ALICE: ADAPTIVE STOCHASTIC GRADIENT DESCENT BASED ON ADVERSARIAL MULTI-ARMED BANDITS

Anonymous authors
Paper under double-blind review

ABSTRACT

Gradient-based optimization algorithms are central to machine learning, and their performance critically depends on the choice of the learning rate. Modern methods adapt learning rates during training using gradient information. In this work, we explore an alternative research direction by viewing learning-rate adaptation through the adversarial bandit setting. We develop a theoretical framework that relates the convergence of adaptive gradient methods to the regret of an adversarial bandit problem. Then, we instantiate our framework using EXP3 within stochastic gradient descent and demonstrate its effectiveness through empirical evaluations.

1 INTRODUCTION

We are interested in solving $\min_{x\in\mathbb{R}^d} f(x)$, where $f:\mathbb{R}^d\to\mathbb{R}^+$ is differentiable, potentially non-convex. Besides, the function f that we aim to optimize can be rewritten in the following way.

Assumption 1. *We have $f(x)=\mathbb{E}_\xi f_\xi(x)$, where ξ parametrizes the function f_ξ.*

Moreover, we assume that ∇f_ξ is an unbiased estimator of ∇f, *i.e.*, we have the following assumption.

Assumption 2. *We have $\forall x\in\mathbb{R}^d,\ \mathbb{E}_\xi[\nabla f_\xi(x)]=\nabla f(x)$.*

In the empirical risk minimization setting, the function f is defined from a loss $\ell:\mathbb{R}^d\times\mathcal{Z}\to\mathbb{R}^+$ and a dataset $\mathcal{S}\subseteq\mathcal{Z}$ as $f(x)=R^\ell_{\mathcal{S}}(x)=\frac{1}{|\mathcal{S}|}\sum_{z\in\mathcal{S}}\ell(x,z)$, where $x\in\mathbb{R}^d$ denotes the model parameters and $\mathcal{Z}$ is the data space. Assumptions 1 and 2 are recovered by sampling a mini-batch $\mathcal{B}\subseteq\mathcal{S}$ uniformly at random and defining $\xi=\mathcal{B}$, and $f_\xi(x)=R^\ell_{\mathcal{B}}(x)=\frac{1}{|\mathcal{B}|}\sum_{z\in\mathcal{B}}\ell(x,z)$.

A classical iterative optimization method to solve our considered minimization problem is the Stochastic Gradient Descent (SGD) algorithm, originally introduced by Cauchy (1847) and later formalized by Robbins (1952); the algorithm is recalled in Algorithm 1.

Algorithm 1 Stochastic Gradient Descent Algorithm

1: Initialize $x_0\in\mathbb{R}^d$
2: **for** $t=0$ to $T-1$ **do**
3: Define the learning rate $\gamma_t\in\mathbb{R}$
4: Perform the gradient step $x_{t+1}\leftarrow x_t-\gamma_t\nabla f_{\xi_t}(x_t)$
5: **return** x_T

Starting from the initial point $x_0\in\mathbb{R}^d$, the algorithm computes at each iteration $t\in\{0,\dots,T-1\}$ the gradient of the noisy function f_{ξ_t} evaluated at x_t. Then, a new point x_{t+1} is computed using the general update rule $x_{t+1}=x_t-\gamma_t\nabla f_{\xi_t}(x_t)$.

Algorithm 1 encompasses many optimization algorithms, such as AdaGrad-Norm (Ward et al., 2019), D-Adapted SGD (Defazio & Mishchenko, 2023), DoG (Ivgi et al., 2023), or DoWG (Khaled et al., 2023), which start from an initialized learning rate γ_0 and adapt γ_t for the next iterations as a function of *(i)* the previous iterates $x_0,\dots,x_{t-1}$, and *(ii)* the (stochastic) gradients $\nabla f_{\xi_0}(x_0),\dots,\nabla f_{\xi_{t-1}}(x_{t-1})$. Other popular methods following the same learning rate adaptation consider learning rates γ_t in $\mathbb{R}^d$ (*i.e.*, one learning rate per component); see *e.g.*, RMSProp (Tieleman

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

& Hinton, 2012), AdaGrad (Duchi et al., 2011), or Adam (Kingma & Ba, 2015). Other methods run several instances of SGD in parallel: Restarted/tuned SGD (Khaled & Jin, 2024; Attia & Koren, 2024) uses different learning rates and aggregates the iterates to obtain the final iterate, while AutoSGD (Surjanovic et al., 2025) runs three different learning rates in parallel before choosing the next iterate as the result of one of the three SGDs.

In this paper, we propose a novel framework that leverages adversarial bandit algorithms, specifically EXP3 (Auer et al., 2002), to dynamically adapt learning rates in SGD, establishing what is, to the best of our knowledge, a previously unexplored connection between bandit learning theory and SGD optimization. Our key contribution is a theoretical convergence bound directly related to the regret of this adversarial bandit problem.

2 Analysis of Stochastic Gradient Descent

In this section, we introduce the assumptions for the analysis of Alice and that is also used for SGD. Then, we present a well-known convergence bound of SGD on which our guarantees are based on.

2.1 Assumptions

Our goal is to solve the minimization problem by finding $x \in \mathbb{R}^d$ that minimizes a given differentiable function $f : \mathbb{R}^d \to \mathbb{R}^+$ (thanks to a variant of Algorithm 1). Hence, our goal is to find iterates $x_1, \dots, x_{T-1} \in \mathbb{R}^d$ such that the gradient $\nabla f(x_{T-1})$ is close to zero indicating that x_{T-1} is near a stationary point, *i.e.*, either a local minimum or a saddle point. In order to keep track of whether the gradients get smaller over time, we are interested in the cumulative squared gradient norms defined by $\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2$ that we aim to minimize. In the paper, we make different assumptions, which are described below. Our main assumption in this paper is the smoothness of the function f.

Assumption 3. *Let $L > 0$, a differentiable function $f : \mathbb{R}^d \to \mathbb{R}^+$ is L-smooth if ∇f is L-Lipschitz, i.e., we have $\forall x, y \in \mathbb{R}^d,\ \|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$.*

Intuitively, Assumption 3 means that the gradient of the function does not change too abruptly. Besides, we may assume that the gradient norm is bounded, *i.e.*, we have the following assumption.

Assumption 4. *The function f has a bounded gradient if $\forall x \in \mathbb{R}^d,\ \|\nabla f(x)\|^2 \leq M$.*

When we are dealing with the stochastic gradient ∇f_ξ, we have to make an assumption about how the true gradient $\nabla f(x)$ is related to $\nabla f_\xi(x)$. Hence, in the following assumption, we consider a trajectory-level variant of the strong growth condition (Schmidt & Roux, 2013; Vaswani et al., 2019), which captures situations where the sum of $\|\nabla f_{\xi_t}(x_t)\|^2$ along the trajectory produced by Algorithm 1 does not diverge more than linearly with respect to the sum of $\|\nabla f(x_t)\|^2$. This is milder than the classical pointwise strong growth condition, as it only needs to hold along the realized trajectory, avoiding a worst-case bound over $\mathbb{R}^d$.

Assumption 5. *If Algorithm 1 is run for T iterations from $x_0 \in \mathbb{R}^d$, we have*

$$\mathbb{E}_{\boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}}\left[\sum_{t=0}^{T-1} \|\nabla f_{\xi_t}(x_t)\|^2\right] \leq \alpha \cdot \mathbb{E}_{\boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}}\left[\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2\right] + \beta.$$

2.2 Convergence Bound

The convergence of SGD in the non-convex setting has been studied (Ghadimi & Lan, 2013); the following theorem presents a classical convergence guarantee for this setting.

Theorem 1. *Suppose that Assumption 2 and Assumption 3 hold. If Algorithm 1 is run with $\gamma_t = \gamma > 0$, then, we have*

$$\mathbb{E}_{\boldsymbol{\xi}_{T-1}}\left[\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2\right] \leq \mathbb{E}_{\boldsymbol{\xi}_{T-1}}\left[\frac{f(x_0) - f(x_T)}{\gamma} + \frac{\gamma L}{2} \sum_{t=0}^{T-1} \|\nabla f_{\xi_t}(x_t)\|^2\right].$$

Theorem 1 highlights that the convergence behavior of the gradient descent, as measured by the cumulative squared gradient norms, is strongly influenced by the choice of the learning rate $\gamma \in \mathbb{R}$. An

inappropriate value of γ can significantly degrade performance, leading to suboptimal convergence guarantees. In Appendix A.1, we give two examples of convergence bounds. However, the learning rates associated with these guarantees depend on various constants, such as L, M, α, or β appearing in Assumptions 3 to 5, which cannot be determined in practice. Instead, a grid search is performed to obtain a good rate. Note that we leverage Theorem 1 in Section 4 to exhibit the relationship between the adversarial bandits and the convergence of Algorithm 1.

3 Adversarial Bandit

To tackle the problem of adaptively selecting the learning rate γ_t of Algorithm 1, we make use of the adversarial bandit framework to adapt γ_t. We recall below the classical adversarial bandit setting.

3.1 Setting

The adversarial multi-armed bandit setting, introduced by Auer et al. (2002), can be formalized as follows. Let $K \in \mathbb{N}$ denote the number of available actions, indexed by the set $[\![1, K]\!] = \{1, \dots, K\}$. At each time step $t \in [\![0, T-1]\!] = \{0, \dots, T-1\}$, with $T \in \mathbb{N}$, a learner picks an action $\mathsf{I}_t \in [\![1, K]\!]$ through sampling a probability distribution p_t, and receives a corresponding feedback. In our context, a feedback at t is given in term of loss $\mathsf{f}_t(\mathsf{I}_t)$. Crucially, the learner only observes $\mathsf{f}_t(\mathsf{I}_t)$ at step t for the sampled action I_t and does not have access to the losses of the other actions. The goal of the learner is to minimize the cumulative loss, $\sum_{t=0}^{T-1} \mathsf{f}_t(\mathsf{I}_t)$. In practice, the learner corresponds to a learning algorithm, and its performance is typically measured through the regret, which quantifies how well it performs compared to the best fixed action.

Definition 1. *The regret against action $j \in [\![1, K]\!]$ of an algorithm is defined as*

$$R_T(j) = \mathbb{E}_{\mathbf{I}_{T-1}}\left[\sum_{t=0}^{T-1} \mathsf{f}_t(\mathsf{I}_t)\right] - \mathbb{E}_{\mathbf{I}_{T-1}}\left[\sum_{t=0}^{T-1} \mathsf{f}_t(j)\right],$$

where $\mathbb{E}_{\mathsf{I}_t}[\cdot]$ denotes the expectation $\mathbb{E}_{\mathsf{I}_0 \sim \mathsf{p}_0, \dots, \mathsf{I}_t \sim \mathsf{p}_t}[\cdot]$.

In this setting, the pseudo-regret $\max_{j \in [\![1,K]\!]} R_T(j)$ is usually considered instead of $R_T(j)$. In this paper, we focus on the regret $R_T(j)$ against action j, which we simply refer to as the regret hereafter. We will also make use of a second loss sequence, under the following assumption.

Assumption 6. *For all $t \in [\![0, T-1]\!]$ and $i \in [\![1, K]\!]$, we have $0 \le \mathsf{f}'_t(i) \le \mathsf{f}_t(i)$.*

We define another regret to better capture the link with the convergence in Theorem 3 and Corollary 1.

Definition 2. *Under Assumption 6, the regret against action $j \in [\![1, K]\!]$ is defined as*

$$R'_T(j) = \mathbb{E}_{\mathbf{I}_{T-1}}\left[\sum_{t=0}^{T-1} \mathsf{f}'_t(\mathsf{I}_t)\right] - \mathbb{E}_{\mathbf{I}_{T-1}}\left[\sum_{t=0}^{T-1} \mathsf{f}_t(j)\right].$$

3.2 EXP3

To minimize the regret in the adversarial bandit setting, a classical algorithm is EXP3. We now recall this algorithm in Algorithm 2, introduced by Auer et al. (2002). Intuitively, Algorithm 2 maintains a probability distribution p_t (with an exponential weighting scheme) over the K actions at each round t (Lines 3), which gradually shifts to favor actions that appear to perform better, based on the observed loss $\mathsf{f}_t(\mathsf{I}_t)$ at each step t. Since the algorithm only observes the loss of the selected action I_t (sampled in Line 4), p_t is updated thanks to an (unbiased) importance-weighted estimator to approximate the losses of all actions (Lines 5 and 6). We now state upper bounds on the Algorithm 2's regret $R_T(j)$ against $j \in [\![1, K]\!]$.

Theorem 2. *Let $\mathsf{f}_t(i) \in [0, 1]$ for all t in $[\![0, T-1]\!]$ and $i \in [\![1, K]\!]$. Let $(\eta_t)_{t \in [\![0, T-1]\!]}$ is positive and non-increasing. EXP3 has a regret against $j \in [\![1, K]\!]$ of*

$$R_T(j) \le \frac{K}{2} \sum_{t=0}^{T-1} \eta_t + \frac{\ln K}{\eta_{T-1}}.$$

If EXP3 runs with $\eta_t = \sqrt{\frac{2 \ln K}{TK}}$ or $\eta_t = \sqrt{\frac{\ln K}{(t+1)K}}$, then, $\forall j \in [\![1, K]\!]$, we have $R_T(j) \le 2\sqrt{TK \ln K}$.

Algorithm 2 EXP3 (Auer et al., 2002)

1: Initialize $\mathsf{L}_0(i) \leftarrow 0$, $\forall i \in \{1, \dots, K\}$
2: **for** $t = 0, \dots, T-1$ **do**
3: Define the probability distribution $\mathsf{p}_t(i) = \frac{\exp(-\eta_t \mathsf{L}_t(i))}{\sum_{k=1}^{K} \exp(-\eta_t \mathsf{L}_t(k))}$
4: Draw an action I_t from the probability distribution p_t
5: For each action $i \in [\![1, K]\!]$, compute the losses $\mathsf{f}_t(i, \mathsf{I}_t) = \frac{\mathsf{f}_t(i)}{\mathsf{p}_t(i)} \mathbb{1}[\mathsf{I}_t = i]$
6: Update the cumulative losses $\mathsf{L}_{t+1}(i) = \mathsf{L}_t(i) + \mathsf{f}_t(i, \mathsf{I}_t)$

The value of the regret $R_T(j)$ depends on the chosen learning rates $\eta_t > 0$; fixing them appropriately yields an upper bound that depends only on T and K. When T is known, using $\eta_t = \sqrt{2 \ln K / TK}$ lets us improve the upper bound's constant (see the proofs in Appendix D); for simplicity, we nonetheless use the (larger) constant associated with $\eta_t = \sqrt{\ln K / (t+1)K}$.

4 Convergence analyses via regret

To analyze the convergence of an instantiation of Algorithm 1, the common strategy is to perform an analysis tailored to the specific algorithm. In this paper, we adopt another proof strategy, in which Theorem 3 is at its core: given an algorithm generating iterates $x_1, \dots, x_{T-1} \in \mathbb{R}^d$, we compare each point $x_t = x_{t-1} - \gamma_{t-1} \nabla f_{\xi_{t-1}}(x_{t-1})$ to a ghost point $\widetilde{x}_t = x_{t-1} - \widetilde{\gamma} \nabla f_{\xi_{t-1}}(x_{t-1})$ obtained from performing a single SGD step with the learning rate $\widetilde{\gamma}$. To prove Theorem 3, we first rely on the following growth-condition-style assumption linking the functions f and $f_{\xi'}$ (parametrized by a single ξ').

Assumption 7. *If Algorithm 1 is run for T iterations from $x_0 \in \mathbb{R}^d$, we have*

$$\mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[\sum_{t=0}^{T-2} f(x_{t+1}) - f(\widetilde{x}_{t+1}) \right] \leq \kappa \cdot \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[\sum_{t=0}^{T-2} f_{\xi'}(x_{t+1}) - f_{\xi'}(\widetilde{x}_{t+1}) \right] + \delta.$$

Put into words, the true gap $f(x_{t+1}) - f(\widetilde{x}_{t+1})$ does not diverge more than linearly with respect to its ξ'-estimated counterpart $f_{\xi'}(x_{t+1}) - f_{\xi'}(\widetilde{x}_{t+1})$; note that γ_t, and hence x_{t+1}, may itself depend on ξ', as γ_t is chosen based on $f_{\xi'}$ in Alice (see Section 5). We are now ready to state Theorem 3.

Theorem 3. *Suppose that Assumptions 1 to 3 and 7 hold. Let Algorithm 1 be run for T iterations from $x_0 \in \mathbb{R}^d$. Let also $\widetilde{x}_1, \dots, \widetilde{x}_T$ be the iterates generated by a single SGD step with the learning rate $\widetilde{\gamma}$ using Algorithm 1, i.e., we have $\widetilde{x}_t = x_{t-1} - \widetilde{\gamma} \nabla f_{\xi_{t-1}}(x_{t-1})$. Then, we have*

$$\mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[\sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \right] \leq \frac{1}{\widetilde{\gamma}} \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[f(x_0) - f(\widetilde{x}_T) \right]$$
$$+ \frac{\widetilde{\gamma} L}{2} \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[\sum_{t=0}^{T-1} \|\nabla f_{\xi_t}(x_t)\|^2 \right] + \frac{\kappa}{\widetilde{\gamma}} \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[\sum_{t=0}^{T-2} f_{\xi'}(x_{t+1}) - f_{\xi'}(\widetilde{x}_{t+1}) \right] + \frac{\delta}{\widetilde{\gamma}},$$

where $\mathbb{E}_{\boldsymbol{\gamma}_{T-1}}[\cdot]$ denotes the expectations $\mathbb{E}_{\gamma_0, \dots, \gamma_{T-1}}[\cdot]$.

Theorem 3 resembles Theorem 1, but there are key differences: *(i)* even though $\widetilde{\gamma}$ is not used in SGD, it appears directly in the bound instead of the learning rate γ_t; and *(ii)* we pay the price of an additional term in the bound (up to constants): $\mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}} \left[\sum_{t=0}^{T-2} f_{\xi'}(x_{t+1}) - f_{\xi'}(\widetilde{x}_{t+1}) \right]$, obtained by checking whether $f_{\xi'}(x_{t+1}) = f_{\xi'}(\widetilde{x}_{t+1})$, *i.e.*, whether Algorithm 1 matches SGD with $\widetilde{\gamma}$. This term is central to our approach: it can be interpreted as an adversarial-bandit regret[1], which we leverage to design a new optimization algorithm.

5 Alice: Adaptive learning-rate via iterative choice using EXP3

Alice adapts learning rates by sampling according to EXP3's probability distribution p_t: at each step t, we sample $\mathsf{I}_t \sim \mathsf{p}_t$, evaluate a set of candidate learning rates $\Gamma_t(\mathsf{I}_t)$ on a fixed reference function

[1] We are not the first to establish a link between bandit methods and hyper-parameter tuning; see Appendix A.4.

$f_{\xi'}$, and keep the minimizer γ_t, incurring the loss $\mathsf{f}'_t(\mathsf{I}_t)$; EXP3 also computes the loss $\mathsf{f}_t(i)$ that every $\gamma(i) \in \Gamma_t(\mathsf{I}_t)$ would have incurred on $f_{\xi'}$, updating its estimate for all such i, not only I_t.

5.1 Adversarial bandit framework for Alice

We relate actions to learning rates, define γ_t, use it to define $\mathsf{f}_t(i)$ and $\mathsf{f}'_t(\mathsf{I}_t)$, and recover the bandit regret upper-bounding the additional term $\mathbb{E}_{\xi',\boldsymbol{\xi}_{T-1},\boldsymbol{\gamma}_{T-1}}\left[\sum_{t=0}^{T-2} f_{\xi'}(x_{t+1}) - f_{\xi'}(\widetilde{x}_{t+1})\right]$ appearing in Theorem 3.

5.1.1 Grid of learning rates

The user only needs to define an interval of learning rates, *i.e.*, $\gamma_{\min}$ and $\gamma_{\max}$, from which we define a set of K learning rates (corresponding to the K actions). More precisely, we construct a geometric grid of $K = 1 + \left\lceil \log_\rho(\gamma_{\max}) \right\rceil - \left\lfloor \log_\rho(\gamma_{\min}) \right\rfloor$ rates between $\underline{\gamma} = \gamma(1)$ and $\overline{\gamma} = \gamma(K)$, with base $\rho > 1$:

$$\forall i \in [\![1, K]\!], \quad \gamma(i) = \rho^{\left\lfloor \log_\rho(\gamma_{\min}) \right\rfloor + (i-1)} = \rho^{\left\lceil \log_\rho(\gamma_{\max}) \right\rceil - (K-i)}.$$

The base ρ is a hyperparameter, which we propose to set with one of two techniques: *(i)* a common choice such as $\rho = 10$, or *(ii)* $\rho = \widehat{\rho}$, minimizing the term $\rho\sqrt{K \ln K}$ appearing in the bound of Corollary 1, *i.e.*, we have

$$\widehat{\rho} = \operatorname*{argmin}_{\rho>1} \left\{ \rho \sqrt{\left(1 + \left\lceil \log_\rho(\gamma_{\max}) \right\rceil - \left\lfloor \log_\rho(\gamma_{\min}) \right\rfloor\right) \ln\left(1 + \left\lceil \log_\rho(\gamma_{\max}) \right\rceil - \left\lfloor \log_\rho(\gamma_{\min}) \right\rfloor\right)} \right\}. \quad (1)$$

For Alice, we consider that $\rho = \widehat{\rho}$ is the default value (see Section 7 and Appendix E).

5.1.2 Learning rate γ_t

A natural choice for the sampled action $\mathsf{I}_t \sim \mathsf{p}_t$ would be $\gamma(\mathsf{I}_t)$ directly; however, our proof allows the learning rate used at round t to differ from $\gamma(i)$, as long as it lies in a set $\Gamma_t(i)$ containing $\gamma(i)$ for every action i, *i.e.*, $\{\gamma(i)\} \subseteq \Gamma_t(i)$. For every action $i \in [\![1, K]\!]$, we define, with an abuse of notation, $\gamma_t(i)$ as the element of $\Gamma_t(i)$ minimizing $f_{\xi'}(x_t - \gamma \nabla f_{\xi_t}(x_t))$:

$$\gamma_t(i) = \operatorname{argmin}_{\gamma \in \Gamma_t(i)} \left\{ f_{\xi'}(x_t - \gamma \nabla f_{\xi_t}(x_t)) \right\}, \qquad \text{where} \qquad \gamma_t := \gamma_t(\mathsf{I}_t). \quad (2)$$

Note that γ_t depends on ξ', and so does the iterate x_{t+1} it produces, which is why we have expectations over ξ' in Theorem 3 and Assumption 7. We instantiate $\Gamma_t(\mathsf{I}_t)$ with four sets, $\Gamma_t^{\mathsf{P}}(i)$, $\Gamma_t^{\mathsf{PS}}(i)$, $\Gamma_t^{\mathsf{US}}(i)$, and $\Gamma_t^{\mathsf{MS}}(i)$; three rely on a safe operator, adding a learning rate 0 to a set $\Gamma'_t(i)$ whenever every rate in it increases $f_{\xi'}$ beyond $f_{\xi'}(x_0)$, *i.e.*, we have

$$[\Gamma'_t(i)]^{\mathsf{S}} = \begin{cases} \Gamma'_t(i) \cup \{0\} & \text{if } \forall \gamma \in \Gamma'_t(i),\ f_{\xi'}(x_t - \gamma \nabla f_{\xi_t}(x_t)) > f_{\xi'}(x_0) \\ \Gamma'_t(i) & \text{otherwise} \end{cases}.$$

This safe operator matters in theory and practice (see further Proposition 1); only $\Gamma_t^{\mathsf{P}}(i)$ does not rely on it. $\Gamma_t^{\mathsf{P}}(i)$ and its safe counterpart $\Gamma_t^{\mathsf{PS}}(i)$ are defined as

$$\Gamma_t^{\mathsf{P}}(i) = \{\gamma(i)\} \quad \text{and} \quad \Gamma_t^{\mathsf{PS}}(i) = \left[\Gamma_t^{\mathsf{P}}(i)\right]^{\mathsf{S}}.$$

Put into words, the plain strategy always commits to $\gamma(i)$, while its safe counterpart also allows $\gamma = 0$ if $\gamma(i)$ increases the loss beyond $f_{\xi'}(x_0)$; Γ^{P} is included only for comparison, as it is not guaranteed to converge without the safe operator. The uniform and max strategies instead pair $\gamma(i)$ with another learning rate $\gamma(\mathsf{J}_t)$, where $\mathsf{J}_t \in [\![1, K]\!]$ does not depend on i and is redrawn every round t:

$$\Gamma_t^{\mathsf{US}}(i) = \left[\Gamma_t^{\mathsf{U}}(i)\right]^{\mathsf{S}} \quad \text{with} \quad \Gamma_t^{\mathsf{U}}(i) = \begin{cases} \{\gamma(i), \gamma(\mathsf{J}_t)\} & \text{if } i \neq \mathsf{J}_t \\ \{\gamma(i)\} & \text{otherwise} \end{cases}, \quad \text{where} \quad \mathsf{J}_t \sim \mathcal{U}([\![1, K]\!]), \text{ and}$$

$$\Gamma_t^{\mathsf{MS}}(i) = \left[\Gamma_t^{\mathsf{M}}(i)\right]^{\mathsf{S}} \quad \text{with} \quad \Gamma_t^{\mathsf{M}}(i) = \begin{cases} \{\gamma(i), \gamma(\mathsf{J}_t)\} & \text{if } i \neq \mathsf{J}_t \\ \{\gamma(i)\} & \text{otherwise} \end{cases}, \text{ where } \mathsf{J}_t = \max\{j \mid \mathsf{p}_t(j) \geq \tau_t\}.$$

Intuitively, for the uniform strategy, J_t is drawn independently of p_t, so every learning rate is equally likely to be paired with $\gamma(i)$. The max strategy instead selects J_t based on the current distribution p_t: using the vanishing threshold $\tau_t = \min(1/t+1, 1/K)$ (or $\min(1/T, 1/K)$ if the horizon T is known), J_t is the largest-index learning rate that EXP3 has not yet ruled out. As Section 7 shows, $\Gamma_t^{\mathsf{PS}}(i)$ offers the best trade-off, so we recommend Alice^{PS} in practice.

5.1.3 Losses $\mathsf{f}_t(i)$ and $\mathsf{f}'_t(\mathsf{l}_t)$

In order to define our bandit framework, we have to define the loss $\mathsf{f}_t(i)$ for each action $i \in [\![1, K]\!]$ for all $t \in [\![0, T-1]\!]$, using the learning rate $\gamma(i)$. Let $\widetilde{i} \in [\![1, K]\!]$ denote the action corresponding to the learning rate $\widetilde{\gamma}$ used in Theorem 3, *i.e.*, $\gamma(\widetilde{i}) = \widetilde{\gamma}$. We define

$$\mathsf{f}_t(i) = f_{\xi'}(x_t - \gamma(i)\nabla f_{\xi_t}(x_t))\,, \qquad \text{where} \qquad \mathsf{f}_t(\widetilde{i}) = f_{\xi'}(x_t - \widetilde{\gamma}\nabla f_{\xi_t}(x_t)) = f_{\xi'}(\widetilde{x}_{t+1}). \tag{3}$$

These quantities are the function values $f_{\xi'}$ obtained after a single stochastic gradient step with the learning rate $\gamma(i)$ (respectively $\widetilde{\gamma}$) defined for action i (respectively $\widetilde{i}$). We similarly define, for every action $i \in [\![1, K]\!]$, the loss $\mathsf{f}'_t(i)$ incurred under the learning rate $\gamma_t(i)$:

$$\mathsf{f}'_t(i) = f_{\xi'}(x_t - \gamma_t(i)\nabla f_{\xi_t}(x_t))\,, \qquad \text{where} \qquad \mathsf{f}'_t(\mathsf{l}_t) = f_{\xi'}(x_{t+1}). \tag{4}$$

Since $\gamma(i) \in \Gamma_t(i)$ and $\gamma_t(i)$ minimizes over $\Gamma_t(i)$ (Equation (2)), we have $\mathsf{f}'_t(i) \le \mathsf{f}_t(i)$ for every $i \in [\![1, K]\!]$, *i.e.*, Assumption 6 holds. At first sight, it may seem weird that a single ξ' is used to evaluate the losses $\mathsf{f}_t(i)$, instead of resampling a new random variable at each round t. This choice is nevertheless crucial: the term B_T appearing in Theorem 4 and Corollary 1 and upper-bounded in Proposition 1 relies on comparing $f_{\xi'}(x_t)$ to $f_{\xi'}(x_0)$ for the same ξ' at every round t. Resampling ξ' at each round would break this comparison, and we would no longer be able to guarantee that B_T stays bounded, and thus that the bound of Corollary 1 converges.

5.1.4 Regret

We are ready to relate the additional term appearing in Theorem 3 to the regret of a bandit algorithm. Using the definition of γ_t in Equation (2) and the loss definitions in Equations (3) and (4), we obtain

$$\mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}}\left[\sum_{t=0}^{T-2} f_{\xi'}(x_{t+1}) - f_{\xi'}(\widetilde{x}_{t+1})\right] = \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \mathbf{l}_{T-1}}\left[\sum_{t=0}^{T-2} \mathsf{f}'_t(\mathsf{l}_t) - \mathsf{f}_t(\widetilde{i})\right] = \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}}\left[R'_{T-1}(\widetilde{i})\right]. \tag{5}$$

5.2 EXP3 for unbounded losses with side observations

Alice uses EXP3 as an adversarial bandit algorithm, but the original EXP3 algorithm (Algorithm 2) requires losses in the bounded interval $[0, 1]$. In our setting, the loss function may instead be unbounded while remaining non-negative, *i.e.*, $\mathsf{f}_t(i), \mathsf{f}'_t(i) \in \mathbb{R}_+$ for all $t \in [\![0, T-1]\!]$ and $i \in [\![1, K]\!]$. We thus adapt EXP3 in Algorithm 3 to handle losses in $\mathbb{R}_+$, introducing a normalization b_t that is updated when a new loss $\mathsf{f}'_t(\mathsf{l}_t)$ is encountered. Our approach differs from prior work on unbounded-loss bandits as it extends Algorithm 2 via a simple normalization, recovering the bounded-loss setting when $\mathsf{b}_0 = 1$, without solving optimization problems or adding extra exploration; see Appendix A.2. Moreover, Algorithm 3 exploits side observations (Alon et al., 2013): playing an action $j \in [\![1, K]\!]$ at round t reveals the loss $\mathsf{f}_t(i)$ of every action i in a side-observation set $\mathsf{S}_{j,t} \subseteq [\![1, K]\!]$. Therefore, sampling l_t lets us update $\mathsf{L}_{t+1}(i)$ for every $i \in \mathsf{S}_{\mathsf{l}_t, t}$ via the importance weight $\mathsf{q}_t(i)$.

5.3 The Alice algorithm

We are now ready to present Alice (Algorithm 4), which adapts the learning rate γ_t thanks to EXP3 for unbounded losses with side observations (Algorithm 3). As for EXP3 (Algorithms 2 and 3), Alice maintains a probability distribution p_t over the actions $[\![1, K]\!]$ (Line 5): we sample $\mathsf{l}_t \sim \mathsf{p}_t$ (Line 7), choose the set $\Gamma_t(\mathsf{l}_t)$ and define γ_t (Line 8), update the iterate to x_{t+1} (Line 9), and compute the loss $\mathsf{f}'_t(\mathsf{l}_t)$ actually incurred (Line 10) to update the normalization b_{t+1} (Line 11); the losses $\mathsf{f}_t(i, \mathsf{l}_t)$ (Line 12), obtained via side observations, are used to update the cumulative losses $\mathsf{L}_{t+1}(i)$ (Line 13). Although the loop in Algorithm 4 is written for $t = 0, \dots, T-1$, Alice does not need T to be fixed in advance: the loop can simply keep running, using by default $\eta_t = \sqrt{\ln K / (t+1)K}$, which does not depend on T (see Section 7 and Appendix E).

Algorithm 3 EXP3 for unbounded losses with side observations

Initialize $\mathsf{L}_0(i) \leftarrow 0, \forall i \in \{1, \dots, K\}$
Initialize $\mathsf{b}_0 \geq 0$
for $t = 0, \dots, T-1$ **do**
  Define the probability distribution $\mathsf{p}_t(i) = \frac{\exp(-\frac{\eta_t}{\mathsf{b}_t}\mathsf{L}_t(i))}{\sum_{k=1}^{K}\exp(-\frac{\eta_t}{\mathsf{b}_t}\mathsf{L}_t(k))}$
  For each action $i \in [\![1, K]\!]$ compute $\mathsf{q}_t(i) = \mathbb{E}_{j\sim\mathsf{p}_t}\, \mathbb{1}[i \in \mathsf{S}_{j,t}]$
  Draw an action I_t from the probability distribution p_t
  Compute the normalization $\mathsf{b}_{t+1} = \max(\mathsf{b}_t,\ \mathsf{f}'_t(\mathsf{I}_t))$
  For each action $i \in [\![1, K]\!]$ compute the losses $\mathsf{f}_t(i, \mathsf{I}_t) = \frac{\mathsf{f}_t(i)}{\mathsf{q}_t(i)}\mathbb{1}\left[i \in \mathsf{S}_{\mathsf{I}_t,t}\right]$
  Update the cumulative losses $\mathsf{L}_{t+1}(i)$=$\mathsf{L}_t(i)$+$\mathsf{f}_t(i, \mathsf{I}_t)$

Algorithm 4 Alice

Initialize $\mathsf{L}_0(i) \leftarrow 0, \forall i \in \{1, \dots, K\}$
Initialize $x_0 \in \mathbb{R}^d$ and draw ξ'
Initialize $\mathsf{b}_0 = f_{\xi'}(x_0)$
for $t = 0, \dots, T-1$ **do**
  Define the probability distribution $\mathsf{p}_t(i) = \frac{\exp(-\frac{\eta_t}{\mathsf{b}_t}\mathsf{L}_t(i))}{\sum_{k=1}^{K}\exp(-\frac{\eta_t}{\mathsf{b}_t}\mathsf{L}_t(k))}$
  For each action $i \in [\![1, K]\!]$ compute $\mathsf{q}_t(i) = \mathbb{E}_{j\sim\mathsf{p}_t}\, \mathbb{1}[\gamma(i) \in \Gamma_t(j)]$
  Draw an action I_t from the probability distribution p_t
  Compute $\gamma_t = \operatorname{argmin}_{\gamma\in\Gamma_t(\mathsf{I}_t)}\{f_{\xi'}(x_t - \gamma\nabla f_{\xi_t}(x_t))\}$
  Compute $x_{t+1} \leftarrow x_t - \gamma_t\nabla f_{\xi_t}(x_t)$
  Compute $\mathsf{f}_t(\mathsf{I}_t) = f_{\xi'}(x_t - \gamma(\mathsf{I}_t)\nabla f_{\xi_t}(x_t))$ and $\mathsf{f}'_t(\mathsf{I}_t) = f_{\xi'}(x_{t+1})$
  Compute the normalization $\mathsf{b}_{t+1} = \max(\mathsf{b}_t,\ \mathsf{f}'_t(\mathsf{I}_t))$
  For each action $i \in [\![1, K]\!]$ compute the losses $\mathsf{f}_t(i, \mathsf{I}_t) = \frac{\mathsf{f}_t(i)}{\mathsf{q}_t(i)}\mathbb{1}\left[\gamma(i) \in \Gamma_t(\mathsf{I}_t)\right]$
  Update the cumulative losses $\mathsf{L}_{t+1}(i)$=$\mathsf{L}_t(i)$+$\mathsf{f}_t(i, \mathsf{I}_t)$

6 Convergence analyses of Alice

We analyze the convergence of Algorithm 4 by first upper-bounding the regret $R'_T(j)$ of Algorithm 3.

6.1 Regret bound of EXP3 for unbounded losses with side observations

To prove the regret of Algorithm 3 in Theorem 4, we first establish Lemma 1.

Lemma 1. *Let $\mathsf{f}_t(i), \mathsf{f}'_t(i) \in \mathbb{R}_+$ for all $t \in [\![0, T-1]\!]$ and $i \in [\![1, K]\!]$ with Assumption 6. Let $(\eta_t)_{t\in[\![0,T-1]\!]}$ is positive and non-increasing. Algorithm 3 has a regret against $j \in [\![1, K]\!]$ of*

$$R'_T(j) \leq \frac{K}{2}\sum_{t=0}^{T-1}\left(\eta_t \cdot \mathbb{E}_{\mathsf{I}_{t-1}}\left[\frac{\|\mathsf{f}'_t\|_\infty^2}{\mathsf{b}_t}\right]\right) + \frac{\mathbb{E}_{\mathsf{I}_{T-2}}\left[\mathsf{b}_{T-1}\right]\ln K}{\eta_{T-1}},$$

where $\|\mathsf{f}'_t\|_\infty = \max_{i\in[\![1,K]\!]}\mathsf{f}'_t(i)$ and $\mathsf{b}_t = \max(\mathsf{b}_{t-1}, \mathsf{f}'_t(\mathsf{I}_t))$.

The bound of Lemma 1 matches that of Theorem 2 up to two extra terms, both due to the normalization b_t: $\mathbb{E}_{\mathsf{I}_{t-1}}\left[\|\mathsf{f}'_t\|_\infty^2/\mathsf{b}_t\right]$ and $\mathbb{E}_{\mathsf{I}_{T-2}}\left[\mathsf{b}_{T-1}\right]$. Using the definition of $\mathsf{S}_{\mathsf{I}_t,t}$, the constant K in the bound could be refined (see Appendix A.3), but we keep it as in Theorem 2 since the gain is negligible.

Theorem 4. *Let $\mathsf{f}_t(i), \mathsf{f}'_t(i) \in \mathbb{R}_+$ for all $t \in [\![0, T-1]\!]$ and $i \in [\![1, K]\!]$ with Assumption 6. Algorithm 3, run with $\eta_t = \sqrt{\frac{2\ln K}{TK}}$ or $\eta_t = \sqrt{\frac{\ln K}{(t+1)K}}$, satisfies*

$$R'_T(j) \leq \mathsf{B}_T\sqrt{TK\ln K}, \quad \textit{where} \quad \mathsf{B}_T = \left[\max_{t\in[\![0,T-1]\!]}\left[\mathbb{E}_{\mathsf{I}_{t-1}}\left[\frac{\|\mathsf{f}'_t\|_\infty^2}{\mathsf{b}_t}\right]\right] + \mathbb{E}_{\mathsf{I}_{T-2}}\left[\mathsf{b}_{T-1}\right]\right].$$

Plugging the same η_t as Theorem 2 into Lemma 1, we recover its $\sqrt{TK\ln K}$ rate up to the constant B_T that every adversarial bandit algorithm for unbounded losses incurs; see Appendix A.2.

6.2 Convergence guarantees of Alice

We are ready to analyze the convergence of Alice (Algorithm 4).

Corollary 1. ***(i)*** *Suppose that Assumptions 1 to 3, 5 and 7 hold. Moreover, suppose that* $\eta_t = \sqrt{\frac{2\ln K}{TK}}$ *or* $\eta_t = \sqrt{\frac{\ln K}{(t+1)K}}$ *and* $\frac{1}{L\alpha} \in [\underline{\gamma}, \overline{\gamma}]$*, then, we have*

$$\mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}}\left[\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla f(x_t)\|^2\right] \leq 2L\alpha\rho \cdot \left(\frac{f(x_0)}{T} + \kappa \cdot \mathsf{B}_T\sqrt{\frac{K\ln K}{T}} + \frac{\delta}{T}\right) + \frac{\beta}{\alpha T}.$$

(ii) *Suppose that Assumptions 1 to 5 and 7 hold. Moreover, suppose that* $\eta_t = \sqrt{\frac{2\ln K}{TK}}$ *or* $\eta_t = \sqrt{\frac{\ln K}{(t+1)K}}$ *and* $\sqrt{\frac{2\rho}{L(T\alpha M+\beta)}\left[f(x_0) + \kappa \cdot \mathsf{B}_T\sqrt{TK\ln K} + \delta\right]} \in [\underline{\gamma}, \overline{\gamma}]$*, then, we have*

$$\mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}, \boldsymbol{\gamma}_{T-1}}\left[\frac{1}{T}\sum_{t=0}^{T-1}\|\nabla f(x_t)\|^2\right] \leq \frac{\sqrt{2L\rho\left(T\alpha M+\beta\right)\left(f(x_0) + \kappa \cdot \mathsf{B}_T\sqrt{TK\ln K} + \delta\right)}}{T},$$

where the term B_T *is defined by* $\mathsf{B}_T = \mathbb{E}_{\xi', \boldsymbol{\xi}_{T-1}}\left[\max_{t\in[\![0,T-1]\!]}\left[\mathbb{E}_{\mathsf{I}_{t-1}}\left[\frac{\|\mathsf{f}'_t\|^2_\infty}{\mathsf{b}_t}\right]\right] + \mathbb{E}_{\mathsf{I}_{T-2}}\left[\mathsf{b}_{T-1}\right]\right]$*, with* $\|\mathsf{f}'_t\|_\infty = \max_{i\in[\![1,K]\!]} f_{\xi'}(x_t - \gamma_t(i)\nabla f_{\xi_t}(x_t))$ *and* $\mathsf{b}_t = \max(\mathsf{b}_{t-1}, f_{\xi'}(x_{t-1} - \gamma_{t-1}\nabla f_{\xi_{t-1}}(x_{t-1}))) = \max_{t'\in[\![0,t-1]\!]} f_{\xi'}\big(x_{t'} - \gamma_{t'}\nabla f_{\xi_{t'}}(x_{t'})\big)$*.*

In words, the convergence guarantee combines two bounds, each holding under different assumptions. Corollary 1 *(i)* converges at a rate of $\mathcal{O}(\sqrt{1/T})$. Corollary 1 *(ii)* instead guarantees convergence at the slower rate $\mathcal{O}(1/\sqrt{\sqrt{T}})$, and when all assumptions hold, the rate is the minimum of the two bounds. Neither bound depends on the selected learning rates (unlike Theorem 1) so Alice is a parameter-free optimization algorithm (see Appendix A.5). Note that choosing the best learning rate in a set has also been explored in the AutoSGD algorithm (Surjanovic et al., 2025); see Appendix A.6 for details. Moreover, note that the term B_T is at first sight not bounded. Fortunately, we show that B_T is bounded by a constant in Proposition 1 when we use the safe operator, *i.e.*, when considering for instance the sets Γ_t^{PS}, Γ_t^{US}, and Γ_t^{MS}.

Proposition 1. *Suppose that Assumption 1 holds and that, for all* $t \in [\![0, T-1]\!]$ *and* $i \in [\![1, K]\!]$*, there exists* $\Gamma'_t(i)$ *such that* $\Gamma_t(i) = \left[\Gamma'_t(i)\right]^{\mathcal{S}}$*, i.e., the safe operator is applied. Taking* $\mathsf{b}_t = \max(\mathsf{b}_{t-1}, \mathsf{f}'_t(\mathsf{I}_t))$ *with* $\mathsf{b}_0 = f_{\xi'}(x_0)$*, we have* $\mathsf{B}_T \leq \mathbb{E}_{\xi'}\left[2f_{\xi'}(x_0)\right] = 2f(x_0)$*.*

7 Experiments

This section illustrates the usefulness of Alice (Algorithm 4) by comparing it to state-of-the-art optimization algorithms. We detail the experimental setting, and we present some results.

Algorithms and hyperparameters. We compare Alice with SGD, Adam (Kingma & Ba, 2015), AutoSGD (Surjanovic et al., 2025), and DoWG (Khaled et al., 2023). These are state-of-the-art within their family: AutoSGD for restarted/tuned SGD (Khaled & Jin, 2024; Attia & Koren, 2024), Adam for the AdaGrad family (Duchi et al., 2011; Ward et al., 2019), and DoWG for parameter-free methods (*e.g.*, DoG (Ivgi et al., 2023), D-Adaptation (Defazio & Mishchenko, 2023)). All methods are initialized with a base learning rate $\gamma_0 \in \{10^{-6}, 10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^0, 10^1, 10^2\}$. For SGD, γ_0 corresponds to the learning rate, while for Adam, DoWG, and AutoSGD it denotes the initial learning rate (for DoWG, $\gamma_0 = r_\varepsilon$). For Adam, we use the default parameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$, and for AutoSGD we set $c = 1/2$ and $C = 2$. Regarding Alice, we denote by Alice^{P}, Alice^{PS}, Alice^{US}, and Alice^{MS} the variants that respectively use the plain set Γ_t^{P} and the safe sets Γ_t^{PS}, Γ_t^{US}, and Γ_t^{MS}. For Alice, the base learning rate satisfies $\gamma_0 = \overline{\gamma}$, which varies for Alice^{P}, while for Alice^{PS}, Alice^{US}, and Alice^{MS} we fix $\overline{\gamma} = 10^2$ and $\underline{\gamma} = 10^{-6}$. We run all experiments for $T = 50{,}000$ iterations and use mini-batches $\mathcal{B}$ of size $|\mathcal{B}| = 256$. Unless stated otherwise (notably in Appendix E), we assume that the horizon is unknown; accordingly, we use $\eta_t = \sqrt{\ln K/(t+1)K}$ and the parameter $\widehat{\rho}$ introduced in Equation (1).

Tasks.[2] We compare the algorithms on a diverse range of machine learning tasks. First, we consider (convex) linear regression tasks on the datasets `abalone`, `cpusmall`, and `housing`. Next, we

[2] More details on the tasks are given in Appendix E.

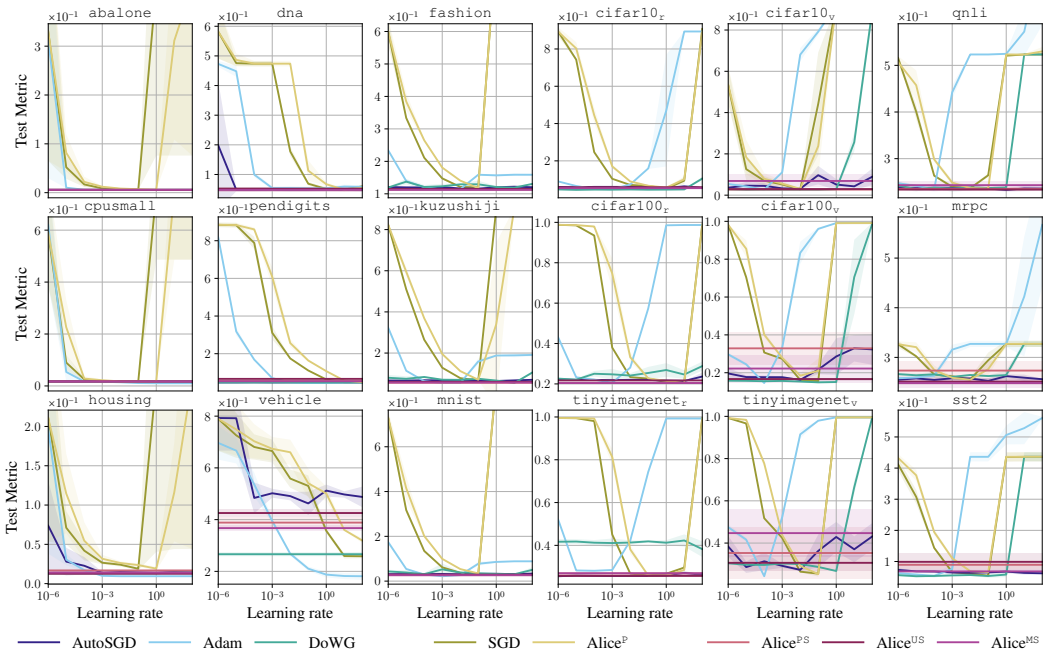

Figure 1: Plots of the best test metric achieved during training *w.r.t.* the learning rate γ_0, for each task and each algorithm. The lines show the mean over seeds, and the shaded areas show the standard deviation. Curves for Alice^{PS}, Alice^{US}, and Alice^{MS} are constant as $\overline{\gamma}$=10^2 and $\underline{\gamma}$=10^{-6}.

consider (convex) logistic regression tasks on `dna`, `pendigits`, and `vehicle`. We then consider a fully connected neural network with two hidden layers of 100 units each for image classification on the datasets `mnist`, `fashion`, and `kuzushiji`. We further consider more challenging image classification tasks on the datasets `cifar10`, `cifar100`, and `tinyimagenet`, using either a ResNet-18 or a Vision Transformer (ViT). We denote by $\texttt{cifar10}_\texttt{r}$, $\texttt{cifar100}_\texttt{r}$, and $\texttt{tinyimagenet}_\texttt{r}$ the tasks using ResNet, and by $\texttt{cifar10}_\texttt{v}$, $\texttt{cifar100}_\texttt{v}$, and $\texttt{tinyimagenet}_\texttt{v}$ the tasks using ViT. Finally, we consider text classification tasks using the BERT-Tiny variant from the BERT Miniatures collection on the datasets `qnli`, `mrpc`, and `sst2`.

Results. We present an overview of the experimental results in Figure 1. For each algorithm, we report the best test risk (where ℓ is the $0/1$ loss or the mean squared error) achieved during training, as a function of γ_0 and averaged over three runs. We use a train/test split to evaluate the optimizers, as also done by, *e.g.*, DoWG (Khaled et al., 2023), since the models are learned with the training set, the test set is never seen during training, avoiding classical overfitting on it. Overall, Alice^{P} behaves similarly to SGD, while Alice^{PS}, Alice^{US}, and Alice^{MS} are comparable to at least one tuned baseline despite requiring no hyperparameter tuning, which Appendix E confirms. Appendix E also shows that Alice adapts γ_t to well-performing learning rates, that the safe mechanism is mostly needed early in training, that ρ or the horizon T barely affect performance, and that Alice^{P} and Alice^{PS} have a time overhead comparable to SGD, Adam, and DoWG while Alice^{US}, Alice^{MS}, and AutoSGD are much slower. Overall, Alice^{PS} (with T unknown, $\rho = \widehat{\rho}$) is our recommended variant of Alice.

8 Conclusion

In this work, we introduced Alice, an optimization algorithm that dynamically selects the learning rate using the adversarial bandit framework, specifically EXP3. We proved convergence bounds for this algorithm and extended the regret analysis of EXP3 to unbounded losses with side observations. Among its variants, Alice^{PS} achieves the best overall trade-off in practice. This work opens the way for future research on optimization algorithms based on multi-armed bandits.

AI USE STATEMENT

We did not use generative AI tools to develop the paper's theoretical results (assumptions, statements, and proofs), nor to design the experiments or interpret their results. We used AI tools to rephrase sentences, draft a first version of some paragraphs, identify bugs in our code, and generate code to produce the figures in this paper. We treated AI assistance as an iterative collaboration: every AI-suggested paragraph and every piece of AI-generated or AI-modified code was checked by the authors, and revised or rewritten whenever needed. We take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

Concerning the theoretical results, all the assumptions are described in the main text of the paper, while the complete proofs are given in Appendices B to D. Appendix A provides additional remarks and discussion. Regarding the experiments, the setting is fully described in Section 7 and Appendix E. We will publicly release the source code, including the implementation of Alice and all the baselines, as well as the scripts used to reproduce the experiments and generate the figures in this paper.

REFERENCES

Noga Alon, Nicolò Cesa-Bianchi, Claudio Gentile, and Yishay Mansour. From Bandits to Experts: A Tale of Domination and Independence. In *Conference on Neural Information Processing Systems*, 2013.

Amit Attia and Tomer Koren. How Free is Parameter-Free Stochastic Optimization? In *International Conference on Machine Learning*, 2024.

Peter Auer, Nicolò Cesa-Bianchi, Yoav Freund, and Robert E. Schapire. The Nonstochastic Multi-armed Bandit Problem. *Society for Industrial and Applied Mathematics Journal on Computing*, 2002.

Augustin Cauchy. Méthode générale pour la résolution des systèmes d'équations simultanées. *Compte Rendu de l'Académie des Sciences de Paris*, 1847.

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

Mingyu Chen and Xuezhou Zhang. Improved Algorithms for Adversarial Bandits with Unbounded Losses. In *ICML Workshop: Aligning Reinforcement Learning Experimentalists and Theorists*, 2024.

Tarin Clanuwat, Mikel Bober-Irizar, Asanobu Kitamoto, Alex Lamb, Kazuaki Yamamoto, and David Ha. Deep Learning for Classical Japanese Literature. *arXiv*, 2018.

Aaron Defazio and Konstantin Mishchenko. Learning-Rate-Free Learning by D-Adaptation. In *International Conference on Machine Learning*, 2023.

Henrique Donâncio, Antoine Barrier, Leah F. South, and Florence Forbes. Dynamic Learning Rate for Deep Reinforcement Learning: A Bandit Approach. *arXiv*, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, 2021.

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

Matthias Feurer, Jan N. van Rijn, Arlind Kadra, Pieter Gijsbers, Neeratyoy Mallik, Sahithya Ravi, Andreas Müller, Joaquin Vanschoren, and Frank Hutter. OpenML-Python: an extensible Python API for OpenML. *Journal of Machine Learning Research*, 2021.

Guillaume Garrigos and Robert M. Gower. Handbook of Convergence Theorems for (Stochastic) Gradient Methods. *arXiv*, 2024.

Saeed Ghadimi and Guanghui Lan. Stochastic First- and Zeroth-Order Methods for Nonconvex Stochastic Programming. *SIAM Journal on Optimization*, 2013.

Hédi Hadiji and Gilles Stoltz. Adaptation to the Range in K-Armed Bandits. *Journal of Machine Learning Research*, 2023.

A. S. Syed Shahul Hameed and Narendran Rajagopalan. MABSearch: The Bandit Way of Learning the Learning Rate–A Harmony Between Reinforcement Learning and Gradient Descent. *National Academy Science Letters*, 2024.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *Conference on Computer Vision and Pattern Recognition*, 2016.

Maor Ivgi, Oliver Hinder, and Yair Carmon. Dog is sgd's best friend: A parameter-free dynamic step size schedule. In *International Conference on Machine Learning*, 2023.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic Best Arm Identification and Hyperparameter Optimization. In *International Conference on Artificial Intelligence and Statistics*, 2016.

Ahmed Khaled and Chi Jin. Tuning-Free Stochastic Optimization. In *International Conference on Machine Learning*, 2024.

Ahmed Khaled, Konstantin Mishchenko, and Chi Jin. DoWG Unleashed: An Efficient Universal Parameter-Free Gradient Descent Method. In *Advances in Neural Information Processing Systems*, 2023.

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

Alex Krizhevsky and Geoffrey Hinton. Learning Multiple Layers of Features from Tiny Images. Technical report, University of Toronto, 2009.

Yann LeCun, Corinna Cortes, and Christopher J.C. Burges. THE MNIST DATASET of handwritten digits, 1998.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization. *Journal of Machine Learning Research*, 2018.

Rui Liu, Tianyi Wu, and Barzan Mozafari. Adam with Bandit Sampling for Deep Learning. In *Advances in Neural Information Processing Systems*, 2020.

Sudeep Raja Putta and Shipra Agrawal. Scale-Free Adversarial Multi Armed Bandits. In *International Conference on Algorithmic Learning Theory*, 2022.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 1952.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015.

Mark Schmidt and Nicolas Le Roux. Fast Convergence of Stochastic Gradient Descent under a Strong Growth Condition. *arXiv*, 2013.

Evan Randall Sparks, Ameet Talwalkar, Michael J. Franklin, Michael I. Jordan, and Tim Kraska. TuPAQ: An Efficient Planner for Large-scale Predictive Analytic Queries. *arXiv*, 2015.

Nikola Surjanovic, Alexandre Bouchard-Côté, and Trevor Campbell. AutoSGD: Automatic Learning Rate Selection for Stochastic Gradient Descent. *arXiv*, 2025.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *Coursera: Neural networks for machine learning*, 2012.

Iulia Turc, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Well-Read Students Learn Better: On the Importance of Pre-training Compact Models. *arXiv*, 2019.

Sharan Vaswani, Francis R. Bach, and Mark Schmidt. Fast and Faster Convergence of SGD for Over-Parameterized Models and an Accelerated Perceptron. In Kamalika Chaudhuri and Masashi Sugiyama (eds.), *International Conference on Artificial Intelligence and Statistics*, 2019.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. In *International Conference on Learning Representations*, 2019.

Rachel A. Ward, Xiaoxia Wu, and Léon Bottou. AdaGrad stepsizes: sharp convergence over nonconvex landscapes. In *International Conference on Machine Learning*, 2019.

Ross Wightman. PyTorch Image Models. https://github.com/rwightman/pytorch-image-models, 2019.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-MNIST: a Novel Image Dataset for Benchmarking Machine Learning Algorithms. *arXiv*, 2017.