the oracle queries are very expensive, therefore measuring the progress of the different first-order methods as a function of the number of gradient evaluations.

Related Work. In the core of stochastic optimization, we find SGD [14], which despite its simplicity, remains a fundamental algorithm. Based on queries to a stochastic first-order oracle queries,

[1] Code for the experiments and plots is available at *https://github.com/ManuelMLMadeira/COCO-Denoiser*.

the iterates generated by SGD to solve (1) are of the form: $x_{k+1} = x_k - \gamma_k \, g_k$, where γ_k denotes the step size. In the convergence analysis of this method, the bias term vanishes under a convenient selection of a fixed step size, but that does not happen to the variance term. To enable convergence, some variance reduction approaches have been proposed, such as *i)* $\boldsymbol{O(1/k)}$ **decreasing step sizes** or *ii)* **averaging algorithms**, where we include the so-called Polyak-Ruppert (PR) averaging [13] and more refined averaging schemes (e.g. [9]). Other improvements on SGD have also been achieved by addressing other weaknesses of the original methods. In particular, the **adaptive (step size) algorithms** overcome sensitivity to initialization by successively adjusting the step size in each dimension according to the magnitude of the progress in that same dimension (e.g. see [6–8, 12, 17], from where we pick as its representative Adam [12]), drastically improving the performance in high condition number problems. More recently, the **variance-reduced (VR) methods** emerged, which despite being originally derived for a finite number of indexes ε in (1) [5, 10, 15], already have streaming versions (e.g. STRSAGA [11]).

Our Approach. We leverage on the convexity and L-smoothness of f. These properties can be merged into *gradient co-coercivity*, which we exploit to denoise a set of gradients $g_1, \dots, g_k$, obtained from an oracle [3] consulted at iterates $x_1, \dots, x_k$, respectively. We refer to our method as the co-coercivity (COCO) denoiser and plug it in existing stochastic first-order algorithms (see Figure 1). The COCO denoiser is obtained from the joint maximum likelihood estimation of the function gradients constrained by the pairwise co-coercivity constraints (see Section 2.1). This estimator can be expressed as convex quadratically constrained quadratic problem (QCQP) [2], for which we derive the closed-form solution when dealing with two observations and introduce an accelerated algorithm for the general case, based on Beck and Teboulle [1] (see Section 2.2). We study the estimator properties, finding empirical evidence that the COCO denoiser yields better gradient estimates than the stochastic oracle and that variance monotonically decreases with the number of gradients used, providing a natural way to trade off variance reduction and computation (see Section 3). Our experiments also illustrate that current stochastic first-order methods (SGD [14], Adam [12] and STRSAGA [11]) benefit from using gradients denoised by COCO, even in settings where the assumptions from COCO are mismatched (see Section 4 and Appendix E).

2. COCO Denoiser

First, we formulate COCO as a maximum likelihood estimator constrained by the co-coercivity conditions; then, we propose efficient methods to compute its solution.

2.1. Maximum Likelihood Estimation

Let the objective function $f : \mathbb{R}^d \to \mathbb{R}$ be convex and L-smooth. A standard result in convex analysis states that the gradient of f is co-coercive [2], which is expressed as

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

Our approach hinges on the following assumptions:

Assumption 2.1 *A Lipschitz constant L for the gradient of f is known.*

Assumption 2.2 *There is access to an oracle which, given an input $x \in \mathbb{R}^d$, outputs a noisy version of the gradient of f at x: $g(x; w) = \nabla f(x) + w$, where $w \in \mathbb{R}^d$ is a sample of a zero mean Gaussian* *distribution, $w \sim \mathcal{N}(0, \Sigma)$, with covariance $\Sigma = \sigma^2 I$. The noise samples are independent across the oracle queries.*

The oracle is consulted at points $x_1, \ldots, x_K$, returning the data vector $g = [g_1, \ \ldots, \ g_K]^T \in \mathbb{R}^{Kd}$, from which our goal is to estimate the true gradients $\nabla f(x_1), \ \ldots, \ \nabla f(x_K)$, arranged in the parameter vector $\theta = [\theta_1, \ \ldots, \ \theta_K]^T \in \mathbb{R}^{Kd}$, where $\theta_k = \nabla f(x_k) \in \mathbb{R}^d$. From 2.2, the observation model is immediate: $g = \theta + w$, where $w = [w_1, \ \ldots, \ w_K]^T \sim \mathcal{N}(0, \Sigma_w)$, with Σ_w block-diagonal, each block being Σ. The maximum likelihood estimate [16] of θ is then

$$\hat{\theta} = \underset{\theta \in \Theta}{\text{argmax}} \ p(g|\theta), \qquad \text{with} \qquad p(g|\theta) = \frac{1}{\sqrt{(2\pi)^n |\Sigma|}} e^{-\frac{1}{2}(g-\theta)^T \Sigma_w^{-1} (g-\theta)},$$

where the parameter vector θ is constrained by the co-coercivity condition (2), *i.e.*,

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

Consequently, the maximum likelihood estimate $\hat{\theta}$ comes from solving the following optimization problem:

$$\begin{aligned} \min_{\theta_1, \ldots, \theta_K} \quad & \sum_{i=1}^{K} \|g_i - \theta_i\|^2 \\ \text{subject to} \quad & \frac{1}{L} \|\theta_m - \theta_l\|^2 \leq \langle \theta_m - \theta_l, x_m - x_l \rangle, \ 1 \leq m < l \leq K. \end{aligned} \tag{3}$$

Since both the objective and the constraints in (3) are convex quadratics, the resulting problem is a convex Quadratically Constrained Quadratic Problem (QCQP) [2]. Since there is one constraint for each pair of query points, the total number of constraints in (3) is $K(K-1)/2$. This quadratic growth motivates an approach where we keep only the K last query points ($1 < K \leq k$, where k is the total number of queried points). For example, for $K = 2$, the denoiser works only with $x_{k-1}, \ x_k, \ g_{k-1}$ and g_k. We define COCO$_K$ to be the denoiser that uses a window of length K.

2.2. Efficient Algorithms for COCO$_K$

To solve the COCO optimization problem (3), we present its closed-form solution for $K = 2$ in Appendix A and propose an iterative method to efficiently compute its approximate solution for arbitrary K. We present a first-order algorithm which explores the COCO structure[2]. The dual problem of the QCQP in (3) can be shown to be

$$\min_s \ \underbrace{\frac{1}{2} \| - A^T s\|^2}_{p^*(-A^T s)} + \underbrace{\sum_{1 \leq m < l \leq K} r_{ml} \|s_{ml}\| - s_{ml}^T c_{ml},}_{q^*(s)} \tag{4}$$

where $s = [s_{12}, s_{13}, \ldots, s_{1K}, s_{23}, s_{2K}, \ldots, s_{K-1K}]^T$ is the dual variable, A is a structured matrix, $c_{ml} = (g_m - (L/2)\, x_m) - (g_l - (L/2)\, x_l)$ and $r_{ml} = L\|x_m - x_l\|/2$. The first term in (4), $p^*(-A^T s) = 1/2 s^T A A^T s$, is differentiable, with $\nabla_s \, p^*(-A^T s) = A A^T s$. Note that $p^*(-A^T s)$ is

[2] A more detailed derivation of the method is provided in Appendix B.

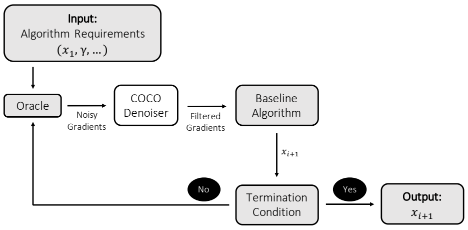

Figure 1: Typical workflow for stochastic optimization (grey), to which we plug-in COCO denoiser (white).

```
Algorithm 1: FDPG for the COCO denoiser
Input: Initial Point: $s_0$; Number of steps: $T$;
Aux. Iterate: $y_0 = s_0$; Aux. Momentum: $t_0 = 1$;
$L$-Smoothness Constant: $L_{p^*}$;
for $k = 1, \ldots, T$ do
    $s_k = \text{prox}_{\frac{1}{L_{p^*}}q^*}\left(y_{k-1} - \frac{1}{L_{p^*}}\nabla p^*(-A^T y_{k-1})\right)$
    $t_k = \frac{1+\sqrt{1+4t_{k-1}^2}}{2}$
    $y_k = s_k + \frac{t_{k-1}-1}{t_k}(s_k - s_{k-1})$
end
Output: Final Point: $s_T$;
```

necessarily L-smooth, with a Lipschitz constant of $L_{p^*} = \sigma^2_{\text{max}}(A)$. On the other hand, despite its non-differentiability, a proximity operator can be efficiently computed for the second term, $q^*(s)$: $\text{prox}_{\mu q^*}(s) = s - \mu(v_{\text{proj}} - c)$, where $v_{\text{proj}} = \text{argmin}_{v \in \mathcal{B}} \|v - (c + s/\mu)\|^2$ collects the projections of $c_{ml} + s_{ml}/\mu$ onto the ball $\mathcal{B}(0,\ r_{ml}) = \{x \in \mathbb{R}^n : \|x\| \leq r_{ml}\}$. Hence, we can use the Fast Dual Proximal Gradient (FDPG) method [1]. This approach consists of applying FISTA to the dual problem of the original one. Since FDPG is a first-order method with a low cost per iteration, we obtain a computationally efficient solution for COCO. After computing an approximate solution for the dual problem, s^*, we can easily recover the primal solution for the QCQP: $\hat{\theta} = -A^T s^* + g$. The FDPG method for the COCO denoiser is summarized in Algorithm 1.

3. COCO Estimator Properties

While in Appendix C we theoretically prove some simple results for the COCO estimator, here we focus our attention on their experimental exploration. In particular, we find empirical evidence that the COCO denoiser decreases the elementwise MSE[3], *i.e.*, that the $\text{MSE}(\hat{\theta}_k) \leq \text{MSE}(g_k)$, thus providing better gradient estimates than the oracle. This result also makes explicit the variance reduction provided by COCO, since $\text{Var}(\hat{\theta}_k) \leq \text{MSE}(\hat{\theta}_k) \leq \text{MSE}(g_k) = \text{Var}(g_k)$. One of the instances generated is represented in Figure 2. We observe that when points are inside the tighter cube, we obtain the best COCO denoising (lower $\text{MSE}(\hat{\theta}_k)$). On the other hand, for the looser cube, the COCO denoising capability is almost non-existent, tending to the oracle values. Regarding the intermediate cube, it is shown in Section D.2 that the more isolated points are the ones with worse $\text{MSE}(\hat{\theta}_k)$. As exposed in Appendix D, we find out that the better perfomance (lower MSE) for closer points is intrinsically related to the fact that smaller distances between points make the co-coercivity constraints in (3) tighter.

We also found evidence that for sufficiently close points, $\text{MSE}(\hat{\theta}_k) = C\sigma^2$, with C being $O(1/K)$, while for the oracle, C is obviously $O(1)$ (see Figure 3). This result for $\text{MSE}(\hat{\theta}_k)$ is the same as for the averaging of normal random variables, enabling a nice interpretation: while direct averaging would require that K gradient observations to be available at each iterate x_k, with COCO_K, we achieve the same $\text{MSE}(\hat{\theta}_k)$ without having to be stuck on that point for K iterates.

[3] Elementwise MSE for COCO: $\text{MSE}(\hat{\theta}_k) = E[\|\hat{\theta}_k - \nabla f(x_k)\|^2]$; for the oracle: $\text{MSE}(g_k) = E[\|g_k - \nabla f(x_k)\|^2]$

COCO can then be interpreted as an extension to the averaging procedure, allowing to integrate information from different points.

4. Stochastic Optimization

In this section, we show that SGD benefits from using COCO gradient estimates when plugged in as represented in Figure 1, both for a scenario that completely matches Assumptions 2.1 and 2.2 (**Synthetic Data**) and one where those assumptions are mismatched (**Real Data**). These experiments are extended to Adam and STRSAGA in Appendix E. We also propose a warm-starting procedure for the COCO denoiser iterative solution method (FDPG) for first-order stochastic methods (detailed in Appendix B.4).

Synthetic Data. The first-order oracle provides observations whose noise is additive and normally distributed, with $\Sigma = 100I$. The objective function is a 10-dimensional ($d = 10$) quadratic, $f(x) = 1/2\ x^T Ax$, where A is an (anisotropic) Hessian matrix. While this is a simple model, every twice-differentiable convex function can in fact be approximated by a quadratic function near an isolated minimizer. Figure 4 illustrates the results of using COCO$_K$ as a plug-in to SGD. We observe an initial *bias regime*, where the algorithms converge similarly, that is successively slowed down and eventually leads to a stagnation, usually called the *variance regime*. We see that COCO leads to improved performance in the variance regime without compromising the bias regime and that the improvement increases with the number K of gradients simultaneously denoised.

Real Data. We test the robustness of plugging in COCO in SGD in real logistic regression problems using the “fourclass” dataset ($n = 862$ data points of dimension $d = 2$) and “mushrooms” dataset ($n = 8124$, $d = 112$) [4]. For the “mushrooms” dataset, we added a Tikhonov regularization term to the objective function, which is formulated according to the typical finite-sum setting. At each gradient evaluation, one of those examples is randomly picked, from which we compute a noisy gradient of the objective. This setup falls out of the assumptions for COCO, since the sampled gradients are not independent and the noise is not additive and normally distributed. The results are also shown in Figure 4. For both datasets, consistent variance improvements are observed for SGD with increasing number K of gradients simultaneously denoised. In contrast to the results for the “fourclass” dataset, we do not detect any significant bias delay in the ‘mushrooms” dataset performance.

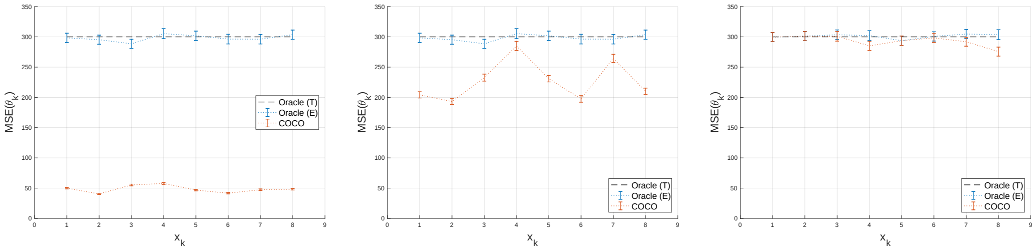

Figure 2: Measuring the amount of noise reduction as a function of proximity of the points. $\widehat{\mathrm{MSE}}(\hat{\theta}_k)$ (COCO), $\widehat{\mathrm{MSE}}(g_k)$ (Oracle (E)), both estimated via Monte-Carlo method ($N = 1000$), and $\mathrm{MSE}(g_k)$ (Oracle (T)) for 8-point configuration in $\mathbb{R}^3$. Each point is sampled from the cube $x_k \in [-l,\ l]^3$. *Left*: $l = 10$; *Center*: $l = 100$; *Right*: $l = 1000$.

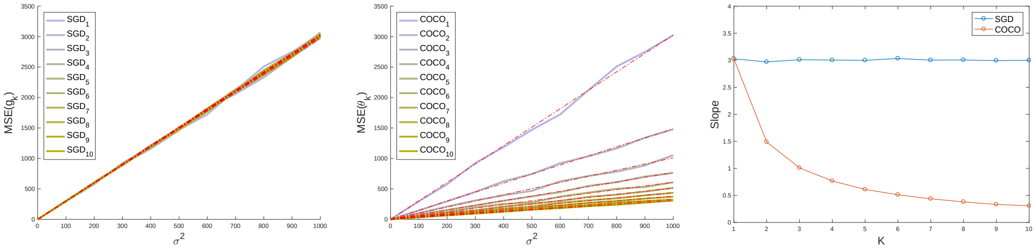

Figure 3: Measuring the amount of noise reduction as a function of the noise level. $\widehat{\mathrm{MSE}}(g_k)$ (*Left*) and $\widehat{\mathrm{MSE}}(\hat{\theta}_k)$ (*Center*), both estimated via Monte-Carlo method ($N = 1000$), as functions of the noise variance σ^2, for several numbers of points considered ($1 \leq K \leq 10$). For each simulation, a different set of K points is sampled from an uniform distribution over the cube $x_k \in [-5,\ 5]^3$. The dashed-dotted red lines result from linear regressions with intercept fixed at 0. *Right:* each of the regressed slopes is depicted as a function of the number of points.

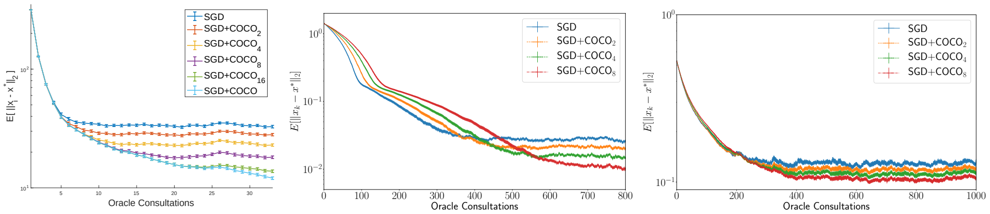

Figure 4: COCO denoiser plugged in SGD. The width of each marker represents the standard error of the mean. *Left:* synthetic problem satisfying the noise model. $E[\|x_i - x^*\|]$ is averaged over 100 runs. *Center:* 100 runs of a logistic regression problem built on the *fourclass* dataset [4]. *Right:* 50 runs of a logistic regression problem built on the *mushrooms dataset* [4].

References

[1] Amir Beck and Marc Teboulle. A fast dual proximal gradient algorithm for convex minimization and applications. *Operations Research Letters*, 2014.

[2] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, Cambridge, UK ; New York, 2004.

[3] Sébastien Bubeck. Convex Optimization: Algorithms and Complexity. *Foundations and Trends in Machine Learning*, 2015. URL http://www.nowpublishers.com/article/Details/MAL-050.

[4] Chih-Chung Chang and Chih-Jen Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2011.

[5] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives. *Advances in Neural Information Processing Systems*, 2014.

[6] Timothy Dozat. Incorporating Nesterov Momentum into Adam. *International Conference on Learning Representations Workshop*, 2016.

[7] John Duchi, Elad Hazan, and Yoram Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 2011.

[8] Geoff Hinton and Tijmen Tieleman. Lecture 6.5 - RMSprop: Divide the Gradient by a Running Average of Its Recent Magnitude. *COURSERA: Neural Networks for Machine Learning*, 4(2): 26–31, 2012. URL https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[9] Prateek Jain, Sham M. Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. Accelerating Stochastic Gradient Descent for Least Squares Regression. *Conference On Learning Theory*, 2018.

[10] Rie Johnson and Tong Zhang. Accelerating Stochastic Gradient Descent using Predictive Variance Reduction. *Advances in Neural Information Processing Systems*, 2013.

[11] Ellango Jothimurugesan, Ashraf Tahmasbi, Phillip Gibbons, and Srikanta Tirthapura. Variance-Reduced Stochastic Gradient Descent on Streaming Data. *Advances in Neural Information Processing Systems*, 2018.

[12] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*, 2015.

[13] Boris Polyak and Anatoli Juditsky. Acceleration of Stochastic Approximation by Averaging. *SIAM Journal on Control and Optimization*, 1992.

[14] Herbert Robbins and Sutton Monro. A Stochastic Approximation Method. *The Annals of Mathematical Statistics*, 1951.

[15] Mark Schmidt, Nicolas Le Roux, and Francis Bach. Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 2017.

[16] Roman Vershynin. Estimation in High Dimensions: A Geometric Perspective. In *Sampling Theory, a Renaissance: Compressive Sensing and Other Developments*, Applied and Numerical Harmonic Analysis, pages 3–66. Springer International Publishing, Cham, 2015.

[17] Matthew D. Zeiler. ADADELTA: An Adaptive Learning Rate Method. *arXiv:1212.5701*, 2012.

Appendix A. Closed-form solution for COCO$_2$

The result for the QCQP in (3), obtained by instantiating the Karush-Kuhn-Tucker conditions, is given by the following theorem.

Theorem 1 *For $K = 2$, the solution to the optimization problem (3) is given by:*
If $\|g_1 - g_2\|^2 \leq L\ \langle g_1 - g_2,\ x_1 - x_2 \rangle$,

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

If $\|g_1 - g_2\|^2 > L\ \langle g_1 - g_2,\ x_1 - x_2 \rangle$,

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

The solution above has an intuitive interpretation: when the 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 coincide with the estimated ones; when they 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 the feasible set, which is a ball. Despite its simplicity, this closed-form solution is of the utmost relevance in practice, since, as shown in Section 4, COCO leads to significant improvements in stochastic optimization, even for this simple case of $K = 2$. Therefore, when available, using the closed-form is obviously preferable to the FDPG method, since it has a total runtime of only $O(d)$ and much lower memory requirements: the closed-form solution only requires two points and respective gradients, both d-dimensional vectors, to be kept in memory, an $O(Kd)$ memory overhead.

Proof For $K = 2$, we first formulate the problem in (3) for a generic Σ:

$$\begin{aligned} \min_{\theta_1,\theta_2} \quad & (g_1 - \theta_1)^T\Sigma^{-1}(g_1 - \theta_1) + (g_2 - \theta_2)^T\Sigma^{-1}(g_2 - \theta_2) \\ \text{subject to} \quad & \|\theta_1 - \theta_2\|^2 - L\langle \theta_1 - \theta_2,\ x_1 - x_2\rangle \leq 0. \end{aligned}$$

In order to solve this problem, the Karush-Kuhn-Tucker (KKT) conditions will now be used. It can be observed that there are no equality constraints. We have:

$$\begin{aligned} f(\theta_1, \theta_2) &= (g_1 - \theta_1)^T\Sigma^{-1}(g_1 - \theta_1) + (g_2 - \theta_2)^T\Sigma^{-1}(g_2 - \theta_2) \\ f_1(\theta_1, \theta_2) &= \|\theta_1 - \theta_2\|^2 - L\langle \theta_1 - \theta_2,\ x_1 - x_2\rangle. \end{aligned}$$

Since both functions are differentiable and convex, we can use $\partial(f(x)) = \{\nabla f(x)\}$[4]. This can be applied for simplification of the stationarity condition, through the linearity of the gradient

[4] $\partial(\cdot)$ denotes the subdifferential operator. For a continuous function $f : \mathbb{R}^d \to \mathbb{R}$, $c \in \mathbb{R}^d$ is a subgradient of f at $x \in \mathbb{R}^d$ if and only if $f(y) - f(x) \geq c^T(y - x)$, with $y \in \mathbb{R}^d$. The set of all the subgradients of f at x is called the subdifferential of f at x, $\partial(f(x))$.

operator. Therefore, the KKT conditions yield the following system of equations:

$$\begin{cases} 2\Sigma^{-1}(\hat{\theta}_1 - g_1) + \mu_1 \left[2(\hat{\theta}_1 - \hat{\theta}_2) - L(x_1 - x_2)\right] = 0 & [\ \textit{i.}\ \text{Stationarity in order to } \hat{\theta}_1\ ] \\ 2\Sigma^{-1}(\hat{\theta}_2 - g_2) - \mu_1 \left[2(\hat{\theta}_1 - \hat{\theta}_2) - L(x_1 - x_2)\right] = 0 & [\ \textit{ii.}\ \text{Stationarity in order to } \hat{\theta}_2\ ] \\ \mu_1 \left(\|\hat{\theta}_1 - \hat{\theta}_2\|^2 - L\langle\hat{\theta}_1 - \hat{\theta}_2,\ x_1 - x_2\rangle\right) = 0 & [\ \textit{iii.}\ \text{Complementary Slackness}\ ] \\ \|\hat{\theta}_1 - \hat{\theta}_2\|^2 - L\langle\hat{\theta}_1 - \hat{\theta}_2,\ x_1 - x_2\rangle \leq 0 & [\ \textit{iv.}\ \text{Primal Feasibility}\ ] \\ \mu_1 \geq 0 & [\ \textit{v.}\ \text{Dual Feasibility}\ ]. \end{cases}$$

From *iii.*, two cases must be considered:

- $\boldsymbol{\mu_1 = 0}$: In this case, from complementary slackness (*iii.*), $\|\hat{\theta}_1 - \hat{\theta}_2\|^2 \leq L\langle\hat{\theta}_1 - \hat{\theta}_2,\ x_1 - x_2\rangle$. In that case, from *i.* and *ii.*, it is easy to conclude that $\hat{\theta}_1 = g_1$ and $\hat{\theta}_2 = g_2$. Therefore, we note that this happen when $\|g_1 - g_2\|^2 \leq L\langle g_1 - g_2,\ x_1 - x_2\rangle$.
- $\boldsymbol{\mu_1 > 0}$: In that case, from complementary slackness (*iii.*), $\|\hat{\theta}_1 - \hat{\theta}_2\|^2 = L\langle\hat{\theta}_1 - \hat{\theta}_2,\ x_1 - x_2\rangle$. By summing *i.* and *ii.*:

$$\hat{\theta}_1 + \hat{\theta}_2 = g_1 + g_2.$$

This equality is particularly interesting and further developed in the proof of Theorem 2. By replacing it in *i.* and *ii.*, we obtain:

$$\begin{aligned} \hat{\theta}_1 &= (\Sigma^{-1} + 2\mu_1 I)^{-1}[(\Sigma^{-1} + \mu_1 I)g_1 + \mu_1 g_2 + \mu_1 \frac{L}{2}(x_1 - x_2)] \\ \hat{\theta}_2 &= (\Sigma^{-1} + 2\mu_1 I)^{-1}[\mu_1 g_1 + (\Sigma^{-1} + \mu_1 I)g_2 - \mu_1 \frac{L}{2}(x_1 - x_2)]. \end{aligned} \tag{5}$$

Then, by replacing those results in $\|\hat{\theta}_1 - \hat{\theta}_2\|^2 = L\langle\hat{\theta}_1 - \hat{\theta}_2,\ x_1 - x_2\rangle$, it yields the following expression:

$$I\mu_1^2 + \Sigma^{-1}\mu_1 - (\Sigma^{-1})^2 C = 0, \tag{6}$$

with $C = (\|g_1 - g_2\|^2 - L\langle g_1 - g_2,\ x_1 - x_2\rangle)/(L^2\|x_1 - x_2\|^2)$. Note that $C \geq 0$, since, otherwise, we would have $\|g_1 - g_2\|^2 < L\langle g_1 - g_2,\ x_1 - x_2\rangle$ and we would be in the case of $\mu_1 = 0$. By considering that $\Sigma = \sigma^2 I$, the equation above yields for each diagonal entry:

$$\mu_1^2 + \frac{1}{\sigma^2}\mu_1 - \left(\frac{1}{\sigma^2}\right)^2 C = 0. \tag{7}$$

The non-diagonal entries are not informative, as they are all zero. The only solution of (7) that respects dual feasibility (*v.*) is:

$$\mu_1 = \frac{1}{\sigma^2}\left(\frac{-1 + \sqrt{1 + 4C}}{2}\right).$$

Replacing this value of μ_1 in (5), we obtain the intended result. ■

Appendix B. Detailed Derivation of the FDPG Method

B.1. Reformulation of the Problem

We start by multiplying the objective function in (3) by $1/2$ for the sake of simplicity in the next steps, yielding the following problem:

$$\begin{aligned} \min_{\theta_1,\ldots,\theta_K} \quad & \frac{1}{2}\sum_{k=1}^{K}\|g_k-\theta_k\|^2 \\ \text{subject to} \quad & \|\theta_m-\theta_l-\frac{L}{2}(x_m-x_l)\| \le \frac{L}{2}\|x_m-x_l\|, \quad 1\le m<l\le K, \end{aligned}$$

This problem remains the same as the one provided in (3), where the new form for the constraints is obtained by completing the square in the expression from the original formulation:

$$\begin{aligned} & \frac{1}{L}\|\theta_m-\theta_l\|^2 \le (\theta_m-\theta_l)^T(x_m-x_l) \\ \Leftrightarrow\ & \|\theta_m-\theta_l\|^2 - L(\theta_m-\theta_l)^T(x_m-x_l) + \frac{L}{4}\|x_m-x_l\|^2 - \frac{L}{4}\|x_m-x_l\|^2 \le 0 \qquad (8) \\ \Leftrightarrow\ & \|\theta_m-\theta_l-\frac{L}{2}(x_m-x_l)\|^2 \le \|\frac{L}{2}(x_m-x_l)\|^2 \\ \Leftrightarrow\ & \|\theta_m-\theta_l-\frac{L}{2}(x_m-x_l)\| \le \|\frac{L}{2}(x_m-x_l)\|, \end{aligned}$$

where in (8) we add and subtract $L\|x_m-x_l\|^2/4$ and all the other steps are simple manipulations. Note that, in this case, $\theta_m-\theta_l \in \mathcal{B}\left(L(x_m-x_l)/2,\ L\ \|x_m-x_l\|/2\right)$[5].

Now, performing the change of variables $\alpha_k=\theta_k-g_k$, the problem becomes:

$$\begin{aligned} \min_{\alpha_1,\ldots,\alpha_K} \quad & \frac{1}{2}\sum_{k=1}^{K}\|\alpha_k\|^2 \\ \text{subject to} \quad & \|\alpha_m-\alpha_l+c_{ml}\| \le r_{ml}, \quad 1\le m<l\le K, \end{aligned}$$

where $c_{ml}=(g_m-(L/2)\ x_m)-(g_l-(L/2)\ x_l)$ and $r_{ml}=L\|x_m-x_l\|/2$.

The indicator function can be defined as

$$\mathbf{1}_E(x)=\begin{cases}0 & \text{if } x\in E\\ \infty & \text{if } x\notin E.\end{cases}$$

Using this definition, the primal problem can be finally formulated as

$$\min_{\alpha} \quad \underbrace{\frac{1}{2}\|\alpha\|^2}_{p(\alpha)} + \underbrace{\mathbf{1}_{\mathcal{B}}(A\alpha+c)}_{q(A\alpha)},$$

where $\alpha=[\alpha_1,\ \alpha_2,\ \ldots,\ \alpha_K]^T$, $A\alpha=[\alpha_1-\alpha_2,\ \alpha_1-\alpha_3,\ \ldots,\ \alpha_1-\alpha_K,\ \alpha_2-\alpha_3,\ \ldots,\ \alpha_{K-1}-\alpha_K]^T$, $c=[c_{12},\ c_{13},\ \ldots,\ c_{1K},\ c_{23},\ \ldots,\ c_{K-1K}]^T$ and $\mathcal{B}=\mathcal{B}(0,\ r_{12})\times\mathcal{B}(0,\ r_{13})\times\ldots\times\mathcal{B}(0,\ r_{1K})\times\mathcal{B}(0,\ r_{23})\times\ldots\times\mathcal{B}(0,\ r_{K-1K})$.

[5] The notation $\mathcal{B}(c,r)$ denotes the set of points within a ball centered at c and of radius r, *i.e.*, $\mathcal{B}(c,r)=\{x\in\mathbb{R}^n : \|x-c\|\le r\}$.

In this formulation, we want to minimize the sum of two convex functions, where the first is differentiable and the second is non-differentiable, but still closed[6]. This is the setup to which the iterative shrinkage-thresholding algorithms (ISTA) are designed for. In particular, when the non-differentiable function is a simple indicator function, that method can be interpreted as the Projected Gradient Descent. However, in this formulation, that function is composed with a linear map A, case in which there is no closed-form for the proximity operator.

Given this, a reformulation using Lagrange duality is used. First, the problem can be rewritten as:

$$\begin{aligned} \min_{\alpha,\beta} \quad & p(\alpha)+q(\beta) \\ \text{subject to} \quad & A\alpha=\beta. \end{aligned}$$

It is possible to write the Lagrangian for the reformulated problem:

$$\begin{aligned} L(\alpha,\beta,s) &= p(\alpha)+q(\beta)+s^T(A\alpha-\beta) \\ &= p(\alpha)+s^TA\alpha+q(\beta)-s^T\beta. \end{aligned}$$

The Lagrange dual function can be computed:

$$\begin{aligned} L(s) &= \inf_{\alpha,\beta} L(\alpha,\beta,s) \\ &= \inf_{\alpha}\left(p(\alpha)+s^TA\alpha\right)+\inf_{\beta}\left(q(\beta)-s^T\beta\right). \end{aligned}$$

Thus,

$$-L(s)=\sup_{\alpha}\left((-A^Ts)^T\alpha-p(\alpha)\right)+\sup_{\beta}\left(s^T\beta-q(\beta)\right).$$

By definition, for a generic function, its (Fenchel) conjugate is defined as $f^*(s)=\sup\limits_{x}\,(s^Tx-f(x))$. Therefore, it is possible to conclude that:

$$-L(s)=p^*(-A^Ts)+q^*(s).$$

It remains to obtain the specific form of $p^*(s)$ and $q^*(s)$. Regarding the former:

$$\begin{aligned} p^*(s) &= \sup_{\alpha}\left(s^T\alpha-\frac{1}{2}\|\alpha\|^2\right) \\ &= \frac{1}{2}\|s\|^2, \end{aligned}$$

where the second equality easily comes from differentiating $s^T\alpha-1/2\ \|\alpha\|^2$ with respect to α and equating to zero. Therefore, the value obtained for α is then replaced on the original expression.

[6] A function $f:\mathbb{R}^d\to\mathbb{R}$ is said to be closed if for each $\alpha\in\mathbb{R}$, the sublevel set $\{x\in\mathrm{dom}f|f(x)\leq a\}$ is a closed set.

Regarding q^*:

$$\begin{aligned} q^*(s) &= \sup_\beta \big(s^T\beta - \mathbf{1}_{\mathcal{B}}(\beta + c)\big) \\ &= \sup_\beta \big\{s^T\beta : \beta + c \in \mathcal{B}\big\} \\ &= \sup_\beta \left\{ \sum_{1\le m<l\le K} s_{ml}^T\beta_{ml} : \beta_{ml} + c_{ml} \in \mathcal{B}(0, r_{ml}) \right\} \\ &= \sum_{1\le m<l\le K} \sup_\beta \big\{s_{ml}^T\beta_{ml} : \|\beta_{ml} + c_{ml}\| \le r_{ml}\big\} \\ &= \sum_{1\le m<l\le K} r_{ml}\|s_{ml}\| - s_{ml}^T c_{ml}. \end{aligned}$$

where the last step is obtained via:

$$\begin{aligned} \sup_b \big\{s^T b : \|b - (-c)\| \le r\big\} &= \sup_b \big\{s^T(-c + u) : \|u\| \le r\big\} \\ &= -s^T c + \sup_b \big\{s^T u : \|u\| \le r\big\} \\ &= -s^T c + r\|s\|, \end{aligned}$$

and the equality $\sup_b \big\{s^T u : \|u\| \le r\big\} = r\|s\|$ is obtained (we assume s different from 0, otherwise, the equality is trivial):

(1) by picking $u = r\frac{s}{\|s\|}$ (note that $\|u\| \le r$), we have $s^T u = r\|s\|$.
This shows $\sup\{s^T u : \|u\| \le r\} \ge r\|s\|$;

(2) from Cauchy-Schwartz inequality: $s^T u \le \|s\|\|u\|$. Since $\|u\| \le r$: $s^T u \le r\|s\|$.
So, $\sup_b\{s^T u : \|u\| \le r\} \le r\|s\|$.

From (1) and (2), we obtain the intended result. Therefore, the minimization problem can be rewritten in the following form:

$$\min_s \; \underbrace{\frac{1}{2}\| - A^T s\|^2}_{p^*(-A^T s)} + \underbrace{\sum_{1\le m<l\le K} r_{ml}\|s_{ml}\| - s_{ml}^T c_{ml}}_{q^*(s)}. \tag{9}$$

At this point, the linear mapping A has now been transferred to the differentiable term. This change allows us now to find a closed-form expression for the proximity operator of $q^*(s)$, as the gradient of the first term can still be computed even considering its composition with A^T.

B.2. Proximity Operator Computation

By definition, the proximity operator of a generic closed, convex function f is:

$$\operatorname{prox}_f(x) = \operatorname*{argmin}_u \frac{1}{2}\|u - x\|^2 + f(u).$$

We are interested in obtaining $\text{prox}_{\mu q^*}(s)$, for any given $\mu > 0$. Thus:

$$\begin{aligned}\text{prox}_{\mu q^*}(s) &= s - \text{prox}_{(\mu q^*)^*}(s) && (10)\\ &= s - \text{prox}_{(q^*)^* \cdot \mu}(s) && (11)\\ &= s - \text{prox}_{q \cdot \mu}(s), && (12)\end{aligned}$$

where in (10) it is applied the well-known Moreau identity $\text{prox}_f(x) = x - \text{prox}_{f^*}(x)$; in (11), we used $(\mu f)^*(x) = f^* \cdot \mu\ (x) = \mu f^*(x/\mu)$ and, in (12), the property $(f^*)^* = f$, which holds for any closed, convex function. Now, note that:

$$\begin{aligned}q \cdot \mu\ (s) &= \mu\, q\left(\frac{s}{\mu}\right)\\ &= \mu\, \mathbf{1}_{\mathcal{B}}\left(\frac{s}{\mu} + c\right)\\ &= \mathbf{1}_{\mathcal{B}}\left(\frac{s}{\mu} + c\right), && (13)\end{aligned}$$

since, in (13), μ can be dropped as $\mathbf{1}_{\mathcal{B}}$ returns either 0 or ∞. Therefore,

$$\begin{aligned}\text{prox}_{\mu q^*}(s) &= s - \underset{u}{\text{argmin}}\left(\frac{1}{2}\|u - s\|^2 + \mathbf{1}_{\mathcal{B}}\left(\frac{u}{\mu} + c\right)\right)\\ &= s - \mu\left(\underset{v}{\text{argmin}}\left(\frac{1}{2}\|\mu(v - c) - s\|^2 + \mathbf{1}_{\mathcal{B}}(v)\right) - c\right) && (14)\\ &= s - \mu\left(\underset{v \in \mathcal{B}}{\text{argmin}}\left(\frac{1}{2}\|v - (c + \frac{s}{\mu})\|^2\right) - c\right)\\ &= s - \mu\ (\ v_{\text{proj}} - c\ ),\end{aligned}$$

where the change of variable $v = \frac{u}{\mu} + c$ was used in (14) and the orthogonal projection of $c_{ml} + s_{ml}/\mu$ onto the ball $\mathcal{B}(0,\ r_{ml})$, with $1 \leq m < l \leq K$, is denoted by v_{ml}, whose stacking results in $v_{\text{proj}} = \underset{v \in \mathcal{B}}{\text{argmin}}\ \|v - (c + s/\mu)\|^2$.

B.3. Fast Dual Proximal Gradient Method

Recalling (9), we now have a first term, $p^*(-A^T s)$, differentiable, for which the gradient has a closed-form and a second term, $q^*(s)$, non-differentiable but for which we can compute also a closed-form and inexpensive proximity operator. We are now in place to apply ISTA, where the iterates are generated by alternating between taking a gradient step of the differentiable function and taking a proximal step.

The gradient for $p^*(-A^T s)$ can be easily computed: $\nabla_s\ p^*(-A^T s) = AA^T s$. Furthermore, from this expression is straightforward to observe that the first term, $p^*(-A^T s)$, is necessarily L-smooth, with Lipschitz constant $L_{p^*} = \sigma_{\max}(A)^2$. Given this, not only the optimal step size for a gradient update is known ($\gamma = 1/L_{p^*}$), but also, just as happened with first-order algorithms (for differentiable functions) in the deterministic convex setting, it is possible to accelerate the ISTA resorting to a Nesterov acceleration similar scheme, *i.e.*, enabling momentum to contribute in the generated iterates. The accelerated version of ISTA is known as FISTA. Moreover, this perspective

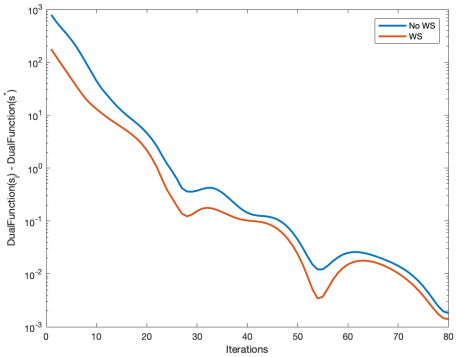

Figure 5: Dual objective function obtained for the different iterates of the COCO$_K$ solution method, using the warm-starting procedure (WS) and without using it (No WS). DualFunction(s) is the dual objective function $p^*(-A^T s)+q^*(s)$, s_i is the vector that results from stacking the different s_{ml} at iteration i, and s^* is the corresponding optimal vector.

of applying FISTA to the dual problem is a well-studied technique, already introduced in this paper as the FDPG method (applied to COCO in Algorithm 1). Through FDPG, it is possible to find an approximation of the optimal solution of the dual problem, s^*. However, we are interested in recovering the solution of the primal problem, α^*, which, nevertheless, can be easily obtained through $\alpha^* = -A^T s^*$. Consequently, the gradient estimates are recovered as $\hat{\theta}_k = \alpha_k^* + g_k$.

Strong duality, *i.e.*, $p(\alpha^*) + q(A\alpha^*) = -\big(p^*(-A^T s^*) + q(s^*)\big)$, holds for this convex optimization problem. For example, a Slater point can be easily obtained by considering $\hat{\theta}_k = L/2\ x_k$, assuming the iterates to be different from each other ($x_i \neq x_j$ if $i \neq j$). This is expectable if we presume that these iterates are generated through a stochastic first-order method.

B.4. Warm-starting

By coupling a baseline algorithm with COCO$_K$, 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 consider taking advantage of the COCO$_K$ solution obtained for the previous iterate to obtain a new solution faster. We propose a warm-starting procedure for the COCO$_K$ solution method (FDPG). 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 a default value. Our implementation of this warm-starting procedure allows the iterative method to start with a much better guess of s^*, thereby achieving satisfactory approximate solutions faster (see Figure 5).

Appendix C. Theoretical Analysis

In this section, we prove a relationship between the COCO output and its input, prove that the COCO gradient estimates jointly outperform the oracle, and show that the co-coercivity constraints become tighter for closer points. For all the theorem proofs below, the starting point is the COCO denoiser formulation for a generic Σ:

$$\begin{aligned} \min_{\theta_1,\ldots,\theta_K} \quad & \frac{1}{2}\sum_{k=1}^{K}(g_k-\theta_k)^T\Sigma^{-1}(g_k-\theta_k) \\ \text{subject to} \quad & \frac{1}{L}\|\theta_m-\theta_l\|^2 \le \langle \theta_m-\theta_l, x_m-x_l\rangle,\ 1\le m<l\le K. \end{aligned} \tag{15}$$

C.1. Relation between the oracle and COCO estimates

We start by relating the centroid of the noisy gradients (COCO input) with the centroid of the denoised ones (COCO output), via the following theorem, which holds for generic Σ.

Theorem 2 *The gradients estimated by COCO and the raw observations have the same centroid:*

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

Intuitively, we expect COCO to keep the centroid and reduce the input fluctuation.
Proof From the COCO denoiser formulation for generic Σ and K points considered (15), the KKT conditions yield K stationarity equations. Its i-th equation is of the form:

$$2\Sigma^{-1}(\hat{\theta}_i-g_i)+\sum_{j=1,j\neq i}^{K}\mu_{ij}[\,2(\hat{\theta}_i-\hat{\theta}_j)-L(x_i-x_j)\,]=0,$$

Summing the K equations, all the constraint terms cancel out pairwisely, yielding:

$$\sum_{i=1}^{K}2\Sigma^{-1}(\hat{\theta}_i-g_i)=0 \Leftrightarrow 2\Sigma^{-1}\sum_{i=1}^{K}(\hat{\theta}_i-g_i)=0 \Leftrightarrow \sum_{i=1}^{K}\hat{\theta}_i=\sum_{i=1}^{K}g_i.$$

■

C.2. Mean squared error (MSE) of COCO estimates

The following theorem states that the COCO estimator outperforms the oracle in terms of MSE ($\text{MSE}(\hat{\theta})=E\left[||\hat{\theta}-\nabla f||^2\right]$, $\text{MSE}(g)=E\left[||g-\nabla f||^2\right]$, with ∇f collecting the gradients $\nabla f(x_k)$).

Theorem 3 *The following inequality holds:*

$$\text{MSE}(\hat{\theta}) \le \text{MSE}(g). \tag{16}$$

Proof The Orthogonal Projection operator on a set S is defined as

$$\begin{aligned} P_S(x) : \quad \mathbb{R}^d &\to \mathbb{R}^d \\ x &\mapsto \operatorname*{argmin}_{y \in S} \|x - y\|. \end{aligned}$$

In the case in which $S \subset \mathbb{R}^d$ is closed and convex, the following property holds:

$$\|P_S(a) - P_S(b)\| \leq \|a - b\|.$$

Let also S be the feasible set of the problem in (15). Note that, in that case, S is a convex and closed set as it results from the intersection of ellipsoids, which are convex and closed sets themselves. Moreover, when $\Sigma = \sigma^2 I$, (15) yields:

$$\hat{\theta} = \operatorname*{argmin}_{\theta \in S} 1/\sigma^2 \, \|\theta - g\|^2 = \operatorname*{argmin}_{\theta \in S} \|\theta - g\| = P_S(g)$$

Noting that $\nabla f = P_S(\nabla f)$ since $\nabla f \in S$, *i.e.*, the true gradients of an L-smooth and convex function are necessarily co-coercive[7], it follows:

$$\|\hat{\theta} - \nabla f\| = \|P_S(g) - P_S(\nabla f)\| \leq \|g - \nabla f\|. \tag{17}$$

Squaring both sides of the inequality in (17) and applying the Expectation operator, the result intended is obtained. ■

C.3. COCO constraints tightness

Each constraint in COCO involves a pair of gradients, g_i and g_j. If they are not co-coercive, COCO outputs co-coercive estimates $\hat{\theta}_i$ and $\hat{\theta}_j$. It is thus interesting to know how often g_i and g_j do not respect 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, $g_1 \perp\!\!\!\perp g_2$[8] 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}:

$$p_{\text{inactive}} = \Phi\left(\frac{L\Delta_x - \Delta_{\nabla f}}{\sqrt{2}\sigma}\right) - \Phi\left(\frac{-\Delta_{\nabla f}}{\sqrt{2}\sigma}\right), \tag{18}$$

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

Proof Note that $g_i \sim \mathcal{N}(\nabla f(x_i), \sigma^2)$. Moreover, the co-coercivity constraint between g_1 and g_2 is inactive when:

$$\begin{aligned} \|g_1 - g_2\|^2 < L\, \langle g_1 - g_2,\; x_1 - x_2 \rangle &\Leftrightarrow (g_1 - g_2)^2 - L(g_1 - g_2)(x_1 - x_2) < 0 \\ &\Leftrightarrow (g_1 - g_2)(g_1 - g_2 - L(x_1 - x_2)) < 0 \\ &\Leftrightarrow 0 < g_1 - g_2 < L(x_1 - x_2). \end{aligned}$$

[7] Note that this statement is only true for $L \geq L_{\text{real}}$, where L_{real} denotes the minimal Lipschitz constant of ∇f.

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

Therefore, noticing that $g_1 - g_2 \sim \mathcal{N}(\nabla f(x_1) - \nabla f(x_2), 2\sigma^2)$ and defining $\Delta_x = x_1 - x_2$ and $\Delta_{\nabla f} = \nabla f(x_1) - \nabla f(x_2)$:

$$\begin{aligned} P(\|g_1 - g_2\|^2 < L \langle g_1 - g_2,\ x_1 - x_2 \rangle) &= P(0 < g_1 - g_2 < L\Delta x) \\ &= \Phi\left(\frac{L\Delta_x - \Delta_{\nabla f}}{\sqrt{2}\sigma}\right) - \Phi\left(\frac{-\Delta_{\nabla f}}{\sqrt{2}\sigma}\right) \\ &= p_{\text{inactive}}. \end{aligned}$$

■

Appendix D. Estimator Properties

D.1. Extension of Theorem 3

As a consequence of Theorem 3, we analyze to what extent the COCO estimator outperforms the oracle. In fact, it is possible to obtain a closed-form result for the $\mathrm{MSE}(g)$ for a general number of points considered, K, a general dimension d and $\Sigma = \sigma^2 I$: $\mathrm{MSE}(g) = Kd\sigma^2$. Regarding $\mathrm{MSE}(\hat{\theta})$, even though without a closed-form solution, we were able to observe its direct dependence on the COCO constraints tightness, as represented in Figure 6.

- For $\Delta_x = 0$, all the curves have $p_{\text{active}} = 1$ and $\mathrm{MSE}(\hat{\theta}) = 100 = \sigma^2/2$. This recovers a well known result for the average of K random variables with Gaussian distributions: their MSE[9] is σ^2/K. In fact, when $\Delta_x = 0$, COCO_K denoiser outputs the average of the observed gradients (recall closed-form solution for COCO_2 - Theorem 1). Furthermore, COCO denoiser can therefore be considered an extension for the variance reduction through averaging method, but which tolerates samples from different points. This can be viewed as one of the main advantages of COCO;
- When the L is underestimated ($\Delta_L < 0$), the $\mathrm{MSE}(\hat{\theta})$ is not guaranteed to be lower than $\mathrm{MSE}(g)$. Nevertheless, there still is a range of Δ_x where $\mathrm{MSE}(\hat{\theta}) \leq \mathrm{MSE}(g)$. The more underestimated L is, the smaller this region becomes. This observation not only recalls that the result from Theorem 3 only holds for $\Delta_L \geq 0$, but also reinforces the importance of ensuring that the L considered for COCO is an upper bound for L_{real};
- When the L is perfectly estimated ($\Delta_L = 0$), just as the p_{active} tends to an intermediate value, so it happens with $\mathrm{MSE}(\hat{\theta})$. This is the ideal situation, as $\mathrm{MSE}(\hat{\theta})$ is minimal for every Δ_L. Moreover, note that when the p_{active} curve stabilizes, the $\mathrm{MSE}(\hat{\theta})$ also stabilizes, reinforcing the expected relation between those curves;
- When the L is overestimated ($\Delta_L > 0$), just as p_{active} tends to 0, the $\mathrm{MSE}(\hat{\theta})$ also tends to the $\mathrm{MSE}(g)$ reference curve. Moreover, it is possible to see that when p_{active} stabilizes around 0, so it happens to $\mathrm{MSE}(\hat{\theta})$ 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";
- Regarding the noise variance, σ^2, it should be stated that, as previously seen in Figure **??**, its increase would shift the stabilization of the curves from the $\Delta_L > 0$ cases towards higher Δ_x.

[9] The MSE corresponds to the variance of an unbiased estimator, which is the case of the average of random variables following normal distributions.

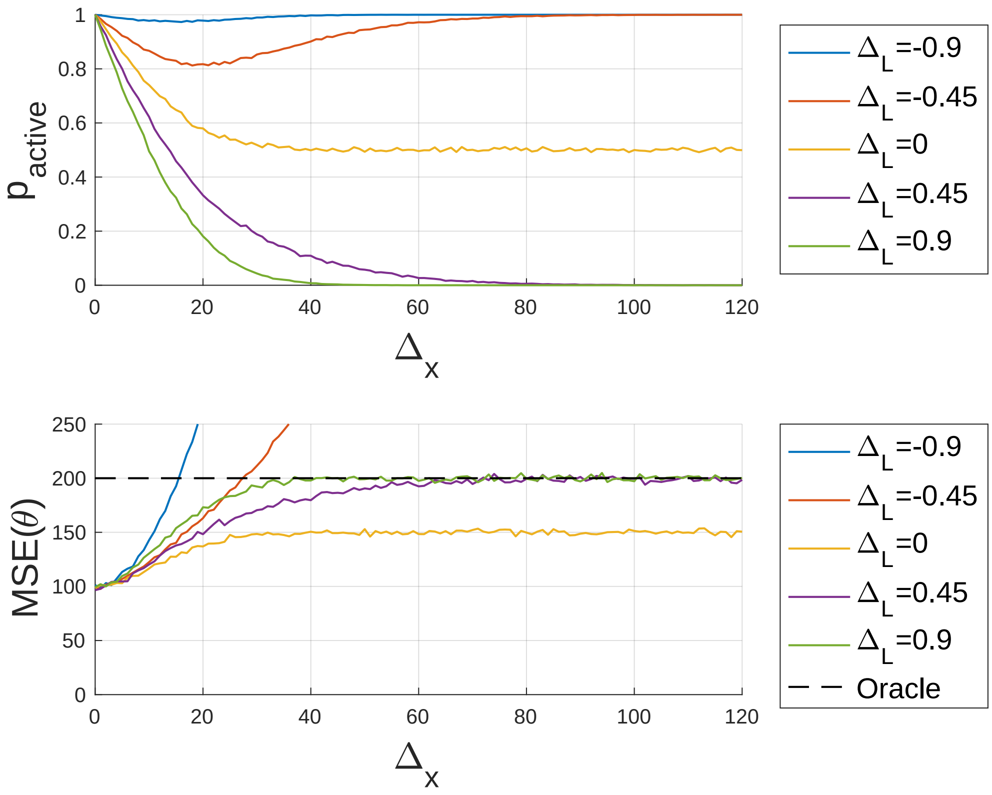

Figure 6: *Top*: Experimental plot for p_{active} as a function of Δ_x for different values of Δ_L. *Bottom*: Computed $\text{MSE}(\hat{\theta})$ (number of Monte-Carlo simulations: $N = 10000$). We have $\text{MSE}(g) = 200$, represented as a dashed line. Both plots are obtained for $f(x) = x^2/2$, with one point fixed at $x_1 = 0$ and a variable point at $x_2 = \Delta_x$. The oracle provides gradient estimates with additive Gaussian noise with $\Sigma = \sigma^2 = 100$.

D.2. Extension to COCO Elementwise MSE Improvement

In this Section, we provide additional empirical evidence obtained as far as the elementwise MSE is concerned, by providing more instances of the results shown in the main body (in Figure 2). Those results are depicted in Figure 7. In particular, we observe that $\mathrm{MSE}(\hat{\theta}_k) \leq \mathrm{MSE}(g_k)$ for every point in every tested setting.

We also emphasize that $\mathrm{MSE}(\hat{\theta})$ does not distribute evenly among the different points, as the $\mathrm{MSE}(\hat{\theta}_k)$ varies from point to point. In particular, points which have other points closer have lower $\mathrm{MSE}(\hat{\theta}_k)$, whereas more isolated points show higher $\mathrm{MSE}(\hat{\theta}_k)$. This can be easily assessed by comparing the relative positions of the points represented in Figure 8 with the $\mathrm{MSE}(\hat{\theta}_k)$ obtained for each of them (center plot from Figure 2).

Appendix E. Stochastic Optimization - Adam and STRSAGA

In this section, we repeat the same experiments as the ones in Section 4 but by plugging COCO denoiser in Adam and STRSAGA, instead of SGD. The results obtained for Adam are represented in Figure 9 and for STRSAGA in Figure 10. The STRSAGA algorithm is not applicable to the synthetic dataset as, by being the streaming counterpart of SAGA [5], still requires the objective function to be a finite-sum of sub-functions generated by each data point received. For this reason, we only present the STRSAGA results for the logistic regression problem.

Regarding Adam, the results for the synthetic dataset in Figure 9 are analogous to the ones obtained for SGD, as we observe an initial *bias regime*, where all the algorithms converge similarly, that is successively slowed down and eventually leads to the variance regime. We see that COCO leads to improved performance in terms of the variance regime without compromising the bias regime and that the improvement increases with the number K of gradients simultaneously denoised. In the logistic regression problems, for the “fourclass dataset”, Adam almost does not show bias compromise (due to its adaptive nature), but its variance gains only appear for higher values of K; for the “mushrooms dataset”, although Adam benefits with COCO, its variance improvements do not consistently improve with K. We also emphasize that for Adam the number of oracle queries is different from the other two algorithms due to its adaptive nature and, thus, faster convergence towards the variance regime.

The STRSAGA results are extremely similar to SGD: for the “fourclass dataset”, despite the bias delay, consistent variance improvements are also observed with increasing number K of gradients simultaneously denoised; for the “mushrooms dataset”, consistent variance improvements are again observed both for SGD and STRSAGA with increasing K but without significant bias delay.

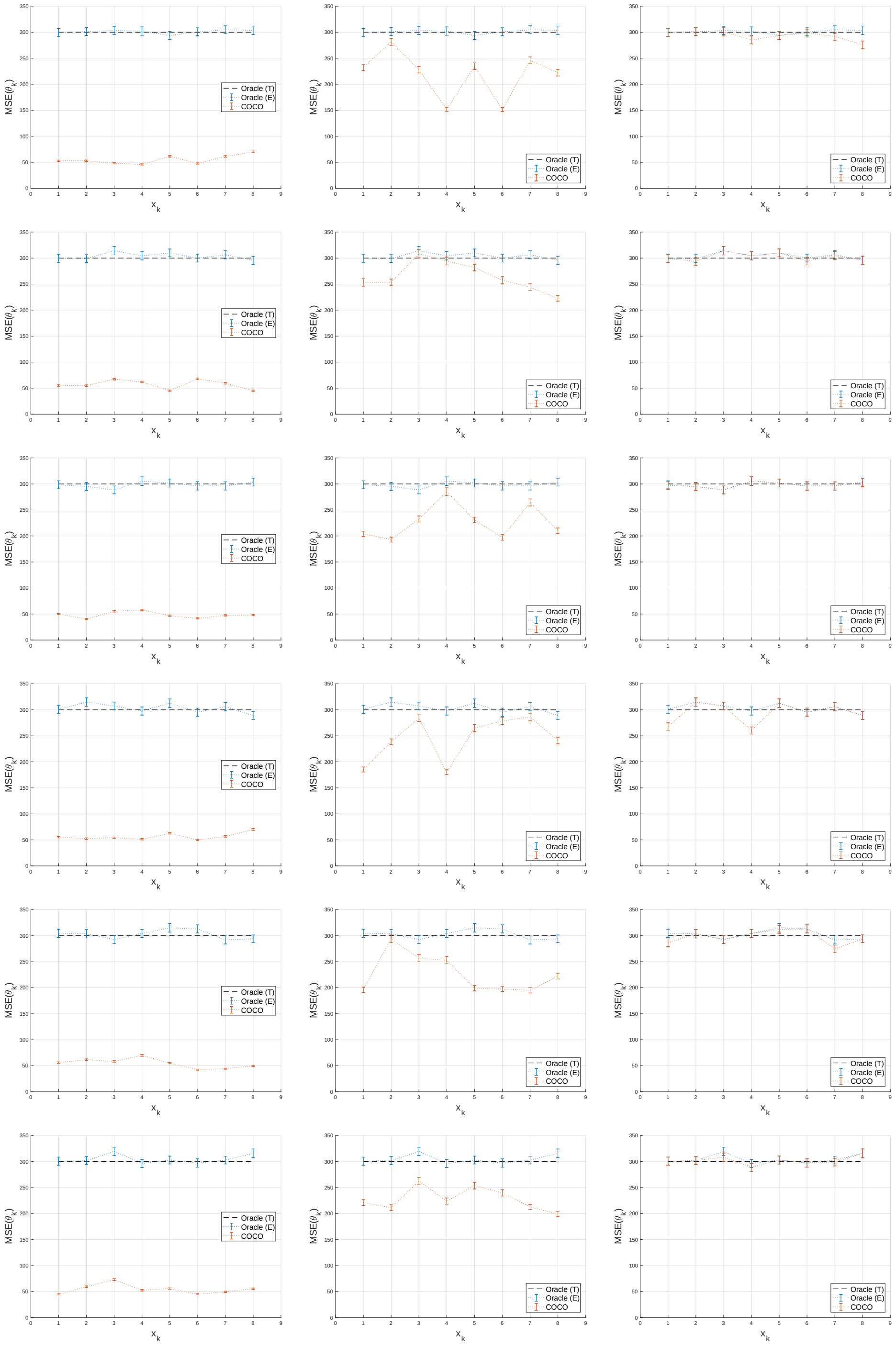

Figure 7: More instances of the same setup of Figure 2, with *Left*: $l = 10$; *Center*: $l = 100$; *Right*: $l = 1000$..

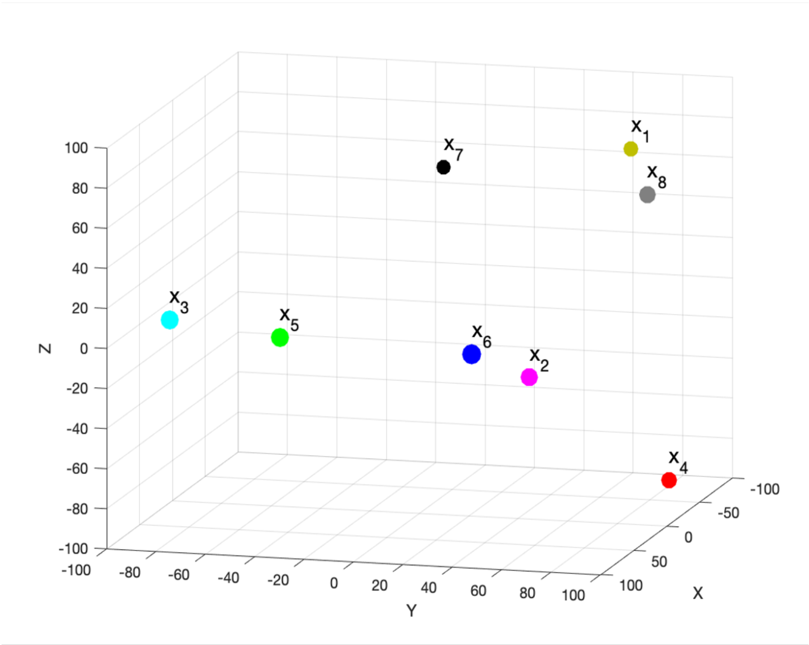

Figure 8: Spatial configuration that yields the results in the plot on the center of 2 (to provide some depth insight, marker size is proportional to the point x-coordinate). From 2, it is possible to observe that x_1, x_2, x_6 and x_8 are the points with the best $\mathrm{MSE}(\hat{\theta}_k)$, followed by x_3 and x_5. Finally, the worst $\mathrm{MSE}(\hat{\theta}_k)$ is obtained for x_4 and x_7. Here we see that this denoising performance can be assigned to the closeness to other points.

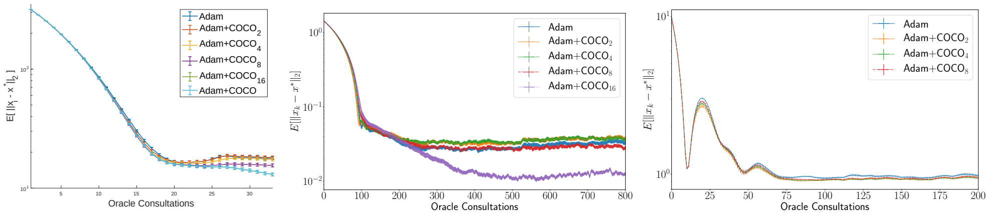

Figure 9: COCO denoiser plugged in Adam. *Left:* synthetic problem satisfying the noise model. $E[\|x_i - x^*\|]$ is averaged over 100 runs. The lines for "Adam + COCO$_{16}$" and "Adam + COCO" are superimposed. *Center:* 100 runs of a logistic regression problem built on the *fourclass* dataset [4]. *Right:* 50 runs of a logistic regression problem built on the *mushrooms dataset* [4]. The width of each marker represents the standard error of the mean.

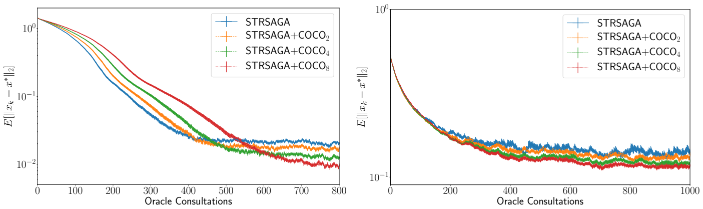

Figure 10: COCO denoiser plugged in STRSAGA. *Left:* 100 runs of a logistic regression problem built on the *fourclass* dataset [4]. *Right:* 50 runs of a logistic regression problem built on the *mushrooms dataset* [4]. The width of each marker represents the standard error of the mean.