Conservative Online Convex Optimization

Martino Bernasconi-de-Luca, Edoardo Vittori,
Francesco Trovò (✉), and Marcello Restelli

Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano,
Piazza Leonardo da Vinci, 32, Milano, Italy
{martino.bernasconideluca, edoardo.vittori, francesco1.trovo,
marcello.restelli}@polimi.it

Abstract. Online learning algorithms often have the issue of exhibiting poor performance during the initial stages of the optimization procedure, which in practical applications might dissuade potential users from deploying such solutions. In this paper, we study a novel setting, namely *conservative online convex optimization*, in which we are optimizing a sequence of convex loss functions under the constraint that we have to perform at least as well as a known default strategy throughout the entire learning process, a.k.a. conservativeness constraint. To address this problem we design a meta-algorithm, namely *Conservative Projection* (CP), that converts any no-regret algorithm for online convex optimization into one that, at the same time, satisfies the conservativeness constraint and maintains the same regret order. Finally, we run an extensive experimental campaign, comparing and analyzing the performance of our meta-algorithm with that of state-of-the-art algorithms.

Keywords: online learning ·

1 Introduction

In the classic Empirical Risk Minimization (ERM) framework [38], the objective is to solve a stochastic optimization problem by minimizing the empirical loss function over a given set of training examples drawn from the unknown distribution. However, using the ERM approach in production exposes the learner to the issue of concept drift [37], *i.e.,* the risk that the distribution producing a training dataset may differ from the one observed during the operational life of the model. A solution to this issue is offered by techniques deriving from the Online Convex Optimization (OCO) field [32], which aim at minimizing a sequence of convex loss functions w.r.t. to the best-fixed strategy in hindsight. Nonetheless, even if they ensure convergence to the optimal solution, OCO techniques notoriously have poor empirical performance during the early stages of the learning process [29], which might dissuade potential users from deploying such solutions. To model this issue, we define a novel *Conservative Online Convex Optimization* (COCO) framework in which the learner has to perform online asymptotically as well as the best-fixed decision in hindsight while satisfying a *conservativeness* constraint, *i.e.,* during the operational life of the system it has to perform no worse than a

given fixed strategy. Furthermore, we propose the Conservative Projection (CP) algorithm, a newly-designed online learning meta-algorithm, applicable to any OCO algorithms, that exploits the strengths of both ERM and OCO solutions.

Learning an optimal strategy while satisfying a conservativeness constraint during the exploration phase is of paramount importance in multiple domains. For instance, an intuitive example can be found in automatic spam filters [6]. Generally, companies optimize offline models on historical data, *e.g.*, past e-mails, until such classifiers perform satisfactorily given the collected dataset. When deploying this product, the company would like to maintain at least the above-mentioned performance while continually optimizing the model, integrating newly collected data, and possibly adapting to data distribution changes. Another field that benefits from being conservative is the financial field, *e.g.,* the asset management sector [7]. In this context, the goal of portfolio managers is to beat a specific market index (a weighted average of a set of stocks), *i.e.,* to perform better than the chosen index and, concurrently, maximize the collected wealth.

The idea of learning while guaranteeing the performance of a fixed and known policy has also been studied in the fields of Reinforcement Learning (RL) [13] and Multi-Armed Bandits (MAB) [40]. To the best of our knowledge, no work explicitly tackles the problem of conservativeness in the OCO framework as defined in this paper. To solve this problem, we extend the techniques from the OCO literature, and propose a meta-algorithm, namely CP, which extends *any* online learning algorithm to satisfy the requirements of the COCO framework. Thanks to the use of a pseudo-loss and a projection in a so-called *conservative ball*, the proposed CP algorithm provides anytime guarantees w.r.t. a fixed default strategy. Specifically, the contributions of this work are:

- the definition of the novel COCO framework, where the objective is to obtain sub-linear regret while performing better than the default strategy during the entire learning process;
- the CP algorithm, which provides a solution to the above-mentioned problem for *any* OCO algorithm. Furthermore, we provide theoretical evidence that CP performs at least as well as the default strategy and that its regret is of the same order as that of the original OCO algorithm considered;
- an in-depth empirical evaluation of the CP algorithm in terms of regret and conservativeness on both simulated and real-world problems, comparing it with state-of-the-art algorithms from the OCO literature.

2 Background

Problems closely related to those of conservativeness have been commonly addressed by *safe* RL techniques. In [15], the authors provide a comprehensive overview of the different definitions of *safety* in RL. The most common assumption is to have access to a safe policy, and the goal is to improve that policy monotonically throughout the learning process. The seminal paper for this setting in [18], which proposes a conservative policy iteration algorithm with monotonic

improvement guarantees for mixtures of greedy policies. This approach is generalized to stationary and stochastic policies in [28,31]. Building on the former, in [25,27,26] the authors have designed monotonically improving policy gradient algorithms for Gaussian, Lipschitz, and, recently, smoothing policies. This setting differs substantially from ours as the underlying environment is assumed to be stochastic.

In the bandit setting, the authors of [23] analyzed the same problem, characterizing the Pareto regret frontier in the stochastic case, *i.e.,* a surface determined by the admissible regret bounds for each arm. Following these seminal works, the interest of the MAB community in conservative exploration has grown in recent years, starting with the work presented in [40], where the authors modified the well-known UCB algorithm [2,3] to guarantee the safety constraint. Later, the idea was applied to contextual linear bandits in [19] and later improved in [14], as wells as to GPUCB, as presented in [35,34]. We inherit the concept of *safety as conservatism* from these works on stochastic bandit feedback and apply it to the context of adversarial full-information feedback.

In the Expert Learning literature, a work similar to ours is [30]. In this work, the authors design a strategy, named (A, B)-prod, that provides regret guarantees w.r.t. the regret of two generic strategies A, and B. However, their conservativeness definition is not comparable to ours, since it does not hold anytime. The question of bounding the regret not only to the best action but also to other strategies is addressed in [17,21], in which the authors proved, for the full information setting, that there exists an algorithm that guarantees a regret of $\mathcal{O}(\sqrt{T})$, with a specific constant for each expert. In particular, the main focus of the paper is to characterise the admissible vectors $\{r_k\}_{k\in K}$ guaranteeing a regret $R_T^k \leq r_k$ w.r.t. each expert k. Even if these works cover a more general theoretical framework than ours, *i.e.,* multi-objective regret minimization, the algorithms therein do not guarantee that their loss is strictly smaller than that of a given expert, and, therefore, their results cannot be compared with ours.

3 Problem Formulation

Let us build on the standard Online Convex Optimization framework [32] in which a learning agent, at each round t, has to select a parameter $\theta_t \in \Theta$, representing a strategy, where $\Theta \subset \mathbb{R}^d$ is a closed and convex set of a finite d dimensional Euclidean space. At each round t, the agent receives a loss $f_t(\theta_t)$ where $f_t : \Theta \to [\epsilon_l, \epsilon_u]$ is a convex and differentiable function, where ϵ_l, ϵ_u are the minimum and maximum value of the function $f_t(\cdot)$, respectively, and $0 \leq \epsilon_l < \epsilon_u$. The objective of the learning agent $\mathfrak{U}$ is to minimize the regret $R_T(\mathfrak{U})$ over a given time horizon $T \in \mathbb{N}$, *i.e.,* the difference between the loss suffered by the algorithm $\mathfrak{U}$ and the one suffered from the best fixed decision in hindsight, formally defined as:

$$R_T(\mathfrak{U}) := L_T - \bar{L}_T,$$

where $L_T := \sum_{t=1}^{T} f_t(\theta_t)$ and $\bar{L}_T := \sum_{t=1}^{T} f_t(\bar{\theta})$ are the loss accumulated by the running algorithm $\mathfrak{U}$ and the smallest loss obtainable by a clairvoyant selection of the parameters, *i.e.,* $\bar{\theta} := \arg\inf_{\theta \in \Theta} \sum_{t=1}^{T} f_t(\theta)$, respectively.

In the COCO setting, we are interested in those algorithms $\mathfrak{U}$ for which the regret $R_T(\mathfrak{U})$ is bounded by a sub-linear function of the time horizon T, and, at the same time, perform throughout the optimization at least as well as an established default parameter $\tilde{\theta} \in \Theta$, selected at the beginning of the learning process. While the former requirement represents the so-called no-regret property of an algorithm [9], the latter one is formally defined as follows:

Definition 1. *An online algorithm $\mathfrak{U}$ is said to be* conservative *if it satisfies the following conservativeness constraint for each $t \in [T]$:*

$$L_t \leq (1 + \alpha)\tilde{L}_t, \tag{1}$$

where $\alpha > 0$ is the conservativeness level required by the problem, and $\tilde{L}_t := \sum_{k=1}^{t} f_k(\tilde{\theta})$ is the cumulative loss of the default parameter $\tilde{\theta}$ over t rounds.[1,2]

From now on, we will refer to the quantity $Z_t(\mathfrak{U}) := (1 + \alpha)\tilde{L}_t - L_t$ as the *budget* of the algorithm $\mathfrak{U}$, *i.e.,* the advantage in terms of loss accumulated by $\mathfrak{U}$ over time w.r.t. the one provided by a constant choice of the default parameter $\tilde{\theta}$. We also assume that there exists $\mu > \epsilon_l$ s.t. $\tilde{L}_t \geq \mu t$, which imply that the fixed strategy $\tilde{\theta}$ is sub-optimal.

We remark that, in this work, we require the constraint in Equation (1) to be satisfied at each round $t \in [T]$. Indeed, any online learning algorithm $\mathfrak{U}$ providing a regret of $R_t(\mathfrak{U}) \leq \xi\sqrt{t}$ is guaranteed to satisfy the above constraint for $t > \left(\frac{\xi}{\alpha\mu}\right)^2$, instead we require that it holds for each $t \in [T]$.[3] Therefore, satisfying the condition imposed by our constraint requires the design of ad-hoc algorithms. Conversely, the design of algorithms which have a higher grade of conservativeness, *i.e.,* $\alpha \leq 0$ is not a viable option due to the following:

Theorem 1. *In the OCO setting, there is no algorithm $\mathfrak{U}$ which obtains $L_t \leq \tilde{L}_t$, unless $\theta_t = \tilde{\theta}$ for all $t \in [T]$.*

Proof. Let k be the first round in which the algorithm $\mathfrak{U}$ plays $\theta_k \neq \tilde{\theta}$. If the loss function is $f_k(x) := f_k(\tilde{\theta}) + ||\tilde{\theta} - x||_2$, then, by the convexity of the space Θ, we can find $c \in (0, 1)$ and $z \in \Theta$ s.t. $\theta_k = c\tilde{\theta} + (1 - c)z$. This implies that $f_k(\theta_k) = f_k(\tilde{\theta}) + (1 - c)||\tilde{\theta} - z||_2 > f_k(\tilde{\theta})$, showing that $L_t > \tilde{L}_t$.

[1] The conservativeness constraint in Equation (1) is expressed in terms of losses, as commonly done in the OCO framework. Its reformulation in terms of rewards, as commonly done in the RL and MAB fields, is straightforward.

[2] We denote with $[T]$ the set $\{1, \ldots, T\}$.

[3] This comes from the fact that $L_t - \tilde{L}_t \leq R_t(\mathfrak{U})$ and $\xi\sqrt{t} \leq \alpha\mu t$ holds for $t > \left(\frac{\xi}{\alpha\mu}\right)^2$.

In other words, it is impossible to guarantee that an algorithm does strictly better than or equal to a given default parameter $\tilde{\theta}$ over the entire time horizon T, unless one always plays the default parameter.

4 The Conservative Projection Algorithm

We begin this section by characterizing a set of parameters in the parameter space Θ which guarantees that their choice implies the conservativeness of an algorithm at round t. Then, we select a specific parameter from this set, thus defining the CP algorithm, and, subsequently, we show it is conservative and it has sub-linear bounds for the regret.

4.1 The Conservative Ball

Let us define the following:

Definition 2. *A* conservative ball $B(\tilde{\theta}, r_t) \in \mathbb{R}^d$ *is a d-dimensional ball centered in* $\tilde{\theta}$ *with radius:*

$$r_t := \left[1 - \left(\frac{L_{t-1} - (1+\alpha)\tilde{L}_{t-1} - \alpha\epsilon_l}{DG} + 1\right)^+\right] D, \tag{2}$$

where $D := \sup_{x,y\in\Theta} ||x-y||_2$ *is a bound on the diameter of the parameter space* Θ*,* $G := \sup_{x\in\Theta} ||\nabla f_t(x)||_2$ *is the upper bound on the norm of the gradient of the loss* $f_t(\cdot)$*,* $||\cdot||_2$ *denotes the L2 norm of a vector, and* $(a)^+$ *denotes the maximum between the quantity a and zero.*

From now on, we refer to this ball as the *conservative ball* since this choice of r_t implies that playing any of the parameters $\theta \in B(\tilde{\theta}, r_t)$ at round t guarantees that the accrued budget $Z_t(\mathfrak{U})$ does not become negative. Formally:

Theorem 2. *Let* $B(\tilde{\theta}, r_t)$ *be the conservative ball defined in Equation* (2) *and assume that Equation* (1) *is satisfied at round* $t-1$*. Then, each parameter* $\theta \in B(\tilde{\theta}, r_t) \cap \Theta$ *satisfies Equation* (1) *at round* t*.*

Proof. Given $\theta \in B(\tilde{\theta}, r_t) \cap \Theta$ we have:

$$f_t(\theta) - (1+\alpha) f_t(\tilde{\theta}) \leq \langle \nabla f_t(\theta), \theta - \tilde{\theta} \rangle - \alpha f_t(\tilde{\theta}) \leq G r_t - \alpha\epsilon_l, \tag{3}$$

where the first inequality is given by the convexity of $f_t(\cdot)$, and the second inequality is given by the Cauchy-Schwarz inequality and by the fact that $\theta \in B(\tilde{\theta}, r_t)$ implies $||\tilde{\theta} - \theta||_2 \leq r_t$. Let us consider two cases: $r_t < D$, and $r_t = D$.

Case $r_t < D$**:** In this case, the value of the radius is $r_t = \frac{(1+\alpha)\tilde{L}_{t-1} - L_{t-1} + \alpha\epsilon_l}{G}$. By substituting it in Equation (3), we conclude that:

$$f_t(\theta) - (1+\alpha) f_t(\tilde{\theta}_t) \leq (1+\alpha)\tilde{L}_{t-1} + L_{t-1}. \tag{4}$$

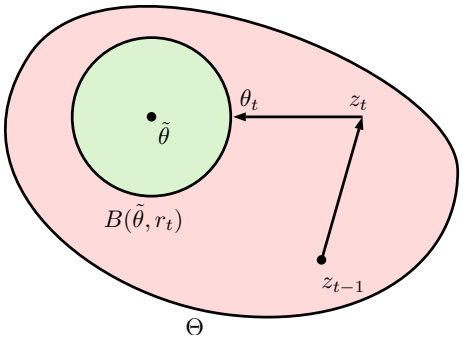

Fig. 1: Visual representation of CP. In green the conservative ball $B(\tilde{\theta}, r_t)$, and in red the parameter set Θ. The CP algorithm selects the parameter θ_t for round t by projecting the parameter z_t, selected by $\mathcal{A}$, on the conservative ball.

Case $r_t = D$: From the fact that $r_t \geq 0$ and using Equation (2), we obtain that:

$$\frac{L_{t-1} - (1+\alpha)\tilde{L}_{t-1} - \alpha\epsilon_l}{GD} + 1 \leq 0 \tag{5}$$

$$GD - \alpha\epsilon_l \leq (1+\alpha)\tilde{L}_{t-1} + L_{t-1}. \tag{6}$$

Combining the above result with the inequality in Equation (3), provides the same result presented in Equation (4).

The proof is concluded by rearranging the terms of Equation (4).

Notice that the projection of a generic parameter z_t on the ball $B(\tilde{\theta}, r_t)$ can be computed analytically and efficiently. Indeed, the projection operation on the conservative ball satisfies the following:

$$\theta_t = \Pi_{B(\tilde{\theta}, r_t)}(z_t) = \beta_t \tilde{\theta} + (1 - \beta_t) z_t, \tag{7}$$

where

$$\beta_t = \begin{cases} 1 - \frac{r_t}{||z_t - \tilde{\theta}||_2} & z_t \notin B(\tilde{\theta}, r_t) \\ 0 & z_t \in B(\tilde{\theta}, r_t) \end{cases}. \tag{8}$$

In what follows, we choose z_t as the parameter provided by a generic OCO algorithm at round t.

4.2 Description of the CP Algorithm

Theorem 2 provides a way to choose a sequence of parameters over time, for which the conservativeness constraint is satisfied. The CP algorithm uses this result by choosing, at each round t, the parameter θ_t in the ball $B(\tilde{\theta}, r_t)$ as close as possible to the prediction z_t provided by the OCO algorithm fed using the pseudo-loss function $g_{t-1}(z_{t-1}) := (1 - \beta_{t-1}) f_{t-1}(z_{t-1})$, *i.e.,* it selects a convex

Algorithm 1 CP

Require: Online learning algorithm $\mathcal{A}$, conservativeness level $\alpha > 0$, default parameter $\tilde{\theta} \in \Theta$

1: Set $\tilde{L}_0 \leftarrow 0$, $L_0 \leftarrow 0$, and $\beta_0 \leftarrow 1$
2: **for** $t \in [T]$ **do**
3: Get point z_t from $\mathcal{A}$ applied to loss $g_{t-1}(z_{t-1})$
4: Compute r_t as in Equation (2)
5: Select $\theta_t = \Pi_{B(\tilde{\theta}, r_t)}(z_t)$
6: Suffer loss $f_t(\theta_t)$
7: Observe $f_t(z_t)$ and $f_t(\tilde{\theta})$
8: Set $g_t(z_t) \leftarrow (1 - \beta_t) f_t(z_t)$
9: **end for**

combination of the default parameter $\tilde{\theta}$ and z_t. The intuition behind this choice is that we want to choose θ_t as close as possible to the no-regret prediction z_t of the OCO algorithm, that is guaranteed to have sub-linear regret. Furthermore, we show that this algorithm increases the radius r_t over time, and therefore, in finite-time, the conservative ball includes the parameter z_t, allowing CP to have a sub-linear regret. Finally, we remark that the CP algorithm is designed so that the more the default parameter $\tilde{\theta}$ is distant from the optimal one, the more the value of the radius r_t increases, which, in its turn, decreases the cost of guaranteeing conservativeness.

The pseudo-code of the CP algorithm is presented in Algorithm 1, and its visual representation is depicted in Figure 1. The algorithm requires as input a generic online learning algorithm $\mathcal{A}$, which selects the parameter z_t to play at each round t, a conservativeness level $\alpha > 0$, and the default parameter $\tilde{\theta} \in \Theta$. At first, we set the initial value of the cumulative losses $L_0 = 0$, that of the default parameter $\tilde{L}_0 = 0$ (Line 1), and we set the parameter $\beta_0 = 1$. Afterwards, at each round t, z_t is chosen by the algorithm $\mathcal{A}$ by considering the pseudo-loss $g_t(x)$ (Line 3). Thanks to a projection operation (Line 5), which projects z_t into the conservative ball $B(\tilde{\theta}, r_t)$, the resulting parameter θ_t satisfies the conservativeness constraint in Equation (1). Finally, the algorithm suffers the loss $f_t(\theta_t)$, and observes $f_t(z_t)$ and $f_t(\tilde{\theta})$, *i.e.,* the loss of the algorithm $\mathcal{A}$ and the default parameter $\tilde{\theta}$, respectively (Lines 7-8).

Notice that, from a computational point of view, the CP algorithm has a small computational overhead w.r.t. the original online learning algorithm $\mathcal{A}$, *i.e.,* an overhead proportional to d, due to the additional projection on the conservative ball and the evaluation of the losses $f_t(\theta_t)$, and $f_t(\tilde{\theta})$.

4.3 Analysis of the CP Algorithm

In this section, we prove that CP has the desired conservativeness property and maintains the sub-linear regret of the subroutine algorithm $\mathcal{A}$. Since the CP algorithm selects a parameter θ_t inside the conservative ball $B(\tilde{\theta}, r_t)$, a

straightforward corollary of Theorem 2 guarantees that the conservativeness constraint is satisfied. Formally:

Corollary 3. *The CP algorithm applied to a generic online learning algorithm $\mathcal{A}$ is conservative.*

Once we established the conservativeness of our approach, we need to prove that the CP algorithm has sub-linear regret. Intuitively, we need to show that the radius r_t grows over time, and eventually includes the entire space Θ, so that from a specific round we are allowed to follow the no-regret choice z_t. Formally, we show the following:

Theorem 3. *Consider any OCO algorithm $\mathcal{A}$ which guarantees a regret of $R_T(\mathcal{A}) \leq \xi\sqrt{T}$. The CP algorithm using $\mathcal{A}$ as subroutine has the following regret bound:*

$$R_T(CP) \leq \xi\sqrt{T} + \tau DG, \tag{9}$$

for any $T > \tau$, where:

$$\tau := \frac{2\alpha\mu(DG + \alpha\mu) + \xi\left(\sqrt{\xi^2 + 4\alpha\mu(DG + \alpha\mu)} + \xi\right)}{2\alpha^2\mu^2}. \tag{10}$$

Proof. Using the convexity of the loss functions on the regret and the definition of θ_t in Equation (7), we have:

$$\begin{aligned} L_T - \tilde{L}_T &\leq \sum_{t=1}^{T}[\beta_t f_t(\tilde{\theta}) + (1-\beta_t)f_t(z_t) - f_t(\tilde{\theta})] \\ &= \sum_{t=1}^{T}(1-\beta_t)[f_t(z_t) - f_t(\tilde{\theta})] \qquad (11) \\ &\leq \sup_{\theta\in\Theta}\left(\sum_{t=1}^{T}(1-\beta_t)[f_t(z_t) - f_t(\theta)]\right) \leq \xi\sqrt{T}. \qquad (12) \end{aligned}$$

This shows that the CP algorithm has sub-linear regret w.r.t. an algorithm that always chooses the default parameter $\tilde{\theta}$ over the entire time horizon T.

Combining Equation (8) and (2), we have:

$$\begin{aligned} \beta_t &\leq 1 - \frac{r_t}{||z_t - \tilde{\theta}||_2} \leq 1 + \frac{L_{t-1} - (1+\alpha)\tilde{L}_{t-1} - \alpha\epsilon_l}{DG} \qquad (13) \\ &\leq 1 + \frac{\xi\sqrt{t} - (t-1)\mu\alpha}{DG}, \qquad (14) \end{aligned}$$

where we used the bound in Equation (12), the fact that the space Θ has radius D, and that $\tilde{\theta}$ is not a no-regret strategy, and, hence, there exists a $\mu > \epsilon_l > 0$ s.t. $\tilde{L}_{t-1} > \mu(t-1)$.

On the other hand, we assumed that $\mathcal{A}$ is a no-regret strategy and, therefore, the regret of the algorithm $\mathcal{A}$ is sub-linear, this means that there exists a round

$\tau > 0$ s.t. Equation (14) is negative, and, consequently, for $t > \tau$, defined in Equation (10) we have $\beta_t = 0$. The value of τ is provided by the solution of the following equation $1 + \frac{\xi\sqrt{\tau} - \tau\mu\alpha}{DG} = 0$.

What we showed above also proves that the CP algorithm for $t > \tau$ eventually plays the same parameter as $\mathcal{A}$ since for all $t > \tau$ the pseudo-losses $g_t(\cdot)$ and the true losses $f_t(\cdot)$ coincide. Indeed, the regret of the CP algorithm can be written as:

$$R_T(CP) \leq \sum_{t=1}^{\tau} \left[\beta_t f_t(\tilde{\theta}) + (1-\beta_t) f_t(z_t) - f_t(\bar{\theta})\right] + \sum_{t=\tau+1}^{T} \left(f_t(z_t) - f_t(\bar{\theta})\right) \quad (15)$$

$$\leq \sum_{t=1}^{\tau} \beta_t \left[f_t(\tilde{\theta}) - f_t(z_t)\right] + \sum_{t=1}^{T} [f_t(z_t) - f_t(\bar{\theta})] \quad (16)$$

$$\leq \sum_{t=1}^{\tau} \beta_t \langle \nabla f_t(\tilde{\theta}), \tilde{\theta} - z_t \rangle + \sum_{t=1}^{T} [f_t(z_t) - f_t(\bar{\theta})] \quad (17)$$

$$\leq \tau DG + \xi\sqrt{T}, \quad (18)$$

where the inequality in Equation (15) uses the convexity of $f_t(\cdot)$. Equation (16) comes from the extension of the time horizon from $\{\tau, \ldots, T\}$ to $\{1, \ldots, T\}$. Equation (17) follows from the convexity of $f_t(\cdot)$ and the inequality in Equation (18) follows from the Cauchy-Schwarz inequality on the first term while the second term is the regret of the used no-regret algorithm $\mathcal{A}$.

A regret of order $\mathcal{O}(\sqrt{T})$ is tight in general OCO problems [1], but there exists specific settings in which a $\mathcal{O}(\log T)$ regret can be achieved, *e.g.*, in the case of H-strongly convex losses or in the case of exp-concave losses [16]. In such settings, the CP algorithm guarantees $\mathcal{O}(\log T)$ regret together with the conservative constraint, formally:

Theorem 4. *Consider any OCO algorithm $\mathcal{A}$ which guarantees a regret of $R_T(\mathcal{A}) \leq \rho \log(T)$. The CP algorithm using $\mathcal{A}$ as subroutine has the following regret bound:*

$$R_T(CP) \leq \rho \log(T) + \tau DG, \quad (19)$$

for any $T > \tau$, where:

$$\tau := \frac{\alpha e^2 \mu(DG + \alpha\mu) + 2\rho\left(\sqrt{\alpha e^2 \mu(DG + \alpha\mu) + \rho^2} + \rho\right)}{e^2\alpha^2\mu^2}. \quad (20)$$

Proof. The proof is similar to that of Theorem 4, we only report the steps that are significantly different from it. From Equation (12), which holds also in this setting, we obtain:

$$L_T - \tilde{L}_T \leq \rho \log(T). \quad (21)$$

This shows that the regret w.r.t. an algorithm which always chooses the default parameter $\tilde{\theta}$ is of the order $\mathcal{O}(\log(T))$. Following the same steps used to derive

Equation (14), we have that $\beta_t \leq 1 + \frac{\rho \log(t) - t\mu\alpha}{DG}$. Therefore, β_t is zero after τ rounds, where τ is defined in Equation (20).[4] Finally, using the same argument used to derive Equation (18), we obtain the bound present in the theorem.

Notice that for Theorem 3 and 4 we have that $\tau \propto 1/\mu$, meaning that for default parameters $\tilde{\theta}$ with smaller accrued losses w.r.t. the optimum $\bar{\theta}$ (and hence smaller μ), the CP algorithm is required to wait longer to play the action prescribed by the no-regret strategy $\mathcal{A}$. Moreover, the bound shows a dependence $\tau \propto 1/\alpha$, meaning that a tighter conservative constraint makes the problem more challenging for the CP algorithm.[5]

5 Experiments

This section provides the experimental study of the proposed algorithm for the COCO setting, where we use OGD [41] as subroutine. We evaluate the performance of the CP-OGD in three settings: a synthetically generated regression problem, and two real-world classification scenarios. We compare our performances to OGD [41], the non-conservative version of the proposed algorithm, AdaGrad [12], a state-of-the-art algorithm of online optimization which has theoretical guarantees on the regret, the Conservative Switching (CS) algorithm, a naive conservative baseline, and the Constrained Reward Doubling Guess (CRDG). CS is a budget-first algorithm we designed. This algorithm plays the fixed default action until enough budget has been accrued, then it plays the no regret strategy. We described it and provide its theoretical properties in Appendix B.1. As for CP, in CS we consider OGD as subroutine and, thus, will refer to it as CS-OGD. CRDG is a conservative baseline obtained by combining the Reward Doubling Guess algorithm [33], originally designed for unconstrained online optimization setting, with the *Constraint Set Reduction* procedure presented in [10]. We provide the its detailed pseudo-code and a discussion on its theoretical properties in Appendix B.2.

For CP-OGD, CS-OGD, and OGD we initialize the learning rate $\eta_t = \frac{K}{\sqrt{t}}$, where $K = \frac{D}{G\sqrt{2}}$ is chosen to minimize the theoretical regret bound of OGD, while for AdaGrad we initialize the parameter $\alpha_t = \frac{1}{\sqrt{t}}$, as prescribed in [20]. For the CRDG algorithm we set $\epsilon = \mu\alpha/2$ to guarantee the conservativeness constraint in Definition 1 with level α (as CP-ODG and CS-ODG do). We evaluate the algorithms in terms of regret $R_t(\mathfrak{U})$, and budget $Z_t(\mathfrak{U})$. The code to run the experiments is available at: https://github.com/martinobdl/safe_OCO.

5.1 Synthetic Regression Dataset

We analyze a synthetic online linear regression environment, where the agent is presented with a vector $x_t \sim U([0,1]^d) \subset \mathbb{R}^d$, *i.e.,* a d dimensional vector drawn

[4] The derivation of τ is provided in Lemma 4, reported in Appendix A for space reasons.

[5] In Appendix D.3 we performed experiments to explore the relationship between the conservativeness and performance of the CP algorithm.

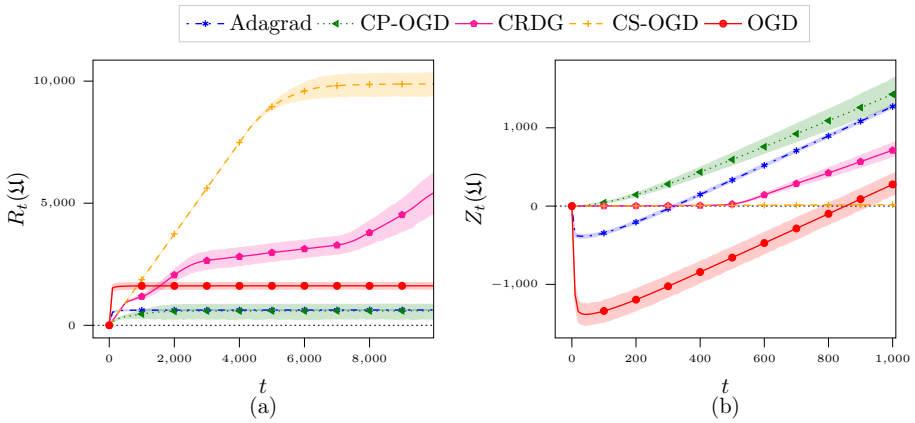

Fig. 2: Results on the synthetically generated regression dataset: (a) regret $R_T(\mathfrak{U})$; (b) magnification of the budget $Z_t(\mathfrak{U})$ over the first 10^3 samples. $\tilde{\theta}$ has been chosen so that $\tilde{D} = 0.5$.

uniformly from $[0, 1]^d$, and the target value is generated as $y_t = \langle x_t, \bar{\theta} \rangle + \gamma_t$, where $\bar{\theta} \in \Theta = [-1, 1]^d$ is the unknown optimal parameter, and γ_t is a noise term that we considered *i.i.d.* with zero mean. Each algorithm provides a prediction $\hat{y}_t = \langle x_t, \theta_t \rangle$, where θ_t is the chosen parameter for round t, and suffers a loss $f_t(\theta_t) := (\langle x_t, \theta_t - \bar{\theta} \rangle - \gamma_t)^2$.

We set $d = 40$, γ_t from a truncated Gaussian distribution $\mathcal{N}(0, 0.15^2)$ with values in $[-1, 1]$, $T = 10^4$, and we fix $\bar{\theta} := [0, \dots, 0]$. The conservativeness level is set to $\alpha = 0.01$. In this setting, the bound on the gradient is $G = 2(\sqrt{2d} + 1)\sqrt{d}$, the minimum and maximum loss are $\epsilon_l = 0$ and $\epsilon_u = \sqrt{2d} + 1$, respectively, and the bound on the diameter of the decision space is $D = \sqrt{2d}$. We ran the experiment 30 times and averaged the results. The confidence intervals on the mean, represented in the figures as semi-transparent areas, are the 95% confidence intervals computed by statistical bootstrap. Multiple default parameters $\tilde{\theta}$ have been considered in this setting so that $\tilde{D} := ||\tilde{\theta} - \bar{\theta}||_2 \in \{0.5, 1, \dots, 3, 3.5\}$.

Results Figure 2 shows the results for experiments where the default parameter has a distance from the optimum of $\tilde{D} = 0.5$. Figure 2a shows that all the algorithms, but CRDG, on average converge to the optimal solution since the regret $R_T(\mathfrak{U})$ is asymptotically approaching a constant value. In particular, AdaGrad and CP-OGD perform comparably in terms of regret, OGD has slightly worse performance, and CRDG and CS-OGD provide a regret more than 3 times larger than the other algorithms over the analyzed time horizon T. The magnification of the budget $Z_t(\mathfrak{U})$ over the first $1,000$ rounds, provided in Figure 2b, shows that OGD and AdaGrad have a negative budget during the first ≈ 900 and ≈ 300 rounds, respectively, while CP-OGD, CS-OGD, and CRDG guarantee the conservativeness constraint at each round, *i.e.,* they have $Z_t(\mathfrak{U}) > 0$, for all $t \in [T]$. These results suggest that the proposed CP-OGD is the only

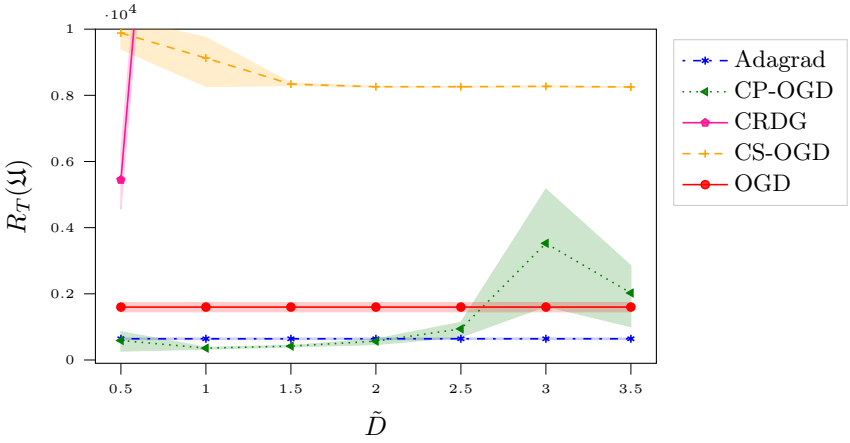

Fig. 3: Regret $R_T(\mathfrak{U})$ at the end of the time horizon T as the distance from the optimum $\tilde{D}$ varies for the synthetic dataset.

algorithm, among the tested ones, capable of maintaining a small regret while, at the same time, being conservative.

Figure 3 presents the behaviour of the regret $R_T(\mathfrak{U})$ as the distance $\tilde{D}$ between the optimum and the default parameter varies. For values of the distance $\tilde{D} < 2.5$, *i.e.,* default parameters which are close to the optimum one, CP-OGD provides a smaller regret than that of all the other algorithms on average. Instead, if $\tilde{D} \geq 3$, the fact that it is constrained to maintain a positive budget penalizes CP-OGD in terms of regret. In such a situation, OGD and AdaGrad provide a smaller regret than CP-OGD. This suggests that the proposed approach might provide a large regret if the default parameter $\tilde{\theta}$ is far from the optimum one $\bar{\theta}$.

5.2 Online Classification: the IMDB Dataset

The second set of experiments has been run on the IMDB dataset [24], consisting of $50,000$ reviews of movies and labels classifying the reviews as positive or negative. Data has been preprocessed as done by [20]. The general setup for the online logistic regression model is as follows: the algorithm processes a single feature vector $x_t \in \{0,1\}^d$ with $d = 10^4$, predicts the probability of belonging to the positive class as $\hat{y}_t \in [0,1]$ as $\hat{y}_t = \sigma(\langle x_t, \theta_t \rangle)$, where $\theta_t \in \Theta = [-2,2]^d$ and suffers a loss given by the binary cross entropy defined as $f_t(\theta_t) = -[y_t \log(\hat{y}_t) + (1-y_t)\log(1-\hat{y}_t)]$, where $y_t \in \{0,1\}$ is the true sample class.[6] In this setting the gradient is bounded by $G = \sqrt{d}$, the diameter by $D = 2\sqrt{d}$ and we set $\alpha = 0.01$. To bound the maximum and minimum loss needed by the CS algorithm, we clipped the loss between $\epsilon_l = 1e^{-4}$ and $\epsilon_u = 10$. The default parameter $\tilde{\theta}$ has been generated by training a batch logistic regression using $1,000$ samples at random from the dataset. Notice that the IMDB dataset

[6] $\sigma(x) := 1/(1 + exp(-x))$ is the sigmoid function.

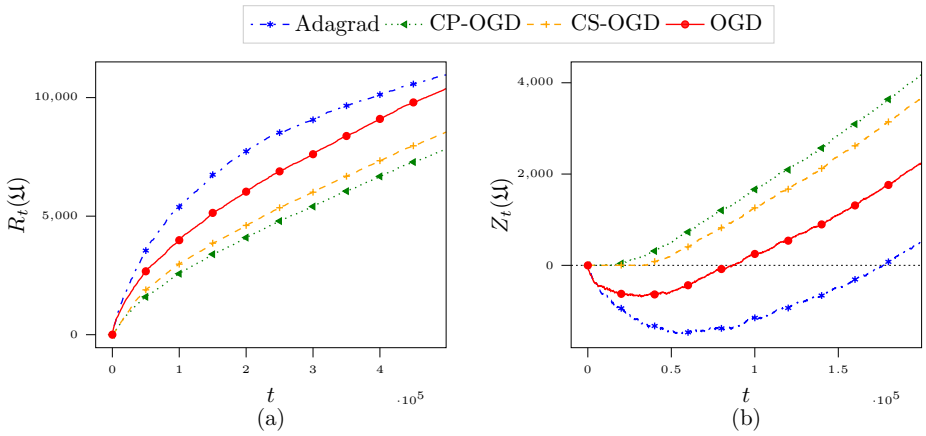

Fig. 4: Results for the IMDB movie dataset: (a) regret, (b) budget for the first 2.5×10^5 samples.

is known to be a challenging setting for OGD [20] due to the sparse nature of its input, setting for which an adaptive step size algorithm, like AdaGrad, generally performs better in terms of regret than the single-pass ones. We could not run the CRDG algorithm on the IMDB dataset since its computational requirements in this setting were too demanding due to the large number of features.

Results Figure 4a shows the regret $R_t(\mathfrak{U})$ for the analyzed algorithms. Both CP-OGD and CS-OGD outperform AdaGrad and OGD in terms of regret. This happens because AdaGrad and OGD surpass the performance of the default parameter only after many rounds. In fact, this specific setting is challenging for OGD [20], while CP-OGD and CS-OGD exploit successfully the information provided by $\tilde{\theta}$. The results suggest that conservative algorithms might also outperform their non-conservative counterparts in some specific challenging optimization problems. Furthermore, Figure 4b shows that, even in this setting, the budget of the OGD and AdaGrad algorithms is negative for the first $\approx 100,000$ and $\approx 200,000$ rounds, respectively, while the budget of the CP-OGD and CS-OGD is positive for all $t \in [T]$, which is in line with the theoretical analysis we provided before.

5.3 Online Classification: the SpamBase Dataset

The SpamBase dataset is taken from the UCI repository and contains $4,601$ emails labeled as spam or ham [11]. The dataset has been normalized so that the input vector $x_t \in [0,1]^d$, with $d = 57$. The safe parameter $\tilde{\theta}$ has been generated by training a batch logistic regression on 100 samples chosen at random from the dataset. The values for the parameters not explicitly defined in this section are the same as those used in the IMDB experiment.

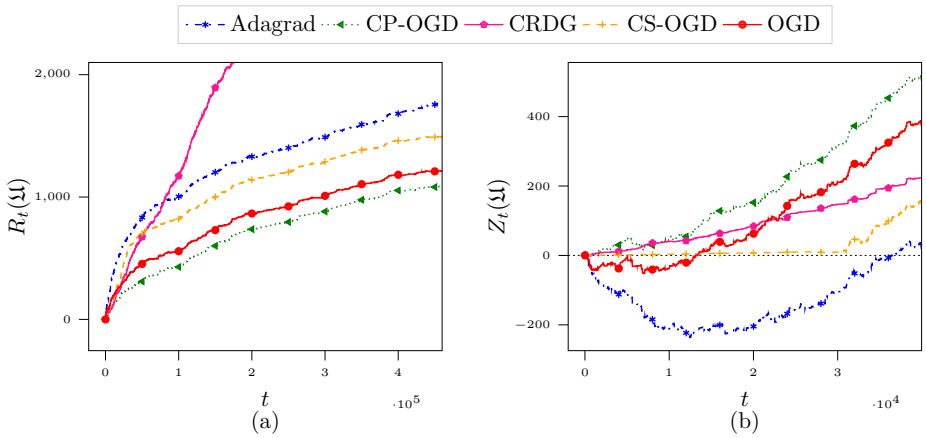

Fig. 5: Results for the SpamBase dataset: (a) regret, (b) budget for the first 4×10^4 samples.

Results Figure 5a shows the regret suffered by the algorithms on the SpamBase dataset. Even in this case, CP-OGD outperforms all the others, and, by looking at Figure 5b, we see that also in this experiment the budget of OGD and AdaGrad is negative for $\approx 10,000$ and $\approx 30,000$ rounds, respectively. Finally, the CRDG algorithm satisfies the budget constrain during the entire learning time horizon but accumulates a large regret over the time horizon.

6 Conclusions

The focus of this paper is to solve the problem of conservative optimization in an online setting with adversarial environments, in which we require an algorithm to provide sub-linear regret while performing at least as well as a given fixed strategy. To solve this problem, we proposed the CP algorithm, showed that it satisfies the conservativeness constraint, and proved that it maintains the same regret order the OCO algorithm it uses as a subroutine. Furthermore, we ran an extensive experimental campaign on synthetic and real-wolrd data, showing that the CP algorithm is competitive in terms of regret with OGD, AdaGrad, CS, and CRDG while also behaving conservatively.

An interesting direction is whether the assumption that the default strategy $\tilde{\theta}$ is fixed can be relaxed to include specific classes of time-varying strategies. Another line of research that might be promising is the use of the definition of the conservative ball to design algorithms also for the unconstrained online optimization setting.

References

1. Abernethy, J., Bartlett, P.L., Rakhlin, A., Tewari, A.: Optimal strategies and minimax lower bounds for online convex games. Tech. rep., University of California, Berkeley, United States of America (2008)
2. Auer, P., Cesa-Bianchi, N., Fischer, P.: Finite-time analysis of the multiarmed bandit problem. Machine learning **47**(2-3), 235–256 (2002)
3. Auer, P., Ortner, R.: Ucb revisited: Improved regret bounds for the stochastic multi-armed bandit problem. Periodica Mathematica Hungarica **61**(1-2), 55–65 (2010)
4. Azuma, K.: Weighted sums of certain dependent random variables. Tohoku Mathematical Journal, Second Series **19**(3), 357–367 (1967)
5. Besson, L., Kaufmann, E.: What doubling tricks can and can't do for multi-armed bandits. arXiv preprint arXiv:1803.06971 (2018)
6. Blanzieri, E., Bryl, A.: A survey of learning-based techniques of email spam filtering. Artificial Intelligence Review **29**(1), 63–92 (2008)
7. Browne, S.: Beating a moving target: Optimal portfolio strategies for outperforming a stochastic benchmark. In: Handbook of the Fundamentals of Financial Decision Making: Part II, pp. 711–730. World Scientific (2013)
8. Cesa-Bianchi, N., Conconi, A., Gentile, C.: On the generalization ability of on-line learning algorithms. IEEE Transactions on Information Theory **50**(9), 2050–2057 (2004)
9. Cesa-Bianchi, N., Lugosi, G.: Prediction, learning, and games. Cambridge university press (2006)
10. Cutkosky, A., Orabona, F.: Black-box reductions for parameter-free online learning in banach spaces. In: Conference On Learning Theory (COLT). pp. 1493–1529. PMLR (2018)
11. Dua, D., Graff, C.: UCI machine learning repository (2017), http://archive.ics.uci.edu/ml
12. Duchi, J., Hazan, E., Singer, Y.: Adaptive subgradient methods for online learning and stochastic optimization. Journal of Machine Learning Research **12**(7) (2011)
13. Garcelon, E., Ghavamzadeh, M., Lazaric, A., Pirotta, M.: Conservative exploration in reinforcement learning. In: International Conference on Artificial Intelligence and Statistics (AISTATS). pp. 1431–1441 (2020)
14. Garcelon, E., Ghavamzadeh, M., Lazaric, A., Pirotta, M.: Improved algorithms for conservative exploration in bandits. In: Conference on Artificial Intelligence (AAAI). pp. 3962–3969 (2020)
15. Garcıa, J., Fernández, F.: A comprehensive survey on safe reinforcement learning. Journal of Machine Learning Research **16**(1), 1437–1480 (2015)
16. Hazan, E., Agarwal, A., Kale, S.: Logarithmic regret algorithms for online convex optimization. Machine Learning **69**(2-3), 169–192 (2007)
17. Hutter, M., Poland, J.: Adaptive online prediction by following the perturbed leader. Journal of Machine Learning Research **6**(Apr), 639–660 (2005)
18. Kakade, S., Langford, J.: Approximately optimal approximate reinforcement learning. In: International Conference on Machine Learning (ICML). vol. 2, pp. 267–274 (2002)
19. Kazerouni, A., Ghavamzadeh, M., Yadkori, Y.A., Van Roy, B.: Conservative contextual linear bandits. In: Neural Information Processing Systems (NeurIPS). pp. 3910–3919 (2017)

20. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
21. Koolen, W.M.: The pareto regret frontier. In: Neural Information Processing Systems (NeurIPS). pp. 863–871 (2013)
22. Lacoste, A., Luccioni, A., Schmidt, V., Dandres, T.: Quantifying the carbon emissions of machine learning. arXiv preprint arXiv:1910.09700 (2019)
23. Lattimore, T.: The pareto regret frontier for bandits. In: Neural Information Processing Systems (NeurIPS). pp. 208–216 (2015)
24. Maas, A., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: Annual meeting of the association for computational linguistics: Human language technologies. pp. 142–150 (2011)
25. Papini, M., Pirotta, M., Restelli, M.: Adaptive batch size for safe policy gradients. In: Neural Information Processing Systems (NeurIPS). pp. 3591–3600 (2017)
26. Papini, M., Pirotta, M., Restelli, M.: Smoothing policies and safe policy gradients. arXiv preprint arXiv:1905.03231 (2019)
27. Pirotta, M., Restelli, M., Bascetta, L.: Policy gradient in lipschitz markov decision processes. Machine Learning **100**(2-3), 255–283 (2015)
28. Pirotta, M., Restelli, M., Pecorino, A., Calandriello, D.: Safe policy iteration. In: International Conference on Machine Learning (ICML). pp. 307–315 (2013)
29. Reddi, S.J., Kale, S., Kumar, S.: On the convergence of adam and beyond. arXiv preprint arXiv:1904.09237 (2019)
30. Sani, A., Neu, G., Lazaric, A.: Exploiting easy data in online optimization. In: Neural Information Processing Systems (NeurIPS) (2014)
31. Schulman, J., Levine, S., Abbeel, P., Jordan, M.I., Moritz, P.: Trust region policy optimization. In: International Conference on Machine Learning (ICML). pp. 1889–1897 (2015)
32. Shalev-Shwartz, S., et al.: Online learning and online convex optimization. Foundations and trends in Machine Learning **4**(2), 107–194 (2011)
33. Streeter, M., McMahan, H.B.: No-regret algorithms for unconstrained online convex optimization. arXiv preprint arXiv:1211.2260 (2012)
34. Sui, Y., Burdick, J., Yue, Y., et al.: Stagewise safe bayesian optimization with gaussian processes. In: International Conference on Machine Learning (ICML). pp. 4781–4789. PMLR (2018)
35. Sui, Y., Gotovos, A., Burdick, J., Krause, A.: Safe exploration for optimization with gaussian processes. In: International Conference on Machine Learning (ICML). pp. 997–1005. PMLR (2015)
36. Tange, O.: GNU parallel 2018. Lulu. com (2018)
37. Tsymbal, A.: The problem of concept drift: definitions and related work. Computer Science Department, Trinity College Dublin **106**(2), 58 (2004)
38. Vapnik, V.: Principles of risk minimization for learning theory. In: Neural Information Processing Systems (NeurIPS). pp. 831–838 (1992)
39. Vittori, E., Bernasconi de Luca, M., Trovò, F., Restelli, M.: Dealing with transaction costs in portfolio optimization: Online gradient descent with momentum. In: ACM International Conference on AI in Finance (ICAIF). pp. 1–8 (2020)
40. Wu, Y., Shariff, R., Lattimore, T., Szepesvári, C.: Conservative bandits. In: International Conference on Machine Learning (ICML). pp. 1254–1262 (2016)
41. Zinkevich, M.: Online convex programming and generalized infinitesimal gradient ascent. In: International Conference on Machine Learning (ICML). pp. 928–936 (2003)

Supplementary Material for Paper "Conservative Online Convex Optimization"

A Proofs and Additional Lemmas

In this section we provide the full proofs of the theorems we deferred in the main paper. Moreover, we add some remarks on the online-to-batch conversion of the proposed method, and its corresponding guarantees.

Lemma 4. *Consider any OCO algorithm $\mathcal{A}$ which guarantees a regret of $R_T(\mathcal{A}) \leq \rho \log(T)$. The last time the CP algorithm using $\mathcal{A}$ as subroutine plays the default parameter $\tilde{\theta}$,* i.e., *$\beta_t > 0$, is upper-bounded by τ, defined as:*

$$\tau := \frac{\alpha e^2 \mu (DG + \alpha\mu) + 2\rho \left(\sqrt{\alpha e^2 \mu (DG + \alpha\mu) + \rho^2} + \rho \right)}{e^2 \alpha^2 \mu^2}.$$

Proof. The CP algorithm plays at each time: $\theta_t = \beta_t \tilde{\theta} + (1 - \beta_t) z_t$, where $\beta_t = 1 - \frac{r_t}{||z_t - \tilde{\theta}||_2}$, and z_t is generated by a no-regret algorithm $\mathcal{A}$. From Equation (2) we know that:

$$\frac{L_{t-1} - (1+\alpha)\tilde{L}_{t-1} - \alpha\epsilon_l}{GD} + 1 \geq \beta_t.$$

By using Equation (21) we have that eventually there is a time t for which:

$$\frac{1}{GD} \left[\rho \log(t) - \alpha\mu(t-1) \right] + 1 = 0, \tag{22}$$

as the left hand side goes to zero for t sufficiently large.

Finding the τ for which β_t becomes zero is equivalent to solving an equation of the type $A \log t = Bt - C$ with $A, B, C > 0$, which has no analytical roots. Thanks to the fact that the logarithm is concave, upper-bounding it in Equation (22) results in an equation whose result gives an upper bound on the solution of the original equation. Using that $\log x < \frac{2\sqrt{x}}{e}$, holding for each $x > 0$, the upper bound on the solution is of Equation (22) is provided by:

$$\frac{1}{GD} \left[\rho \frac{2\sqrt{t}}{e} - \alpha\mu(t-1) \right] + 1 = 0,$$

whose solution concludes the proof.

B Baseline approaches

In what follows we present the CS and CRDG algorithms, which will be used as baseline for our experiments.

Algorithm 2 CS

Require: Online learning algorithm $\mathcal{A}$, conservativeness level $\alpha > 0$, default parameter $\tilde{\theta} \in \Theta$

1: Set $\tilde{L}_0 \leftarrow 0$, $L_0 \leftarrow 0$
2: **for** $t \in [T]$ **do**
3: **if** $L_{t-1} + \epsilon_u - (1+\alpha)\epsilon_l \leq \tilde{L}_{t-1}(1+\alpha)$ **then**
4: $z_t \leftarrow \mathcal{A}(f_{t-1}(z_{t-1}))$
5: Select $\theta_t \leftarrow z_t$
6: **else**
7: $z_t \leftarrow z_{t-1}$
8: Select $\theta_t \leftarrow \tilde{\theta}$
9: **end if**
10: Suffer loss $f_t(\theta_t)$
11: Observe feedback $f_t(z_t)$ and $f_t(\tilde{\theta})$
12: **end for**

B.1 The Conservative Switching Algorithm

In this section, we design a more immediate approach to solve the COCO problem, which will be compared with the CP algorithm in the experimental section. The algorithm follows the idea by [40], and adapts it to the OCO setting: play the action z_t only if the conservativeness constraint is satisfied at round t and the current budget is big enough to sustain any loss at the next iteration; otherwise, play the default parameter $\tilde{\theta}$. The complete pseudo-code implementing this approach in the COCO setting, namely *Conservative Switching* (CS), is presented in Algorithm 2. The algorithm works as follows: at each round t, the algorithm computes its budget (Algorithm 2, Line 5). If the current budget is large enough to ensure the conservative constraint is satisfied even after suffering the loss at round t (Line 3), CS queries the action z_t from the no-regret algorithm $\mathcal{A}$ (Line 5). Otherwise, CS plays the default action $\tilde{\theta}$ (Line 8) keeping fixed the optimistic action z_t.

In what follows, we prove that the CS algorithm satisfies the conservativeness constraint in Equation (1) and has sublinear regret bound of the same order of the underlying algorithm $\mathcal{A}$.

Theorem 5. *The CS algorithm applied to a generic online learning algorithm $\mathcal{A}$ is conservative.*

Proof. Let k be a time in which we played the optimistic action z_t given by algorithm $\mathcal{A}$, otherwise the constraint is trivially verified by the fact that the default parameter is inside the conservative ball. In this specific case we have that the following condition (Line 3 in Algorithm 2) is satisfied:

$$\begin{aligned} L_{k-1} + \epsilon_u - (1+\alpha)\epsilon_l &\leq \tilde{L}_{k-1}(1+\alpha) \\ L_{k-1} &\leq \tilde{L}_{k-1}(1+\alpha) + (1+\alpha)\epsilon_l - \epsilon_u. \end{aligned} \tag{23}$$

Moreover, we have that, due to the fact that the loss function is bounded from below by ϵ_l, we have:

$$\tilde{L}_{k-1}(1+\alpha) + (1+\alpha)\epsilon_l \leq \tilde{L}_{k-1}(1+\alpha) + f_k(\tilde{\theta})(1+\alpha) = \tilde{L}_k(1+\alpha). \tag{24}$$

The loss of the CS algorithm becomes:

$$\begin{aligned} L_k &= L_{k-1} + f_k(z_k) \\ &\leq \underbrace{\tilde{L}_{k-1}(1+\alpha) + (1+\alpha)\epsilon_l}_{\leq \tilde{L}_k(1+\alpha)} \underbrace{-\epsilon_u + f_k(z_k)}_{\leq 0} \qquad (25) \\ &\leq \tilde{L}_k(1+\alpha), \qquad (26) \end{aligned}$$

where Equation (25) follows from the fact that we played the $\mathcal{A}$ algorithm for the round k, thus the condition in Equation (23) holds, and Equation (26) is derived using Equation (24) and from the fact that the loss function is bounded from below by ϵ_l. This concludes the proof.

Theorem 6. *Consider any OCO algorithm $\mathcal{A}$ which guarantees a regret of $R_T(\mathcal{A}) \leq \xi\sqrt{T}$. The CS algorithm using $\mathcal{A}$ as subroutine has the following regret bound:*

$$R_T(CS) \leq \xi\sqrt{T} + \tau DG,$$

where:

$$\tau := \frac{\xi^2 - 2\alpha\mu(\epsilon_u - \epsilon_l)}{2\alpha^2\mu^2} + \frac{1}{2}\sqrt{\frac{\xi^4 + 4\alpha\xi^2\mu(\epsilon_u - \epsilon_l)}{\alpha^4\mu^4}}.$$

Proof. Let us define $C_\alpha := \epsilon_u - (1+\alpha)\epsilon_l$ and let $k \geq 1$ be a time in which we played the default strategy and define S and R as the set of rounds in which the CS algorithm played the default parameter and the parameter chosen by $\mathcal{A}$ up to time k, respectively. Formally:

$$\begin{aligned} S &= \{t \leq k \text{ s.t. } (1+\alpha)\tilde{L}_{t-1} - L_{t-1} \leq C_\alpha\}, \\ V &= \{t < k \text{ s.t. } (1+\alpha)\tilde{L}_{t-1} - L_{t-1} > C_\alpha\}. \end{aligned}$$

By definition of the cumulative loss L_{k-1} and since $S \cup V = [k-1]$, we have:

$$\begin{aligned} \sum_{t\in V} f_t(z_t) &= L_{k-1} - \sum_{t\in S\setminus\{k\}} f_t(\tilde{\theta}) \\ &\geq (1+\alpha)\tilde{L}_{k-1} - C_\alpha - \sum_{t\in S\setminus\{k\}} f_t(\tilde{\theta}) \\ &= (1+\alpha)\tilde{L}_{k-1} - C_\alpha \underbrace{- \sum_{t\in S\setminus\{k\}} f_t(\tilde{\theta}) - \sum_{t\in V} f_t(\tilde{\theta})}_{=-\tilde{L}_{k-1}} + \sum_{t\in V} f_t(\tilde{\theta}) \\ &= \alpha\tilde{L}_{k-1} - C_\alpha + \sum_{t\in V} f_t(\tilde{\theta}), \end{aligned}$$

where the first inequality follows from the fact that $k \in \tilde{S}$ and, therefore, $L_{k-1} \geq (1+\alpha)\tilde{L}_{k-1} - C_\alpha$. Finally, using that $\tilde{L}_k > \mu k$, since the default parameter $\tilde{\theta}$ is not a no-regret strategy we get:

$$\sum_{t \in V} f_t(z_t) \leq \alpha \tilde{L}_{k-1} - C_\alpha + \sum_{t \in V} f_t(\tilde{\theta}) \tag{27}$$

$$\sum_{t \in V} [f_t(z_t) - f_t(\tilde{\theta})] \geq k\alpha\mu - (\epsilon_u - \epsilon_l). \tag{28}$$

Since the algorithm $\mathcal{A}$ has been run only on the set V, the left hand side is bounded by the regret of $\mathcal{A}$ on the set V, and, consequently, also on the entire time horizon k. Taking the limit $k \to +\infty$, there will be a time τ in which Equation (28) is not verified anymore, proving that the last time the algorithm plays the default parameter satisfies $t_0 \leq \tau < +\infty$.

Solving for τ the following equation:

$$\xi\sqrt{\tau} = \alpha\tau\epsilon_l - (\epsilon_u - \epsilon_l),$$

we get:

$$\tau := \frac{\xi^2 - 2\alpha\mu(\epsilon_u - \epsilon_l)}{2\alpha^2\mu^2} + \frac{1}{2}\sqrt{\frac{\xi^4 + 4\alpha\xi^2\mu(\epsilon_u - \epsilon_l)}{\alpha^4\mu^4}}.$$

With this result we can bound the regret of the CS algorithm as follows:

$$R_T(CS) \leq \xi\sqrt{T} + \tau DG.$$

Theorem 7. *Consider any OCO algorithm $\mathcal{A}$ which guarantees a regret of $R_T(\mathcal{A}) \leq \rho\log(T)$. The CS algorithm using $\mathcal{A}$ as subroutine has the following regret bound:*

$$R_T(CS) \leq \rho\log(T) + \tau DG,$$

where:

$$\tau := \frac{2\rho^2 + \alpha e^2\mu(\epsilon_u - \epsilon_l)}{\alpha^2 e^2\mu^2} + 2\sqrt{\frac{\rho^4 + \alpha e^2\rho^2\mu(\epsilon_u - \epsilon_l)}{\alpha^4 e^4\mu^4}}.$$

Proof. The proof follows the same steps as the one of Theorem 6 up to Equation (28).

Now the left hand side can be bounded by $\rho\log(k)$ that, on its turn, is bounded as $\rho\log k \leq \rho\frac{2\sqrt{k}}{e}$, which holds for $k > 1$. Thanks to this inequality the value of an upper bound τ on the value of the last instant CS plays the default parameter $\tilde{\theta}$ is provided by the analytical solution to the following equation:

$$\rho\frac{2\sqrt{\tau}}{e} = \alpha\tau\epsilon_l - (\epsilon_u - \epsilon_l).$$

Algorithm 3 *RD-1D*

Require: Learning rate η_1, upper bound $\bar{H}$, initial parameter θ_0
1: Set $i \leftarrow 1$, $Q_1 \leftarrow 0$
2: **for** $t \in [T]$ **do**
3: Play θ_t and suffer loss $f_t(\theta)$
4: $Q_i \leftarrow Q_i - f_t(\theta_t)$
5: **if** $Q_i < \eta_i \bar{H}$ **then**
6: $\theta_{t+1} \leftarrow \theta_t - \eta_1 \nabla f_t(\theta_t)$
7: **else**
8: $i \leftarrow i + 1$
9: $Q_i \leftarrow 0$
10: $\eta_i \leftarrow 2\eta_{i-1}$
11: $\theta_t = \theta_0 - \eta_1 \nabla f_t(\theta_t)$
12: **end if**
13: **end for**

With the above result we can bound the regret of the CP algorithm as follows:

$$R_T(CS) \leq \rho \log T + \tau DG,$$

which concludes the proof.

Even if from these results it is not possible to state that CP attains a strictly better regret than CS, we will show through an experimental campaign that CP achieves a better empirical performance. The intuition behind this superior performance is that during the first phase of the optimization, *i.e.,* $r_t < D$, we are less constrained using the CP algorithm since we are allowed to select the parameter for the next round on the conservative ball $B(\tilde{\theta}, r_t)$ border. Conversely, the CS algorithm plays the default parameter $\tilde{\theta}$ until enough budget is collected. Concerning the computational cost of the CS algorithm, it has a constant computational overhead w.r.t. the original algorithm $\mathcal{A}$ due to the evaluation of the losses $f_t(\theta_t)$, and $f_t(\tilde{\theta})$.

B.2 The Constrained Reward Doubling Guess Algorithm

In this section we provide the description of the Conservative Reward Doubling Guess (CRDG). The pseudo-code of the CRDG algorithm is provided in Algorithms 3-6.

In particular, the RD-1D algorithm, presented in Algorithm 3, performs a search, using a gradient descend approach, on the space $\mathbb{R}$ (Lines 6 and 11), and restarts from the point θ_0 (Line 11) every times it collects enough wealth, formally, if $Q_i \geq \eta_i \bar{H}$. At every restart, it doubles its learning rate (Line 10). Notice that the RD-1D algorithm requires the knowledge of an upper bound on the variance of the loss gradients $\bar{H} \geq \sum_{t=1}^{T} (\nabla f_t)^2$. Conversely, if the quantity $\bar{H}$ is unknown, one can resort to the RD-1D-Guess algorithm, presented in Algorithm 4. This algorithm performs the doubling trick [5] on the quantity $\bar{H}$, using the RD-1D algorithm as a subroutine.

Algorithm 4 *RD-1D-Guess*

Require: Learning rate ε, initial parameter θ_0
Set $i \leftarrow 1$, $H_i = 1$, $\eta_i = \varepsilon$, $H = 0$
while $t \in [T]$ **do**
  $\mathcal{A} = RD\text{-}1D(\eta_i, H_i, \theta_0)$
  while $H \leq H_i$ **do**
    Play θ_t from algorithm $\mathcal{A}$ and suffer loss $f_t(\theta_t)$
    $H \leftarrow H + \nabla f_t(\theta_t)^2$
    $t \leftarrow t + 1$
  end while
  $i \leftarrow i + 1$
  $H = 0$
  $H_i = 2H_{i-1}$
  $\eta_i = \eta_{i-1}/4$
end while

Algorithm 5 *RD-ND-Guess*

Require: Learning rate ε, initial parameter θ_0
for $k \in [d]$ **do**
  Set $\mathcal{A}_k = RD\text{-}1D\text{-Guess}(\varepsilon/k, \theta_{0,k})$
end for
while $t \in [T]$ **do**
  for $k \in [d]$ **do**
    Get $\theta_{t,k}$ from $\mathcal{A}_k$
    Play θ_t and suffer loss $f_t(\theta_t)$
  end for
end while

Algorithm 6 *CRDG*

Require: Learning rate ε, initial parameter θ_0, parameter set Θ
Set $\mathcal{A} = RD\text{-}ND\text{-Guess}(\varepsilon, \theta_0)$
for $t \in [T]$ **do**
  Get z_t from $\mathcal{A}$
  Play $\theta_t = \Pi_\Theta(z_t)$ and suffer loss $f_t(\theta)$
  Observe the gradient of the loss $\nabla f_t(\theta_t)$
  Update $\mathcal{A}$ using $\nabla g_t(\theta_t)$
end for

The extension of the RD-1D-Guess algorithm to parameter spaces $\Theta \subseteq \mathbb{R}^d$, with $d > 1$, is provided by RD-ND-Guess, which uses an instance RD-1D-Guess as subroutine, applying it to each one of the d coordinates separately. The pseudo-code of the RD-ND-Guess is presented in Algorithm 6, which, at the beginning, sets d instances $\{\mathcal{A}_1, \ldots, \mathcal{A}_D\}$ of the RD-1D-Guess algorithm, and at round t, selects the k-th component $\theta_{t,k}$ of the parameter θ_t querying the algorithm $\mathcal{A}_k$. Each Algorithm $\mathcal{A}_k$ is run by providing it with the k-th coordinate of the gradient of the loss $\nabla f_t(\theta_{t,k})$.

The aforementioned algorithms have been designed to work in unconstrained domains. However, they can be adapted to work in a convex parameter space Θ, by utilizing the *Constrained Set Reduction* (CSR) Algorithm described in [10]. The CRDG algorithm, presented in Algorithm 6, describes the CSR meta-algorithm applied to Algorithm 6. It works by projecting the parameter predicted by the CRDG algorithm into the set Θ (Line 3). Moreover, it requires that the losses fed to the CRDG algorithm are redefined to penalize parameter outside the parameter space Θ using the following pseudo-loss function:

$$g_t(x) := \frac{1}{2}\left[\langle x, \nabla f_t(\theta_t)\rangle + ||\nabla f_t(\theta_t)||_2 S_\Theta(x)\right],$$

where $S_\Theta(x) := \arg\inf_{y\in\Theta} ||x-y||_2$ is the distance between x and the set Θ. Finally, the gradient of such a function is used to update the subroutine (Line 6).

The Reward Doubling Guess (RDG) algorithm has been proposed by [33] to solve instance of the Unconstrained Online Optimization problem. In this framework the guarantees are given w.r.t. a so called *comparator parameter* $\mathring{\theta} \in \mathbb{R}^d$. When the algorithm starts from the origin of $\mathbb{R}^d$, the RDG algorithm provides a regret bound of:

$$R_T(\mathring{\theta}) \leq ||\mathring{\theta}||_2 \sqrt{T} \log\left[\frac{d(1+||\mathring{\theta}||_2)T}{\varepsilon}\right], \tag{29}$$

where $\varepsilon > 0$ is the learning rate of the procedure. Without loss of generality the algorithm can start from a generic point in $\mathbb{R}^d$, and restate the bound in terms of the distance from the starting point and the general comparator parameter $\mathring{\theta}$. In the case the comparator parameter is the starting point of the algorithm the RDG algorithm, it guarantees a regret of:

$$R_T(\mathring{\theta}) \leq \varepsilon.$$

The use of the RDG algorithm in a constrained setting, *i.e.,* over a convex parameter set $\Theta \subset \mathbb{R}^d$, requires to use a conversion provided by the *Constraint Set Reduction* algorithm described by [10]. We named *Constraint Reward Doubling Guess* (CRDG) the combination of this algorithm together with the RDG algorithm. Thanks to the reduction algorithm we project onto the set Θ, and convert any algorithm with regret R_T in the unconstrained setting to an algorithm that guarantees $2R_T$ in the constrained one. Overall, the resulting algorithm provides the following guarantee:

$$R_T(\mathring{\theta}) \leq 2||\mathring{\theta}||_2 \sqrt{T} \log\left[\frac{d(1+||\mathring{\theta}||_2)T}{\varepsilon}\right], \quad \forall \mathring{\theta} \in \Theta, \tag{30}$$

and

$$R_T(\tilde{\theta}) \leq 2\varepsilon.$$

for a specific known parameter $\tilde{\theta} \in \Theta$.

To guarantee the budget constraint of Equation (1) as required by the COCO framework, we need to set $\varepsilon = \mu\alpha/2$ in the CRDG algorithm. Two main difference emerge from this analysis. First, setting such ε requires the a priori knowledge of the parameter μ, which conversely is not necessary to run either the CP or the CS algorithms. Second, the regret bound in Equation (30) has an extra term of $\log(T)$ compared with ones provided by the CP and CS algorithms. Moreover, the choice of the learning rate as $\varepsilon = \mu\alpha/2$ is too conservative to provide a performing algorithm in practice, as showed in experiments of Section 5.

C Online to batch conversion for COCO

In this section, we present the results implied by a low regret and conservative algorithm in the COCO setting in the case *i.i.d.* examples drawn from a fixed distribution. In particular, we are concerned with minimizing the expected error $e(h_\theta) = \mathbb{E}_{x,y\sim\mathcal{D}}[l(h_\theta(x), y)]$, where $x \in \mathcal{X}, y \in \mathcal{Y}$ are examples drawn from a joint distribution $\mathcal{D}$ over the space $\mathcal{X} \times \mathcal{Y}$, $h_\theta : \mathcal{X} \to \hat{\mathcal{Y}}$ is the prediction function parameterized by θ, and $l : \hat{\mathcal{Y}} \times \mathcal{Y} \to \mathbb{R}$ is a loss function convex in the first argument. If we run an online algorithm over the samples (x_t, y_t) with $f_t(\theta_t) = l(h_{\theta_t}(x_t), y_t)$, it has been shown that: [8] Any online algorithm $\mathfrak{U}$ that attains regret $R_T(\mathfrak{U})$ satisfies, with probability at least $1 - \delta$:

$$e(h_{\langle\theta\rangle_T}) - \inf_{\theta\in\Theta} e(h_\theta) \leq \frac{R_T(\mathfrak{U})}{T} + 2\sqrt{\frac{2\log(2/\delta)}{T}},$$

where $\langle\theta\rangle_T := \frac{1}{T}\sum_{t=1}^{T}\theta_t$, and θ_t are the parameters predicted by the online algorithm $\mathfrak{U}$.

In the COCO setting, it is not sufficient that this conversion maintains the regret guarantees since we are also concerned with the conservativeness constraint. It is possible to show that the expected error of the strategy running CP is close to that of the chosen default strategy $\tilde{\theta}$:

An algorithm $\mathfrak{U}$ that observes the conservativeness constraint, satisfies, with probability at least $1 - \delta$:

$$e(h_{\langle\theta\rangle_T}) - (1+\alpha)e(\tilde{\theta}) \leq 2(1+\alpha)\sqrt{\frac{2\log(1/\delta)}{T}}.$$

Proof. From the proof of Proposition 1 by [8] it is known that, with probability at least $1 - \delta$, the following holds:

$$e(h_{\langle\theta\rangle_T}) \leq \frac{1}{T}L_T + \sqrt{\frac{2\log(1/\delta)}{T}}. \tag{31}$$

On the other hand, let us define $U_t := \sum_{k=1}^{t}[l(h_{\tilde{\theta}}(x_k), y_k) - e(h_{\tilde{\theta}})]$, and notice that U_t is a martingale with respect to the natural filtration of the sequence of random variables $(x_1, y_1), \ldots, (x_t, y_t)$. Applying the Hoeffding-Azuma inequality [4]

on U_t, we get that, with probability at least $1-\delta$, we have:

$$\frac{1}{T}\tilde{L}_T - e(h_{\tilde{\theta}}) \leq \sqrt{\frac{2\log(1/\delta)}{T}}. \tag{32}$$

By combining Equation (31) and Equation (32) we conclude that, with probability at least $1-\delta$, we have:

$$\begin{aligned} & e(h_{\langle\theta\rangle_T}) - (1+\alpha)e(h_{\tilde{\theta}}) \\ & \leq \frac{1}{T}[L_T - (1-\alpha)\tilde{L}_T] + (1+\alpha)\sqrt{\frac{2\log(1/\delta)}{T}} \\ & \leq (1+\alpha)\sqrt{\frac{2\log(1/\delta)}{T}}, \end{aligned}$$

where the last inequality follows from the fact that the algorithm $\mathfrak{U}$ satisfies the conservativeness constraint.

D Experiments: Technical Details and Additional Results

In this section we present the technical details on the results presented in Section 5, and additional results on the synthetically generated setting presented in the experimental section, and add an experiment on a portfolio optimization application.

D.1 Technical Details

The code has been run on a Intel(R) Xeon(R) CPU E5-4610 v2 @ 2.30GHz CPU with 256 GiB of system memory. The operating system was Ubuntu 16.04.4 LTS, and the experiments have been run on Python 3.5.2. The libraries used in the experiments, with the corresponding version were:

- `bootstrapped==0.0.2`
- `certifi==2020.12.5`
- `chardet==4.0.0`
- `cycler==0.10.0`
- `idna==2.10`
- `kiwisolver==1.3.1`
- `matplotlib==3.3.4`
- `numpy==1.20.1`
- `pandas==1.2.2`
- `Pillow==8.1.0`
- `pyparsing==2.4.7`
- `python-dateutil==2.8.1`
- `pytz==2021.1`
- `PyYAML==5.4.1`
- `requests==2.25.1`
- `scipy==1.6.0`
- `six==1.15.0`
- `tqdm==4.56.1`
- `urllib3==1.26.3`

In addition we used *GNU parallel* [36] to parallelize the experiments.

D.2 CO_2 Emissions

Experiments were conducted using a private infrastructure, which has a carbon efficiency of 0.35 kgCO_2 eq/kWh. A cumulative of 15 hours of computation was performed on hardware of type Intel Xeon E5-2699 (TDP of 145W) to perform the code corresponding to the experiments presented in Section 5. Total emissions are estimated to be 0.76 kgCO_2eq of which 0% were directly offset. Carbon footprint estimations have been conducted using the Machine Learning Impact calculator at https://mlco2.github.io/impact#compute presented in [22].

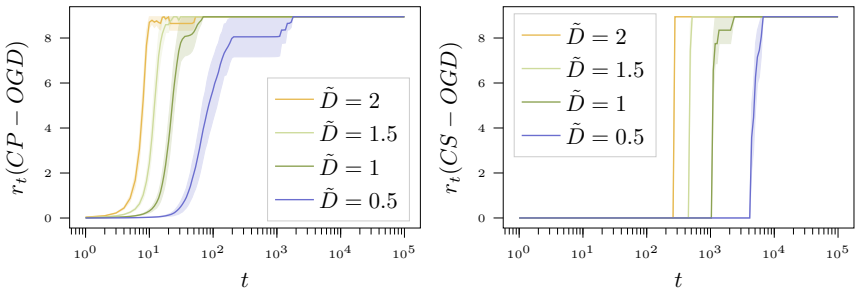

Fig. 6: Evolution of the conservative ball radius r_t: (a) CP-OGD algorithm, (b) CS-OGD algorithm. The confidence intervals on the mean, represented in the figures as semi-transparent areas, represents the 95% confidence intervals and are computed by statistical bootstrap.

D.3 Additional Results on the Synthetic Regression Dataset

We analysed the behaviour of the conservative ball radius r_t over the rounds t, as the value of $\tilde{D}$ varies in the setting of the synthetic regression dataset. Recall that a value of $r_t = D$ implies that the conservative ball includes the parameter space Θ and, therefore, CP-OGD is playing the same parameter vector as OGD. The results are presented in Figure 6, in which the t axis have been represented in logarithmic scale. We reported the results only for $\tilde{D} \leq 2$, as for $\tilde{D} > 2$ the results were not significantly different than those of $\tilde{D} = 2$. Figure 6a shows that the CP-OGD, generally, has a conservative ball radius $r_t \approx D = \sqrt{2d}$ before CS-OGD. This suggests that the ability of CP-OGD being able to increase the radius provides an advantage in terms of regret w.r.t. CS-OGD.

D.4 Online Portfolio Optimization

In what follows we discuss the adoption of the COCO framework in the specific application of *portfolio optimization*.

Problem Formulation Online Portfolio Optimization (OPO) [39] can be modelled as a sequential decision problem of an investor that, at each time t, has to allocate its wealth among d stocks. The allocation is represented by the vector $\theta_t \in \Delta_{d-1} \subset \mathbb{R}^d$. Subsequently, the market chooses the vector of the rewards for the stocks $r_t \in [l, u]^d$, where $l = e^{-\epsilon_l}$ and $u = e^{-\epsilon_u}$, and the investor suffer a loss $f_t(\theta_t) = -\log(\langle \theta_t, r_t \rangle)$. With this choice of loss, we get that the wealth $W_t(\mathfrak{U})$ at time t of an algorithm $\mathfrak{U}$ is defined as:

$$W_t(\mathfrak{U}) = e^{-L_t},$$

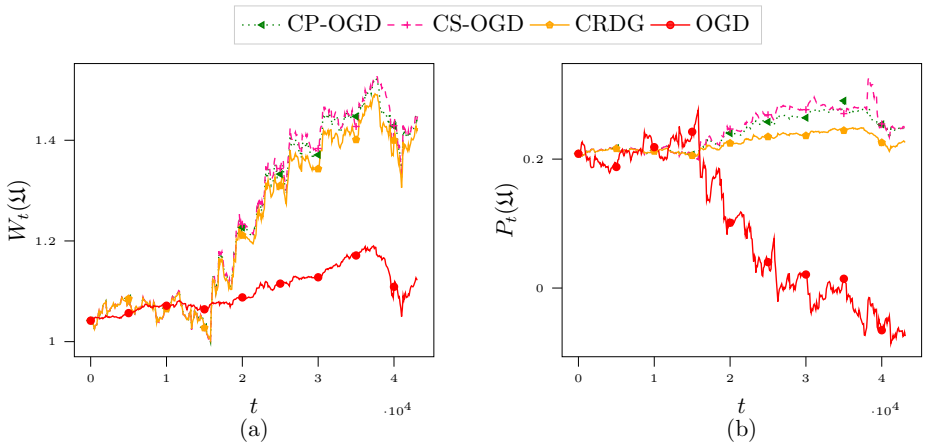

Fig. 7: Results on the financial environment: (a) wealth $W_t(\mathfrak{U})$, (b) wealth budget $P_t(\mathfrak{U})$.

where we recall that $L_t = \sum_{h=1}^{t} f_h(\theta_h)$ is the cumulative loss suffered by the algorithm $\mathfrak{U}$. The default parameter $\tilde{\theta} \in \Delta_{d-1}$ represents the index that an investor wants to outperform over the entire time horizon T.

In the financial context, portfolios are generally compared in terms of wealth. This translates in a different formulation of the conservativeness constraint defined as follows:

$$W_t(\mathfrak{U}) \geq (1-\kappa)\tilde{W}_t \ \forall t \in [T], \tag{33}$$

where $\tilde{W}_t$ the wealth gained by playing $\tilde{\theta}$ over t steps, and $\kappa \in (0,1)$ represents the conservativeness level in this context. It is easy to show the above constraint is implied by the one in Equation (33) using $\alpha = \frac{1-\kappa}{\epsilon_l T}$.

Notice that the log-loss used in the OPO case is not positive, therefore, the use of the CP-OGD and CS-OGD algorithms requires to shift the loss to positive values, *i.e.,* using the following loss function:

$$f_t(\theta_t) = -\log(\langle \theta_t, r_t \rangle) + \epsilon_l. \tag{34}$$

Experimental Results We used a public dataset of 502 stocks collected with minute frequency over from 2017/9/11 to 2018/02/16 for a total of 43,148 days.[7] Among the stocks that had a one-time step return of $\pm 4\%$, we selected 100 random stocks, and chose uniformly from them the default strategy $\tilde{\theta}$. The conservative level κ has been set to $\kappa = 0.2$, the diameter in this setting is $D = \sqrt{2}$ and the gradient is bounded by $G = \frac{\epsilon_u}{\epsilon_l}$, where $\epsilon_l = 0$ and $\epsilon_u = \log(u) - \log(l)$ since we used the shifted loss defined in Equation (34). As for the previous experiments,

[7] https://www.kaggle.com/nickdl/snp-500-intraday-data.

we used $\eta_t = \frac{K}{\sqrt{t}}$, with $K = \frac{D}{\sqrt{2G}}$, as learning rate for all the analysed algorithms. This choice for K minimizes the theoretical bound on the regret of the OGD algorithm.

We evaluated the algorithms in terms of wealth $W(\mathfrak{U})$ and in term of wealth budged, defined as:

$$P_t(\mathfrak{U}) = W_t(\mathfrak{U}) - (1 - \kappa)\tilde{W}_t.$$

Results The results of the experiment are presented in Figure 7. In Figure 7(a) we can see that wealth of the portfolios generated by the CP-OGD and CS-OGD algorithm outperforms the OGD algorithm. This suggests that in some cases the information given by the default strategy can greatly help the performance in a difficult domain such as the financial one. Moreover, the budget of the OGD algorithm, presented in Figure 7(b), does not satisfy the budget constraint of Equation (33), while the CP-OGD and CS-OGD satisfy the constraint at all $t \in [T]$. This confirms the theoretical results provided above, stating that the conservativeness is still assured by CP-OGD and CS-OGD with a specific choice of the parameter α.