Bandit Linear Optimization for Sequential Decision Making and Extensive-Form Games

Gabriele Farina,[1] **Robin Schmucker,**[2] **Tuomas Sandholm**[1,2,3,4,5]

[1]Computer Science Department, Carnegie Mellon University, [2]Machine Learning Department, Carnegie Mellon University
[3]Strategic Machine, Inc., [4]Strategy Robot, Inc., [5]Optimized Markets, Inc.
gfarina@cs.cmu.edu, rschmuck@cs.cmu.edu, sandholm@cs.cmu.edu

Abstract

Tree-form sequential decision making (TFSDM) extends classical one-shot decision making by modeling tree-form interactions between an agent and a potentially adversarial environment. It captures the online decision-making problems that each player faces in an extensive-form game, as well as Markov decision processes and partially-observable Markov decision processes where the agent conditions on observed history. Over the past decade, there has been considerable effort into designing online optimization methods for TFSDM. Virtually all of that work has been in the *full-feedback* setting, where the agent has access to *counterfactuals*, that is, information on what *would have happened* had the agent chosen a different action at any decision node. Little is known about the *bandit* setting, where that assumption is reversed (no counterfactual information is available), despite this latter setting being well understood for almost 20 years in one-shot decision making. In this paper, we give the first algorithm for the bandit linear optimization problem for TFSDM that offers both (i) linear-time iterations (in the size of the decision tree) and (ii) $O(\sqrt{T})$ cumulative regret in expectation compared to any fixed strategy, at all times T. This is made possible by new results that we derive, which may have independent uses as well: 1) geometry of the dilated entropy regularizer, 2) autocorrelation matrix of the natural sampling scheme for sequence-form strategies, 3) construction of an unbiased estimator for linear losses for sequence-form strategies, and 4) a refined regret analysis for mirror descent when using the dilated entropy regularizer.

1 Introduction

Tree-form sequential decision making (TFSDM) models multi-stage online decision-making problems (Farina, Kroer, and Sandholm 2019). In TFSDM, an agent interacts sequentially with a potentially reactive environment in two ways: (i) selecting actions at decision points and (ii) partially observing the environment at observation points. Decision points and observation points alternate along a tree structure. TFSDM captures the online decision process that each player faces in an extensive-form game, as well as Markov decision processes and partially-observable Markov decision processes where the agent conditions on observed history. Regret minimization, one of the main mathematical abstractions in the field of online learning, has proved to be an extremely versatile tool for TFSDM. For instance, over the past decade, regret minimization algorithms such as counterfactual regret minimization (CFR) (Zinkevich et al. 2007) and related newer, faster algorithms have become popular for solving zero-sum games (Tammelin et al. 2015; Brown and Sandholm 2015; Brown, Kroer, and Sandholm 2017; Brown and Sandholm 2017a, 2019a). These newer algorithms served as an important component in the computational game-solving pipelines that achieved several recent milestones in computing superhuman strategies in two-player limit Texas hold'em (Bowling et al. 2015), two-player no-limit Texas hold'em (Brown and Sandholm 2017b,c) and multi-player no-limit Texas hold'em (Brown and Sandholm 2019b).

However, those methods rely on having access to *counterfactuals*, that is, information on what would have happened had the agent chosen a different action at any decision point. While this assumption is reasonable when regret minimization algorithms are used in self-play (for instance, as a way to converge to a Nash equilibrium in an extensive-form game), it limits their applicability in *online* decision-making settings, where the algorithm is deployed to learn strategies (for instance, exploitative strategies) against a non-stationary and potentially adversary opponent. In the *bandit* setting that assumption is reversed, and no counterfactual information is available to the online decision maker. Despite this latter setting being well understood for almost 20 years in *one-shot* decision making, surprisingly little is known about the bandit optimization setting in *sequential* decision making. Part of the reason for this gap is that the multi-stage nature of TFSDM poses challenges that are not present in one-shot decision making, such as 1) not knowing what the environment would have done in parts of the tree that were not reached (and not even knowing the current path of play if you do not observe the environment's actions), and 2) having an exponential number of available sequential policies to choose from.

In this paper we give the first algorithm for the well-established bandit linear optimization problem in TFSDM and show that it achieves $O(\sqrt{T})$ expected regret compared to any fixed strategy even when playing against a reactive environment, while at the same time only requiring a single linear time tree traversal per iteration. To our knowledge, there has been only one prior approach to bandit linear op-

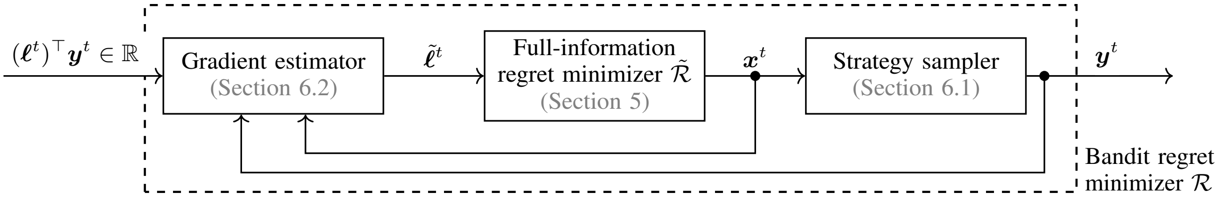

Figure 1: Overview of the construction of our bandit regret minimizer $\mathcal{R}$.

timization that offers both (i) iterations that are polynomial in the number of sequences in the decision process and (ii) $\tilde{O}(\sqrt{T})$ expected regret compared to any fixed strategy (Abernethy, Hazan, and Rakhlin 2008). That work was for general convex sets. By focusing on TFSDM, we achieve faster iterations and convergence in fewer iterations. Our algorithm runs in linear time per iteration unlike the prior algorithm which requires that an eigendecomposition of a Hessian matrix be computed at each iteration—a cubic-time operation. Our expected regret is $O(\sqrt{T})$ instead of the prior algorithm's $O(\sqrt{T \log T})$. One application of our algorithm and theory is to find exploitative strategies for an agent in extensive-form games against a non-adaptive opponent (that is, an opponent that cannot learn from our prior play in previous iterations of the extensive-form game) but one that can randomize and condition its actions on its observations of our play within any iteration of the extensive-form game. We provide experiments in this setting. To our knowledge, this is the first implementation of bandit optimization for TFSDM.

A known weakness of the approach of Abernethy, Hazan, and Rakhlin (2008), which is also a weakness in our approach, is that the bound on regret holds not with high probability but only in expectation. This weakness can be eliminated in theory if iterations are allowed to take exponential time in the number of sequences in the decision process (Bartlett et al. 2008; Hazan and Li 2016) or a recent manuscript suggests that it can be achieved by accepting slower $O(T^{2/3})$ convergence (Braun and Pokutta 2016). Another approach achieves $\tilde{O}(n^{9.5}\sqrt{T})$ regret, where n is the size of the input TFSDM problem, at the cost of having each iteration incur into a factor that grows proportionally to the time horizon T (Bubeck, Lee, and Eldan 2017). Due to this weakness, in our approach, the one of Abernethy, Hazan, and Rakhlin (2008) and other methods that do not enjoy a high-probability regret bound, when used in self play in two-player zero-sum games, the average regrets of the players might not converge to zero—but if they do, the average strategies converge to a Nash equilibrium. It is an open problem (except for relatively simple settings like simplex (Auer et al. 2002) and sphere (Abernethy and Rakhlin 2009)) whether in-high-probability $\tilde{O}(\sqrt{T})$ regret bounds can be obtained in the bandit setting in polynomial-time iterations. Abernethy and Rakhlin (2009) presented a template for deriving such bounds, but several pieces therein need to be instantiated to complete the proof of bounds. The theory of the present paper offers solutions for some of those pieces for general TFSDM problems, as we will discuss, so our paper may help pave the way to solving the open problem for TFSDM.

1.1 Overview of Our Approach

In this subsection we give an overview of the key ideas behind our method. We assume some basic familiarity with the concept of full-information and bandit regret minimizers; both concepts are recalled in Section 3.

The approach we follow in this paper combines several tools and insights. We construct a bandit regret minimizer $\mathcal{R}$ starting from a *full-information* regret minimizer $\tilde{\mathcal{R}}$, that is, one that has access to the full loss vector at each iteration. Our bandit regret minimizer $\mathcal{R}$ works as follows:

(i) the next strategy $\boldsymbol{y}^t$ for $\mathcal{R}$ is computed starting from the strategy $\boldsymbol{x}^t$ output by $\tilde{\mathcal{R}}$. We employ a specific unbiased *sampling scheme* to sample $\boldsymbol{y}^t$ from $\boldsymbol{x}^t$. At all times t, we guarantee that $\mathbb{E}[\boldsymbol{y}^t | \boldsymbol{y}^1, \dots, \boldsymbol{y}^{t-1}] = \boldsymbol{x}^t$;

(ii) each loss evaluation (that is, the negative of the reward of the strategy that we played in the most recent iteration) $(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t \in \mathbb{R}$ is used to constructs an artificial loss vector $\tilde{\boldsymbol{\ell}}^t$ in a specific way that makes it an unbiased estimator of $\boldsymbol{\ell}^t$. This artificial loss vector is then passed to $\tilde{\mathcal{R}}$.

The construction of $\mathcal{R}$ is summarized pictorially in Figure 1. We implement $\tilde{\mathcal{R}}$ using the *online mirror descent* algorithm paired with a type of regularizer called the *dilated entropy distance-generating function (DGF)*. The reasons behind this choice are twofold. First, it enables an efficient implementation of $\tilde{\mathcal{R}}$, since projections onto sequential strategy spaces based on the dilated entropy DGF amount to a (linear-time) traversal of the decision process. Second, it serves as the basis for defining a *local, time-dependent* norm $\|\cdot\|_t$ that combines well with the regret bound of online mirror descent. Two steps are critical in the proof of the regret bound for the overall regret minimizer $\mathcal{R}$. First, we show that, in expectation, $\|\tilde{\boldsymbol{\ell}}\|_{*,t}$ is upper bounded by a small (time-independent) constant c (the same property would not hold for a generic time-independent norm). This, combined with the local-norm regret bound mentioned above, can be used to show that the regret cumulated by $\tilde{\mathcal{R}}$ is $O(\sqrt{T})$ in expectation. Second, we use the unbiasedness of $\boldsymbol{y}^t$ and $\tilde{\boldsymbol{\ell}}^t$ to conclude that the expected regret accumulated by $\mathcal{R}$ matches that of $\tilde{\mathcal{R}}$.

1.2 Relationships to Related Research

The idea of constructing a bandit regret minimizer starting from a full-information regret minimizer was used in Abernethy and Rakhlin (2009). A general construction of an unbiased estimator $\tilde{\boldsymbol{\ell}}^t$ of $\boldsymbol{\ell}^t$ starting from the loss evaluation $(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t$ appears in Bartlett et al. (2008). We generalize their argument to handle strategy domains where the vector space spanned by all decision vectors is rank deficient (this is the case for sequential strategy spaces), and give several new, fundamental properties about the autocorrelation matrix of the standard sampling scheme for sequence-form strategies. The idea of using time-dependent norms to obtain a tighter regret analysis than time-independent norms appeared in, for example, Abernethy, Hazan, and Rakhlin (2008); Abernethy and Rakhlin (2009); Shalev-Shwartz (2012), while the use of the dilated entropy regularizer in the context of sequential decision making and extensive-form games for other purposes goes back to the original work by Hoda et al. (2010), with important newer practical observations by Kroer et al. (2020).

EXP3 (Auer et al. 2002) is credited to be the first bandit regret minimizer for simplex domains. GEOMETRICHEDGE (Dani, Kakade, and Hayes 2008) is a general-purpose bandit regret minimizer that can be applied to any set of decisions. However, it requires one to compute a barycentric spanner (Awerbuch and Kleinberg 2004), which in our setting would have prohibitive pre-processing cost. Furthermore, it runs in exponential time per iteration in the general case, and it is not known whether that can be avoided in our setting.

Lanctot et al. (2009) suggested as a side note that a specific *online* variant of their Monte Carlo CFR (MCCFR) algorithm (as opposed to the usual *self-play* MCCFR algorithm) could be used for online decision making without counterfactuals. Their paper did not provide theoretical guarantees for that online variant. The well-established bandit linear optimization setting considered in this paper is quite different from the one that online MCCFR implicitly operates on. First, in bandit optimization (our setting), each strategy is output *before* the environment reveals feedback, and the only feedback that the environment gives is a single real-valued reward $(\boldsymbol{\ell}^t)^\top \boldsymbol{x}^t$. In contrast, in online MCCFR the feedback is not just the final payoff, as online MCCFR needs to know which path was followed in the game tree and the terminal leaf, so that regrets can be updated for all decision nodes of the player on the path from the root to the leaf. So, even if a version of online MCCFR with theoretical guarantees were developed, it would *not* be an algorithm for bandit linear optimization, but rather an algorithm for a different (and easier, since more feedback is given to the decision maker) online learning setting. Depending on the applications, that setting—which, to our knowledge, has never been investigated nor formally proposed—might be more or less natural than bandit linear optimization. Since the bandit linear optimization model does not require that the decision maker observe the path of play, it can be used to model settings in which (i) there is no path in the game tree, because the loss given by the environment does not represent playing against an opponent; (ii) the decision maker does not interact immediately with the environment: the output strategy is evaluated at a later time by the environment and feedback is given only then; (iii) the environment does not inform the decision maker of the specific trajectory taken in the interaction out of privacy concerns; or any combination of the above.

2 Review of Sequential Decision Making

The decision process of an TFSDM problem is structured as a tree of decision points—in which an action must be selected by the agent—and observation points—in which the environment reveals a signal to the agent. We denote the set of decision points in the TFSDM problem as $\mathcal{J}$, and the set of observation points as $\mathcal{K}$. At each decision point $j \in \mathcal{J}$, the agent selects an action from the set A_j of available actions. At each observation point $k \in \mathcal{K}$, the agent observes a signal s_k from the environment out of a set of possible signals S_k. We denote by ρ the transition function of the process. Picking action $a \in A_j$ at decision point $j \in \mathcal{J}$ results in the process transitioning to $\rho(j, a) \in \mathcal{J} \cup \mathcal{K} \cup \{\diamond\}$, where $\diamond$ denotes the end of the decision process. Similarly, the process transitions to $\rho(k, s) \in \mathcal{J} \cup \mathcal{K} \cup \{\diamond\}$ after the agent observes signal $s \in S_k$ at observation point $k \in \mathcal{K}$. In line with the game theory literature, we call a pair (j, a) where $j \in \mathcal{J}$ and $a \in A_j$ a *sequence*. The set of all sequences is denoted as $\Sigma := \{(j, a) : j \in \mathcal{J}, a \in A_j\}$. For notational convenience, we will often denote an element (j, a) in Σ as ja without using parentheses. Given a sequence $ja \in \Sigma$, we denote by $\boldsymbol{u}_{ja}$ the vector such that $(\boldsymbol{u}_{ja})_{j'a'} = 1$ if the (unique) path from the root node to action a' at decision point j' passes through action a at decision point j, and $(\boldsymbol{u}_{ja})_{j'a'} = 0$ otherwise. Finally, given a node $v \in \mathcal{J} \cup \mathcal{K}$, we denote by p_v its *parent sequence*, defined as the last sequence (that is, decision point-action pair) encountered on the path from the root to v. If the agent does not act before v (that is, v is the root of the process or only observation points are encountered on the path from the root to v), we let $p_v = \varnothing$. We use the symbol N_v to denote the number of decision points in the subtree rooted at v. If v itself is a decision point, v is included in the count.

Strategies in TFSDM problems A strategy for an agent in an TFSDM problem specifies a distribution over the set of actions A_j at each decision point $j \in \mathcal{J}$. We represent a strategy using the *sequence-form representation*, that is, as a vector $\boldsymbol{x} \in \mathbb{R}^{|\Sigma|}_{\geq 0}$ whose entries are indexed by Σ. The entry x_{ja} contains the product of the probabilities of all actions at all decision points on the path from the root of the process to action a at decision point $j \in \mathcal{J}$. In order to be a valid sequence-form strategy, the entries in $\boldsymbol{x}$ must satisfy the following consistency constraints (Romanovskii 1962; Koller, Megiddo, and von Stengel 1994; von Stengel 1996):

$$\begin{aligned} \sum_{a \in A_j} x_{ja} &= x_{p_j} \quad \forall j \in \mathcal{J} \text{ s.t. } p_j \neq \varnothing, \\ \sum_{a \in A_j} x_{ja} &= 1 \quad \forall j \in \mathcal{J} \text{ s.t. } p_j = \varnothing. \end{aligned} \tag{1}$$

Since $\varnothing$ is not an element in Σ, there is no entry in $\boldsymbol{x}$ that corresponds to $\varnothing$, and the notation $x_\varnothing$ is invalid. We will slightly abuse notation and refer to $x_\varnothing$ to mean the constant value 1. Finally, we let $\Pi \subseteq \mathbb{R}^{|\Sigma|}_{\geq 0}$ be the finite set of all *pure* (also known as *deterministic*) sequence-form strategies, that is, strategies that assign probability 1 to exactly one action at each decision point. The set of all sequence-form strategies, denoted Q, is the convex hull $Q := \operatorname{co}\Pi$ of the set of pure strategies Π.

3 Regret Minimization

A regret minimizer is an abstraction for a repeated decision maker. The decision maker repeatedly interacts with an unknown (possibly adversarial) environment by choosing points $\boldsymbol{x}^1, \dots, \boldsymbol{x}^T$ from a set $\mathcal{X} \subseteq \mathbb{R}^n$ of feasible decisions and incurring a linear loss $(\boldsymbol{\ell}^1)^\top \boldsymbol{x}^1, \dots, (\boldsymbol{\ell}^T)^\top \boldsymbol{x}^T$ after each iteration. For the purposes of this paper, the points are strategies (policies) for the agent, so we will use the terms point and strategies interchangeably in this section.

The quality metric for a regret minimizer is its *regret*, which measures the difference in loss against the best *fixed* (that is, time-independent) decision in hindsight. Formally, given a decision $\boldsymbol{z} \in \mathcal{X}$, the regret cumulated against $\boldsymbol{z}$ up to time T is defined as

$$R^T(\boldsymbol{z}) := \sum_{t=1}^{T} (\boldsymbol{\ell}^t)^\top (\boldsymbol{x}^t - \boldsymbol{z}).$$

A "good" (aka. *Hannan consistent*) minimizer is such that the regret compared to *any* $\boldsymbol{z} \in \mathcal{X}$ grows sublinearly in T. This paper is interested in two types of regret minimizers, which differ in the feedback that the algorithm receives.

Full-Information Setting. In the *full-information* setting, at all time steps $t = 1, \dots, T$, the regret minimizer interacts with the environment as follows:

- NextStrategy(): the agent outputs the next point $\boldsymbol{x}^t \in \mathcal{X} \subseteq \mathbb{R}^n$. The next decision can depend on the past decisions $\boldsymbol{x}^1, \dots, \boldsymbol{x}^{t-1}$ as well as the corresponding feedback $\boldsymbol{\ell}^1, \dots, \boldsymbol{\ell}^{t-1}$, which we define next;
- ObserveLoss($\boldsymbol{\ell}^t$): the environment selects a loss vector $\boldsymbol{\ell}^t \in \mathbb{R}^n$ and the agent observes $\boldsymbol{\ell}^t$. The loss vector can depend on the decisions $\boldsymbol{x}^1, \dots, \boldsymbol{x}^t$ that were output by the regret minimizer so far.

Our construction of $\tilde{\mathcal{R}}$ (Section 4) provides a full-information regret minimizer for the set $\mathcal{X} = Q$. So, $\tilde{\mathcal{R}}$'s decisions are (potentially randomized) sequence-form strategies.

Bandit Setting. In the *bandit* setting the environment does *not* reveal the selected loss vector $\boldsymbol{\ell}^t$ at each iteration, but only the evaluation $(\boldsymbol{\ell}^t)^\top \boldsymbol{x}^t$ of the loss function for the latest decision $\boldsymbol{x}^t$. Formally, at all time steps $t = 1, \dots, T$, the regret minimizer interacts with the environment as follows:

- NextStrategy(): the agent outputs the next point $\boldsymbol{x}^t \in \mathcal{X} \subseteq \mathbb{R}^n$. As in the full-information setting, the next strategy can depend on the past strategies and corresponding feedbacks, which we define next;
- ObserveLossEvaluation($(\boldsymbol{\ell}^t)^\top \boldsymbol{x}^t$): the environment selects a loss vector $\boldsymbol{\ell}^t \in \mathbb{R}^n$ and the agent observes $(\boldsymbol{\ell}^t)^\top \boldsymbol{x}^t$. We assume without loss of generality that $(\boldsymbol{\ell}^t)^\top \boldsymbol{x}^t \in [0, 1]$ at all t. The loss vector can depend on the decisions $\boldsymbol{x}^1, \dots, \boldsymbol{x}^{t-1}$ that were output by the regret minimizer *before* time t, but *not* on $\boldsymbol{x}^t$.

Since the regret minimizer only observes $(\boldsymbol{\ell}^t)^\top \boldsymbol{x}^t$, it cannot compute any *counterfactual* information (that is, compute the value of the loss at a decision other than the one that was output). Currently, the bandit setting represents the hardest setting in which the information-theoretic upper bound of $\tilde{O}(\sqrt{T})$ regret is known to be attainable, but very little is known about sequential decision making under that setting, and existing algorithms are not computationally practical.[1]

4 Dilated Entropy and Local Norms

The dilated entropy distance-generating function (DGF) is a regularizer that induces a notion of distance that is suitable for the sequence-form strategies spaces. This regularizer was first introduced in the context of extensive-form games (Hoda et al. 2010). Kroer et al. (2020)—with earlier results by Kroer et al. (2015)—analyzed several properties of this function, including its 1-strong convexity with respect to the ℓ_1 and ℓ_2 norms. They also showed that the dilated entropy DGF leads to state-of-the-art convergence guarantees in iterative methods for computing Nash equilibrium in two-player zero-sum extensive-form games of perfect recall. We define this kind of DGF as follows.

Definition 1. *Let* $\operatorname{co}\Pi$ *be the set of sequence-form strategies for the TFSDM problem. The* dilated entropy *distance-generating function for* $\operatorname{co}\Pi$ *is the function* $\varphi : \mathbb{R}^{|\Sigma|}_{>0} \to \mathbb{R}_{\geq 0}$ *defined as*

$$\varphi : \boldsymbol{x} \mapsto \sum_{j \in \mathcal{J}} w_j \left(x_{p_j} \log |A_j| + \sum_{a \in A_j} x_{ja} \log \frac{x_{ja}}{x_{p_j}} \right),$$

where the weights w_j *are defined recursively according to:*

$$w_j = 2 + 2 \max_{a \in A_j} \{ w_{\rho(j,a)} \}, \quad w_k = \sum_{s \in S_k} w_{\rho(j,s)}, \quad w_\diamond = 0.$$

The range of φ is a game-dependent constant, and usually polynomial in the size of the TFSDM problem (Kroer et al. 2017). The (unique) minimum of φ is attained by the sequence-form strategy that at each decision point uniformly randomizes among all available actions (that is, $x_{ja} = x_{p_j}/|A_j|$ for all $j \in \mathcal{J}, a \in A_j$).

The dilated entropy DGF has the benefit that its gradient and its Fenchel conjugate function can be evaluated efficiently via a linear-time pass of the decision process (Hoda et al. 2010). In particular, for all $\boldsymbol{z} \in \mathbb{R}^{|\Sigma|}_{>0}$, there exists an exact algorithm, denoted Gradient, to compute $\nabla\varphi(\boldsymbol{z})$ in linear time in $|\Sigma|$. Also, there exists an exact algorithm, denoted ArgConjugate, to compute $\nabla\varphi^*(\boldsymbol{z}) = \arg\max_{\hat{\boldsymbol{x}} \in \operatorname{co}\Pi} \{\boldsymbol{z}^\top \hat{\boldsymbol{x}} - \varphi(\hat{\boldsymbol{x}})\}$ in linear time in $|\Sigma|$. This makes φ an appealing candidate regularizer in many TFSDM

[1] A third online learning setting—called the *semi-bandit optimization setting*—has been proposed in the literature (György et al. 2007; Kale, Reyzin, and Schapire 2010; Audibert, Bubeck, and Lugosi 2014; Neu and Bartók 2013). The feedback that the decision maker receives at all times t in that setting is the component-wise product $\boldsymbol{\ell}^t \circ \boldsymbol{x}^t$. The semi-bandit feedback provides counterfactual information. Instead, in this paper we are interested in the bandit setting, where no counterfactual information is available.

optimization algorithms, including the full-information regret minimizer $\tilde{\mathcal{R}}$ that we use in this paper. In Appendix B in the full version of this paper[2] we give pseudocode for GRADIENT and ARGCONJUGATE.

As mentioned in the introduction, the analysis of our bandit regret minimizer needs to take into consideration the particular geometry of the dilated entropy DGF. Specifically, at each point $\boldsymbol{x} \in Q$ in the sequence-form strategy space, the dilated entropy DGF induces a pair of primal-dual *local* norms $(\|\cdot\|_{\boldsymbol{x}}, \|\cdot\|_{*,\boldsymbol{x}})$ defined for all $\boldsymbol{z} \in \mathbb{R}^{|\Sigma|}$ as

$$\|\boldsymbol{z}\|_{\boldsymbol{x}} := \sqrt{\boldsymbol{z}^\top \nabla^2 \varphi(\boldsymbol{x})\, \boldsymbol{z}}; \quad \|\boldsymbol{z}\|_{*,\boldsymbol{x}} := \sqrt{\boldsymbol{z}^\top (\nabla^2 \varphi(\boldsymbol{x}))^{-1} \boldsymbol{z}},$$

where $\nabla^2\varphi(\boldsymbol{x})$ denotes the Hessian matrix of φ at $\boldsymbol{x}$. Since $\nabla^2\varphi(\boldsymbol{x})$ is positive-definite, it is known that $\|\cdot\|_{*,\boldsymbol{x}}$ is well-defined and that it is indeed dual to $\|\cdot\|_{\boldsymbol{x}}$, in the sense that $\|\boldsymbol{z}\|_{*,\boldsymbol{x}} = \max\{\boldsymbol{z}^\top \boldsymbol{w} : \|\boldsymbol{w}\|_{\boldsymbol{x}} \le 1\}$ for all $\boldsymbol{z} \in \mathbb{R}^{|\Sigma|}$.

To our knowledge, we are the first to explore the local norms induced by the dilated entropy DGF. These norms are a fundamental ingredient in our construction, and here we give several properties that we will use in later sections. In Appendix B.2 in the full version of this paper we give several results regarding analytical properties of these norms, including a useful characterization of the inverse Hessian matrix of the DGF φ at a generic strategy $\boldsymbol{x} \in Q$ in terms of sum of dyadics.

5 Construction of $\tilde{\mathcal{R}}$

Our full-information regret minimizer $\tilde{\mathcal{R}}$ is constructed using online mirror descent—one of the most well-studied full-information regret minimization algorithms in online learning—instantiated with the dilated entropy DGF φ (Definition 1) as the regularizer and the set $Q \subseteq \mathbb{R}^{|\Sigma|}$ of sequence-form strategies in the game as the domain of feasible iterates. Pseudocode for $\tilde{\mathcal{R}}$ is given in Algorithm 1, where $\eta > 0$ is a stepsize parameter that can be tuned at will.

Those properties are key to the analysis of the regret cumulated by Algorithm 1 as a function of the local dual norms of the loss vectors $\tilde{\boldsymbol{\ell}}^t$; that analysis is rather lengthy and we defer it to Appendix C in the full version of this paper. Here, we only state a key result.

Theorem 1. *Let D be the maximum depth of any node in the decision process, and let $\boldsymbol{z} \in Q$. If $\tilde{\boldsymbol{\ell}}^t \in \mathbb{R}^{|\Sigma|}_{\ge 0}$ at all times t, then at all times T the regret $\tilde{R}^T(\boldsymbol{z})$ cumulated by $\tilde{\mathcal{R}}$ satisfies:*

$$\tilde{R}^T(\boldsymbol{z}) \le \frac{\varphi(\boldsymbol{z})}{\eta} + \eta\sqrt{3D} \cdot \sum_{t=1}^{T} \|\tilde{\boldsymbol{\ell}}^t\|^2_{*,\boldsymbol{x}^t}. \qquad (2)$$

Incidentally, since the range of φ over Q only depends on the TFSDM problem structure and not on the time T, by setting $\eta = \Theta(1/\sqrt{T})$, we obtain a regret bound of the form

$$\mathbb{E}\left[\tilde{R}^T(\boldsymbol{z})\right] = O\left(\frac{1}{\sqrt{T}}\,\mathbb{E}\left[\sum_{t=1}^{T} \|\tilde{\boldsymbol{\ell}}^t\|^2_{*,\boldsymbol{x}^t}\right]\right).$$

In the next section, we show how to construct $\tilde{\boldsymbol{\ell}}$ in the right hand side and we prove that the right hand side is small in expectation. Then in Section 7 we prove that the expectation of the regret on the left hand side equals the expectation of the regret of the bandit regret minimizer $\mathcal{R}$.

6 Unbiased Loss Estimate and Construction of $\mathcal{R}$

As mentioned in Section 1.1, two different components are crucial for our bandit regret minimizer $\mathcal{R}$: the sampling scheme and the construction of the unbiased loss estimates.

6.1 Sampling Scheme for TFSDM

At each time step t, the bandit regret minimizer $\mathcal{R}$ internally calls $\tilde{\mathcal{R}}$.NEXTSTRATEGY() and receives a sequence-form strategy $\boldsymbol{x}^t \in Q$. Then, $\mathcal{R}$ samples and returns a *pure* sequence-form strategy $\boldsymbol{y}^t \in \Pi$ such that $\mathbb{E}_t[\boldsymbol{y}^t] = \boldsymbol{x}^t$. We use the natural sampling scheme for sequence-form strategies: at each decision point j, we pick an action $a \in A_j$ according to the distribution $x^t_{ja}/x^t_{p_j}$ induced by the sequence-form strategy $\boldsymbol{x}^t$. It is well known (and straightforward to verify—see Appendix D.2 in the full version of this paper) that this sampling scheme is unbiased.

As we will show, in order to balance exploration and exploitation along the structure of the TFSDM problem and construct unbiased loss estimates, an analysis of the autocorrelation matrix $\boldsymbol{C}^t := \mathbb{E}[\boldsymbol{y}^t(\boldsymbol{y}^t)^\top \mid \boldsymbol{y}^1, \dots, \boldsymbol{y}^{t-1}]$ of the sampling scheme, as well as its inverse, can be used. To our knowledge, we are the first to study the autocorrelation matrix of the natural sampling scheme for sequence-form strategies. We do so in Appendix D.3.

6.2 Computation of the Loss Estimate $\tilde{\boldsymbol{\ell}}^t$

Our construction of the unbiased loss estimate extends and generalizes that of Dani, Kakade, and Hayes (2008), in that it can be applied even when the set of strategies is rank deficient, for example here where the pure strategies Π of the sequence form strategy space only span a strict subspace of the natural Euclidean space $\mathbb{R}^{|\Sigma|}$ to which the sequence-form strategies belong. In particular, we relax the notion of unbiasedness to mean the weaker condition that the projection $\tilde{\boldsymbol{\ell}}^t$ onto the direction[3] $\operatorname{dir}\Pi$ of Π be an unbiased estimator of the projection of the original (and unknown) $\boldsymbol{\ell}^t$ onto $\operatorname{dir}\Pi$:

$$\mathbb{E}_t[\tilde{\boldsymbol{\ell}}^t]^\top \boldsymbol{w} = (\boldsymbol{\ell}^t)^\top \boldsymbol{w} \qquad \forall\, \boldsymbol{w} \in \operatorname{dir}\Pi, \qquad (\star)$$

where $\mathbb{E}_t[\,\cdot\,]$ is an abbreviation for $\mathbb{E}_t[\,\cdot \mid \boldsymbol{y}^1, \dots, \boldsymbol{y}^{t-1}]$, that is, the expectation conditional on the previous decisions of $\mathcal{R}$. The main technical tool in our construction is the use of a generalized inverse of the autocorrelation matrix of $\boldsymbol{y}^t$, as shown by the next proposition (the proof is in Appendix D in the full version of this paper).

Proposition 1. *Let π^t be a conditional distribution over Π, given the previous decisions $\boldsymbol{y}^1, \dots, \boldsymbol{y}^{t-1}$, and suppose that the support of π^t has full rank (that is,* $\operatorname{span}\operatorname{supp}\pi^t =$

[2] The full version of this paper, including appendix, is available on arXiv.

[3] The direction $\operatorname{dir}\mathcal{X}$ of a set $\mathcal{X}$ is the subspace defined as $\operatorname{dir}\mathcal{X} := \operatorname{span}\{\boldsymbol{u} - \boldsymbol{v} : \boldsymbol{u}, \boldsymbol{v} \in \mathcal{X}\}$.

Algorithm 1: Full-information regret minimizer $\tilde{\mathcal{R}}$

```
Data: $\eta$ is a step-size parameter.
function SETUP()
    for $j \in \mathcal{J}$ in top-down order do
        for $a \in A_j$ do $x^1_{ja} \leftarrow \frac{x_{p_j}}{|A_j|}$
function NEXTSTRATEGY(): return $\boldsymbol{x}^t$
function OBSERVELOSS($\tilde{\boldsymbol{\ell}}^t$)
    $\boldsymbol{g} \leftarrow \eta\tilde{\boldsymbol{\ell}}^t -$ GRADIENT($\boldsymbol{x}^t$)        [▷ Section 4]
    $\boldsymbol{x}^{t+1} \leftarrow$ ARGCONJUGATE($-\boldsymbol{g}$)        [▷ Section 4]
```

Algorithm 2: Bandit regret minimizer $\mathcal{R}$

```
function SETUP()
    $\tilde{\mathcal{R}}$.SETUP()                                  [▷ Algorithm 1]
function NEXTSTRATEGY()
    $\boldsymbol{x}^t \leftarrow \tilde{\mathcal{R}}$.NEXTSTRATEGY()               [▷ Algorithm 1]
    $\boldsymbol{y}^t \leftarrow$ SAMPLE($\boldsymbol{x}^t$)                      [▷ Section 6.1]
    return $\boldsymbol{y}^t$
function OBSERVELOSSEVALUATION($l := (\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t$)
    $\tilde{\boldsymbol{\ell}}^t \leftarrow$ LOSSESTIMATE($l, \boldsymbol{x}^t, \boldsymbol{y}^t$)     [▷ Algorithm 3]
    $\tilde{\mathcal{R}}$.OBSERVELOSS($\tilde{\boldsymbol{\ell}}^t$)                [▷ Algorithm 1]
```

Algorithm 3: LOSSESTIMATE($l, \boldsymbol{x}^t, \boldsymbol{y}^t$)

```
$\tilde{\boldsymbol{\ell}}^t \leftarrow \mathbf{0} \in \mathbb{R}^{|\Sigma|}_{\geq 0}$
subroutine TRAVERSE($v, \alpha_v$)
    if $v \in \mathcal{K}$ then
        for $s \in S_v$ do
            TRAVERSE$\left(\rho(v,s), \frac{\alpha_v}{|S_v|} + \frac{|S_v|-1}{|S_v|}(1-l)y^t_{p_v}\right)$
    else                                                [▷ that is, $v \in \mathcal{J}$]
        for $a \in A_v$ do
            if $\rho(v,a) \neq \diamond$ then
                $\ell^t_{ja} \leftarrow \frac{y^t_{va}}{x^t_{va}}(N_v - N_{\rho(v,a)})$
                TRAVERSE$\left(\rho(v,a), \frac{x_{p_v}}{x_{va}}\alpha_v\right)$
            else if $\rho(v,a) = \diamond$ then
                $\ell^t_{ja} \leftarrow \frac{\alpha_v}{x^t_{p_v}} + \frac{y^t_{va}}{x^t_{va}}(l + N_v - 1)$
TRAVERSE($r, 0$)                      [▷ $r$: root of the decision process]
return $\tilde{\boldsymbol{\ell}}^t$
```

span Π)*. Let $\boldsymbol{C}^t := \mathbb{E}_t[\boldsymbol{y}^t(\boldsymbol{y}^t)^\top]$ be the autocorrelation matrix of $\boldsymbol{y}^t$, and let $\boldsymbol{C}^{t-}$ be any generalized inverse of $\boldsymbol{C}^t$, that is, any matrix such that $\boldsymbol{C}^t\boldsymbol{C}^{t-}\boldsymbol{C}^t = \boldsymbol{C}^t$. Furthermore, let $\boldsymbol{b}^t$ be such that $\mathbb{E}_t[\boldsymbol{b}^t] \perp \operatorname{dir}\Pi$. Then, the random variable*

$$\tilde{\boldsymbol{\ell}}^t := [(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t] \cdot \boldsymbol{C}^{t-}\, \boldsymbol{y}^t + \boldsymbol{b}^t \tag{3}$$

satisfies ($\star$).

Crucially, the loss estimate $\tilde{\boldsymbol{\ell}}^t$ in Proposition 1 can be constructed using only the bandit feedback (loss evaluation) $(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t$ that was received at time t after the regret minimizer output $\boldsymbol{y}^t$ as its decision. At each time t, we use Proposition 1 to construct the loss estimate $\tilde{\boldsymbol{\ell}}^t$. The main conceptual leap is to identify

(i) a choice of generalized inverse $\boldsymbol{C}^{t-}_*$ for the autocorrelation matrix $\boldsymbol{C}^t$ of $\boldsymbol{y}^t$ returned by Algorithm 6; and

(ii) a particular choice for the (random) vector $\boldsymbol{b}^t_*$ such that $\mathbb{E}_t[\boldsymbol{b}^t_*] \perp \operatorname{dir}\Pi$ so that (a) the expression $[(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t]\boldsymbol{C}^{t-}_*\boldsymbol{y}^t + \boldsymbol{b}^t_*$ can be evaluated in $O(|\Sigma|)$ time and (b) the resulting loss function $\tilde{\boldsymbol{\ell}}^t$ is nonnegative, as required by $\tilde{\mathcal{R}}$ (Theorem 1).

At a high level, the particular construction that we use generates $\boldsymbol{C}^{t-}_*$ and $\boldsymbol{b}^t_*$ inductively in a bottom-up fashion by traversing the decision process, and heavily relies on the combinatorial structure of the autocorrelation matrix $\boldsymbol{C}^t$. All details and proofs are in Appendix D.4 in the full version of this paper.

The resulting algorithm is Algorithm 3, where we let $l := (\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t$ denote the bandit feedback at iteration t in accordance to Algorithm 2.

Proposition 2. *At all times t, the vector $\tilde{\boldsymbol{\ell}}^t$ returned by* LOSSESTIMATE($l, \boldsymbol{x}^t, \boldsymbol{y}^t$) *satisfies* ($\star$)*. Furthermore, Algorithm 3 amounts to a single traversal of the tree structure of the TFSDM problem and runs in linear time in the number of sequences $|\Sigma|$.*

In the special case where the decision process only has one decision point (i.e., the strategy space is a simplex), the loss estimate constructed by Algorithm 3 coincides with that of the EXP3 algorithm of Auer et al. (2002). However, for general sequential decision processes, the loss estimate is significantly more complicated and not based on an importance-sampling-based argument anymore.

Finally, because of the assumption that $l = (\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t \in [0, 1]$, which can be assumed without loss of generality, the loss estimate constructed as just described is non-negative: $\tilde{\boldsymbol{\ell}}^t \in \mathbb{R}^{|\Sigma|}_{\geq 0}$. So, Theorem 1 applies.

6.3 Norm of the Loss Estimate

In theory, each entry of $\tilde{\boldsymbol{\ell}}^t$ can be arbitrarily large since x^t_{ja} can be arbitrarily small. As a consequence, the Euclidean norm $\|\tilde{\boldsymbol{\ell}}^t\|_2$ of the loss estimate can be arbitrarily large, even in expectation. This shows the importance of having Equation (2) expressed in terms of the *local* norms $\|\cdot\|_{*,\boldsymbol{x}^t}$ instead of a generic time-invariant norm. Indeed, it is possible to give guarantees on the expectation of the local dual norm of $\tilde{\boldsymbol{\ell}}^t$, as we do in the next theorem.

Theorem 2. *At all times t, the loss estimate $\tilde{\boldsymbol{\ell}}^t \in \mathbb{R}^{|\Sigma|}_{\geq 0}$ returned by* LOSSESTIMATE($l, \boldsymbol{x}^t, \boldsymbol{y}^t$)*, where r is the root of the sequential decision process, satisfies*

$$\mathbb{E}_t\left[\|\tilde{\boldsymbol{\ell}}^t\|^2_{*,\boldsymbol{x}^t}\right] \leq 4 \cdot |\Sigma|^3.$$

The proof is in Appendix E.3 in the full version of this paper. Theorem 2 is one of the deepest results in this paper. It ties together the sampling scheme (Section 6.1), the construction of the loss estimates (Section 6.2), and the geometry of the local norms (Section 4) induced by the dilated entropy DGF. It combines properties of the particular choice of generalized inverse $\boldsymbol{C}_*^{t-}$ and orthogonal vector $\boldsymbol{b}_*^t \perp \operatorname{dir}\Pi$ with an inductive argument on the TFSDM problem structure.

7 The Full Algorithm

We construct our bandit regret minimizer $\mathcal{R}$ (Algorithm 2) starting from the full-information regret minimizer $\tilde{\mathcal{R}}$ of Algorithm 1. The resulting algorithm is surprisingly easy to implement, and requires only two linear traversals of the decision process per iteration. The regret $R^T(\boldsymbol{z})$ of $\mathcal{R}$ is linked to the regret $\tilde{R}^T(\boldsymbol{z})$ of $\tilde{\mathcal{R}}$: using the definition of regret and the law of total expectation together with the standard bandit optimization assumption that $\boldsymbol{\ell}^t$ is conditionally independent from $\boldsymbol{y}^t$, as well as Lemma 12 and $(\star)$, we immediately find that $\mathbb{E}[R^T(\boldsymbol{z})] = \mathbb{E}[\tilde{R}^T(\boldsymbol{z})]$ Theorem 1 gives an upper bound for the regret $\tilde{R}^T(\boldsymbol{z})$ of $\tilde{\mathcal{R}}$ as a function of the sequence of the loss estimates $\tilde{\boldsymbol{\ell}}^1, \dots, \tilde{\boldsymbol{\ell}}^T$. Taking expectations in Equation (2) and using Theorem 2,

$$\begin{aligned}\mathbb{E}[\tilde{R}^T(\boldsymbol{z})] &\leq \frac{\varphi(\boldsymbol{z})}{\eta} + \eta\sqrt{3D}\cdot\mathbb{E}\left[\sum_{t=1}^{T}\mathbb{E}_t\left[\|\tilde{\boldsymbol{\ell}}^t\|^2_{*,\boldsymbol{x}^t}\right]\right]\\ &\leq \frac{\varphi(\boldsymbol{z})}{\eta} + 4\eta\,|\Sigma|^3\,\sqrt{3D}\cdot T.\end{aligned}$$

Setting $\eta = 1/(2|\Sigma|^{3/2}\sqrt{T})$, we obtain the following theorem, which is the bottom-line result of this paper.

Theorem 3. *Let D be the maximum depth of any node in the decision process. Then, assuming $(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t \in [0,1]$ at all times $t = 1, \dots, T$, the regret $R^T(\boldsymbol{z})$ cumulated by Algorithm 2 satisfies*

$$\mathbb{E}[R^T(\boldsymbol{z})] \leq 2(\varphi(\boldsymbol{z}) + \sqrt{3D})\,|\Sigma|^{3/2}\cdot\sqrt{T} \qquad \forall\, \boldsymbol{z} \in \operatorname{co}\Pi.$$

Theorem 3 shows that the expected regret cumulated by our algorithm is $O(\sqrt{T})$. This improves on the algorithm of Abernethy, Hazan, and Rakhlin (2008), whose regret is $O(\sqrt{T\log T})$ and that assumes that $T \geq |\Sigma|$. We conclude this section with a word of caution. Our algorithm, like the one of Abernethy, Hazan, and Rakhlin (2008), guarantees that $\max_{\boldsymbol{z}\in Q}\mathbb{E}[R^T(\boldsymbol{z})]$ is small, but *not* that $\mathbb{E}[\max_{\boldsymbol{z}\in Q} R^T(\boldsymbol{z})]$ is small. Depending on the application, this may or may not be sufficient. This limitation is well known (e.g., (Abernethy and Rakhlin 2009)) and is one of the main drivers behind regret minimizers that provide high-probability regret bounds. In the conclusions we will discuss how the techniques of the present paper are relevant toward that effort.

8 Experimental Evaluation

We implemented our bandit regret minimizer (Algorithm 2) and the algorithm of Abernethy, Hazan, and Rakhlin (2008) (using a logarithmic barrier) (from now on, denoted AHR), which is, to our knowledge, the only prior algorithm that is known to guarantee $\tilde{O}(\sqrt{T})$ regret and polynomial-time iterations in the bandit optimization setting. We compared them on four domains: a simple 2×3 matrix game (which is an instance of an TFSDM problem with no observation points and only one decision node), and three standard extensive-form games in the computational game theory literature, namely Kuhn poker (Kuhn 1950), 3-rank Goofspiel(Ross 1971), and Leduc poker (Southey et al. 2005). The sequential decision making problem faced by the first player has 13 sequences in Kuhn poker, 262 sequences in Goofspiel, and 337 sequences in Leduc poker. A complete description of those games is available in Appendix F in the full version of this paper. The two algorithms face the same strong opponent that at each iteration plays according to a fixed strategy $\bar{\boldsymbol{s}}$ that is part of a Nash equilibrium of the game. For our method, we use the theoretical step size multiplied by 5, while for the method of Abernethy, Hazan, and Rakhlin (2008) we multiply their step size parameter by 2; these changes do not affect the theoretical guarantees but improved the practical performances of both algorithms. Figure 2 shows the regret of the algorithms compared to always playing the best-response strategy against $\bar{\boldsymbol{s}}$.

We also report the empirical performance of online MCCFR (as proposed in a side note by Lanctot et al. (2009)) although, as we discussed at length in the introduction, online MCCFR—unlike the other two algorithms— 1) is not an algorithm for the bandit optimization setting (as discussed in the introduction, it also needs to observe the actions of the opponent, while our algorithm and AHR do not; in the experiment we give online MCCFR that additional benefit), and 2) does not have a known guarantee of sublinear regret in this setting. We ran each algorithm 100 times. Figure 2 shows these runs with thin light-colored lines. For the non-anytime algorithms (ours and AHR), we divided the desired runtime (e.g., 3 hours in Leduc poker) by the average time per iteration of each algorithm. We also plot the average regret across the 100 runs of each algorithm with a thick line.

In all games, our method yielded lower regret than AHR at all times. In the matrix game and Kuhn poker, our algorithm converges to a smaller regret than online MCCFR, despite the fact that we do not get to observe the path of play and online MCCFR does. In the larger and deeper games (Goofspiel and Leduc poker), our algorithm still clearly exhibits the guaranteed $O(\sqrt{T})$ cumulative regret, but has higher regret than online MCCFR. Empirically, the regret cumulated by AHR seems to match the theoretical $\tilde{O}(\sqrt{T})$ guarantee well in the small games, but not in Goofspiel and Leduc poker. The reason for this is twofold. First, the runtime cost of each iteration of AHR in those latter games is roughly three orders of magnitude higher than either our method or online MCCFR. A significant fraction (roughly 20%) of that runtime is due to the fact that AHR needs to compute an eigendecomposition of a Hessian matrix of the log-barrier at the current point, an expensive operation whose overhead grows roughly cubically with the number of sequences in the game. We use the Eigen 3.3.3 library to compute the eigendecomposition at each time t. Due to this reason, AHR performs significantly fewer iterations in the allotted time (three hours). The second issue that we identified with AHR is that it tends to suffer from serious numerical difficulties as the size of the TFSDM

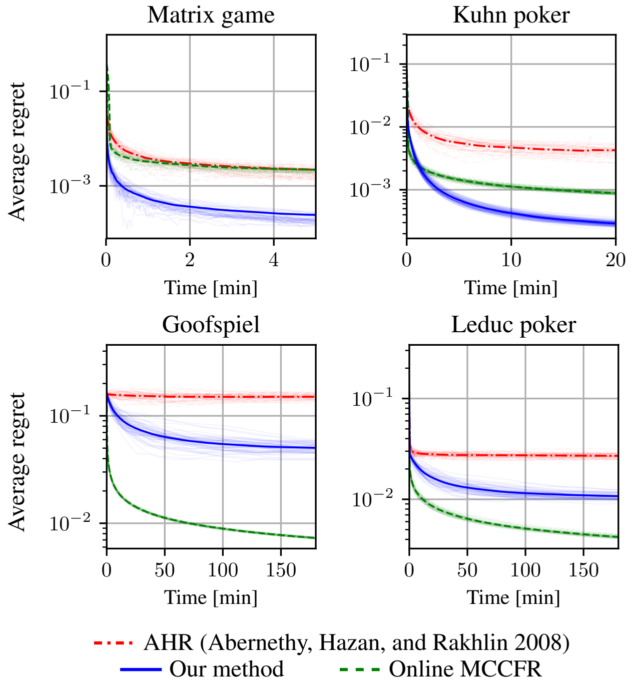

Figure 2: Evolution of the average regret in different bandit linear optimization algorithms (AHR and ours), as well as the online MCCFR algorithm (not an algorithm for bandit linear optimization).

problem grows.

9 Conclusions and Future Research

In this paper, we developed an algorithm for the bandit linear optimization on tree-form sequential decision making (TFSDM) problems. Our bandit regret minimizer is superior to that of Abernethy, Hazan, and Rakhlin (2008) both computationally (each iteration runs in linear time in the number of sequences in the problem) and in terms of cumulated regret (the regret is $O(\sqrt{T})$ instead of $O(\sqrt{T \log T})$. We also presented the first implementations of bandit optimization for TFSDM. Our method combines and contributes a number of ideas and tools. First, we gave several new results concerning the local analytic properties of the dilated entropy regularizer (the leading regularizer for TFSDM). We use those analytic properties to obtain a stronger regret bound for the online mirror descent algorithm instantiated with the dilated entropy regularizer. Second, we study several properties of the natural sampling scheme for sequence-form strategies. Those properties are key to efficiently constructing an unbiased estimator of the loss vector $\boldsymbol{\ell}^t$ starting from the loss evaluation $(\boldsymbol{\ell}^t)^\top \boldsymbol{y}^t$ at a pure strategy $\boldsymbol{y}^t$. In order to construct the unbiased estimator, we extended and generalized an argument by Bartlett et al. (2008) to our context. Finally, we combined the stronger regret bound for mirror descent together with the unbiased loss estimator to construct our bandit regret minimizer, by showing that the unbiased loss estimator has a dual norm that is bounded by a small time-independent constant.

A known weakness in Abernethy, Hazan, and Rakhlin (2008), which is also a weakness in our approach, is that the bound on regret (i) only holds in expectation as opposed to high probability, and (ii) provides a guarantee on $\max_{\boldsymbol{z} \in Q} \mathbb{E}[R^T(\boldsymbol{z})]$ but *not* on $\mathbb{E}[\max_{\boldsymbol{z} \in Q} R^T(\boldsymbol{z})]$. As discussed in the introduction, this weakness can be eliminated in theory if iterations are allowed to take exponential time in the number of sequences or by accepting a slower convergence rate. Due to this weakness, our approach and that of Abernethy, Hazan, and Rakhlin (2008) sometimes does not work for equilibrium finding in games through self play.

We leave the problem of designing an algorithm for the bandit linear optimization problem for TFSDM that guarantees both $O(\sqrt{T})$ regret *with high probability* and linear-time iterations as an open future direction. This would solve (i) and thereby also (ii). Abernethy and Rakhlin (2009) presented a template for deriving such algorithms, but several pieces therein need to be instantiated to complete the proof of bounds. The theory in this paper offers solutions for some of those pieces for general TFSDM problems. Our regularizer, the sampling scheme, the construction of the loss estimates, and the use of local norms can be used within that general framework to provide high-probability results. So, our results may help solve the open problem for TFSDM in the future.

Acknowledgments

This material is based on work supported by the National Science Foundation under grants IIS-1718457, IIS-1901403, and CCF-1733556, and the ARO under award W911NF2010081. Gabriele Farina is supported by a Facebook fellowship.

References

Abernethy, J.; Hazan, E.; and Rakhlin, A. 2008. Competing in the dark: An efficient algorithm for bandit linear optimization. In *In Proceedings of the 21st Annual Conference on Learning Theory (COLT)*.

Abernethy, J. D.; and Rakhlin, A. 2009. Beating the adaptive bandit with high probability. *2009 Information Theory and Applications Workshop* .

Audibert, J.-Y.; Bubeck, S.; and Lugosi, G. 2014. Regret in online combinatorial optimization. *Mathematics of Operations Research* 39(1): 31–45.

Auer, P.; Cesa-Bianchi, N.; Freund, Y.; and Schapire, R. E. 2002. The Nonstochastic Multiarmed Bandit Problem. *SIAM Journal of Computing* 32: 48–77.

Awerbuch, B.; and Kleinberg, R. D. 2004. Adaptive routing with end-to-end feedback: distributed learning and g eometric approaches. In *Proceedings of the Annual Symposium on Theory of Computing (STOC)*. ACM.

Bartlett, P. L.; Dani, V.; Hayes, T.; Kakade, S.; Rakhlin, A.; and Tewari, A. 2008. High-probability regret bounds for bandit online linear optimization. In *Conference on Learning Theory (COLT)*.

Borwein, J.; and Lewis, A. S. 2010. *Convex analysis and nonlinear optimization: theory and examples*. Springer Science & Business Media.

Bowling, M.; Burch, N.; Johanson, M.; and Tammelin, O. 2015. Heads-up Limit Hold'em Poker is Solved. *Science* 347(6218).

Braun, G.; and Pokutta, S. 2016. An efficient high-probability algorithm for Linear Bandits. ArXiv e-print cs/1610.02072.

Brown, N.; Kroer, C.; and Sandholm, T. 2017. Dynamic Thresholding and Pruning for Regret Minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Brown, N.; and Sandholm, T. 2015. Regret-Based Pruning in Extensive-Form Games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.

Brown, N.; and Sandholm, T. 2017a. Reduced Space and Faster Convergence in Imperfect-Information Games via Pruning. In *International Conference on Machine Learning (ICML)*.

Brown, N.; and Sandholm, T. 2017b. Safe and nested subgame solving for imperfect-information games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.

Brown, N.; and Sandholm, T. 2017c. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science* 359: 418–424.

Brown, N.; and Sandholm, T. 2019a. Solving imperfect-information games via discounted regret minimization. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Brown, N.; and Sandholm, T. 2019b. Superhuman AI for multiplayer poker. *Science* 365: 885–890.

Bubeck, S.; Lee, Y. T.; and Eldan, R. 2017. Kernel-based methods for bandit convex optimization. In *Proceedings of the Annual Symposium on Theory of Computing (STOC)*, 72–85.

Cesa-Bianchi, N.; and Lugosi, G. 2006. *Prediction, learning, and games*. Cambridge University Press.

Dani, V.; Kakade, S. M.; and Hayes, T. P. 2008. The Price of Bandit Information for Online Optimization. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.

Farina, G.; Kroer, C.; and Sandholm, T. 2019. Online Convex Optimization for Sequential Decision Processes and Extensive-Form Games. In *AAAI Conference on Artificial Intelligence (AAAI)*.

György, A.; Linder, T.; Lugosi, G.; and Ottucsák, G. 2007. The on-line shortest path problem under partial monitoring. *Journal of Machine Learning Research* 8(Oct): 2369–2403.

Hazan, E.; and Li, Y. 2016. An optimal regret algorithm for bandit convex optimization. ArXiv e-print cs/1603.04350.

Hoda, S.; Gilpin, A.; Peña, J.; and Sandholm, T. 2010. Smoothing Techniques for Computing Nash Equilibria of Sequential Games. *Mathematics of Operations Research* 35(2).

Kale, S.; Reyzin, L.; and Schapire, R. E. 2010. Non-stochastic bandit slate problems. In *Advances in Neural Information Processing Systems*, 1054–1062.

Koller, D.; Megiddo, N.; and von Stengel, B. 1994. Fast algorithms for finding randomized strategies in game trees. In *Proceedings of the 26th ACM Symposium on Theory of Computing (STOC)*.

Kroer, C.; Waugh, K.; Kılınç-Karzan, F.; and Sandholm, T. 2020. Faster algorithms for extensive-form game solving via improved smoothing functions. *Mathematical Programming*.

Kroer, C.; Waugh, K.; Kılınç-Karzan, F.; and Sandholm, T. 2015. Faster First-Order Methods for Extensive-Form Game Solving. In *Proceedings of the ACM Conference on Economics and Computation (EC)*.

Kroer, C.; Waugh, K.; Kılınç-Karzan, F.; and Sandholm, T. 2017. Theoretical and Practical Advances on Smoothing for Extensive-Form Games. In *Proceedings of the ACM Conference on Economics and Computation (EC)*.

Kuhn, H. W. 1950. A Simplified Two-Person Poker. In Kuhn, H. W.; and Tucker, A. W., eds., *Contributions to the Theory of Games*, volume 1 of *Annals of Mathematics Studies, 24*, 97–103. Princeton, New Jersey: Princeton University Press.

Lanctot, M.; Waugh, K.; Zinkevich, M.; and Bowling, M. 2009. Monte Carlo Sampling for Regret Minimization in Extensive Games. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.

Ling, C. K.; Fang, F.; and Kolter, J. Z. 2019. Large Scale Learning of Agent Rationality in Two-Player Zero-Sum Games. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Neu, G.; and Bartók, G. 2013. An efficient algorithm for learning with semi-bandit feedback. In *International Conference on Algorithmic Learning Theory*, 234–248. Springer.

Rakhlin, A. 2009. Lecture Notes on Online Learning. Unpublished lecture notes, http://www-stat.wharton.upenn.edu/~rakhlin/courses/stat991/papers/lecture_notes.pdf.

Romanovskii, I. 1962. Reduction of a Game with Complete Memory to a Matrix Game. *Soviet Mathematics* 3.

Ross, S. M. 1971. Goofspiel—the game of pure strategy. *Journal of Applied Probability* 8(3): 621–625.

Shalev-Shwartz, S. 2012. Online Learning and Online Convex Optimization. *Foundations and Trends® in Machine Learning* 4(2). ISSN 1935-8237. doi:10.1561/2200000018.

Southey, F.; Bowling, M.; Larson, B.; Piccione, C.; Burch, N.; Billings, D.; and Rayner, C. 2005. Bayes' Bluff: Opponent Modelling in Poker. In *Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence (UAI)*.

Tammelin, O.; Burch, N.; Johanson, M.; and Bowling, M. 2015. Solving Heads-up Limit Texas Hold'em. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI)*.

von Stengel, B. 1996. Efficient Computation of Behavior Strategies. *Games and Economic Behavior* 14(2): 220–246.

Zinkevich, M.; Bowling, M.; Johanson, M.; and Piccione, C. 2007. Regret Minimization in Games with Incomplete Information. In *Proceedings of the Annual Conference on Neural Information Processing Systems (NIPS)*.