# Online Convex Optimization for Sequential Decision Processes and Extensive-Form Games

**Gabriele Farina, Christian Kroer, Tuomas Sandholm**
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
{gfarina,ckroer,sandholm}@cs.cmu.edu

## Abstract

Regret minimization is a powerful tool for solving large-scale extensive-form games. State-of-the-art methods rely on minimizing regret locally at each decision point. In this work we derive a new framework for regret minimization on sequential decision problems and extensive-form games with general compact convex sets at each decision point and general convex losses, as opposed to prior work which has been for simplex decision points and linear losses. We call our framework *laminar regret decomposition*. It generalizes the CFR algorithm to this more general setting. Furthermore, our framework enables a new proof of CFR even in the known setting, which is derived from a perspective of decomposing polytope regret, thereby leading to an arguably simpler interpretation of the algorithm. Our generalization to convex compact sets and convex losses allows us to develop new algorithms for several problems: regularized sequential decision making, regularized Nash equilibria in zero-sum extensive-form games, and computing approximate extensive-form perfect equilibria. Our generalization also leads to the first regret-minimization algorithm for computing reduced-normal-form quantal response equilibria based on minimizing local regrets. Experiments show that our framework leads to algorithms that scale at a rate comparable to the fastest variants of counterfactual regret minimization for computing Nash equilibrium, and therefore our approach leads to the first algorithm for computing quantal response equilibria in extremely large games. Our algorithms for (quadratically) regularized equilibrium finding are orders of magnitude faster than the fastest algorithms for Nash equilibrium finding; this suggests regret-minimization algorithms based on decreasing regularization for Nash equilibrium finding as future work. Finally we show that our framework enables a new kind of scalable opponent exploitation approach.

## Introduction

*Counterfactual regret minimization (CFR)* (Zinkevich et al. 2007), and the newest variant *CFR*$^+$ (Tammelin et al. 2015), have been a central component in several recent milestones in solving imperfect-information *extensive-form games (EFGs)*. Bowling et al. (2015) used CFR$^+$ to near-optimally solve heads-up limit Texas hold'em. Brown and Sandholm (2017b) and Moravčík et al. (2017) used CFR variants, along with other techniques (Brown and Sandholm 2017a), to create AIs that beat professional poker players at the larger game of heads-up no-limit Texas hold'em.

We can view the CFR approach more generally as a methodology for setting up regret minimization for sequential decision problems (whether single- or multi-agent), where each decision point requires selecting either an action or a point from the probability distribution over actions. The crux of CFR is counterfactual regret, which leads to a definition of regret local to each decision point. CFR can then be viewed as the observation, and proof, that bounds on counterfactual regret, which can be minimized locally, lead to bounds on the overall regret. To minimize local regret, the framework relies on regret minimizers that operate on a simplex (typically of probabilities over the available actions), such as *regret matching* (RM) (Hart and Mas-Colell 2000) or the newer variant *regret matching*$^+$ (RM$^+$) (Tammelin et al. 2015).

In this paper we consider the more general problem of how to minimize regret over a sequential decision-making (SDM) polytope, where we allow arbitrary compact convex subsets of simplexes at each decision point (as opposed to only simplexes in CFR), and general convex loss functions (as opposed to only linear losses in CFR). This allows us to model a form of online convex optimization over SDM polytopes. We derive a decomposition of the polytope regret into local regret at each decision point. This allows us to minimize regret locally as with CFR, but for general compact convex decision points and convex losses. We call our decomposition *laminar regret decomposition (LRD)*. We call our overall framework for convex losses and compact convex decision points *laminar regret minimization (LRM)*. As a special case, our framework provides an alternate view of why CFR works—one that may be more intuitive for those with a background in online convex optimization.

Our generalization to compact convex sets allows us to model entities such as $\epsilon$-perturbed simplexes (Farina and Gatti 2017; Farina, Kroer, and Sandholm 2017; Kroer, Farina, and Sandholm 2017), and thus yields new algorithms for computing approximate equilibrium refinements for EFGs. General convex losses in SDM and EFG contexts have, to our knowledge, not been considered before. This generalization enables fast algorithms for many new settings.

One is to compute regularized zero-sum equilibria. If we apply a convex regularization function at each simplex, we can apply our framework to solve the resulting game. For the negative entropy regularizer this is equivalent to the dilated entropy distance function used for solving EFGs with first-order methods (Hoda et al. 2010; Kroer et al. 2015; 2018). Ling, Fang, and Kolter (2018) show that dilated-entropy-regularized EFGs are equivalent to quantal response equilibria (QRE) in the corresponding reduced normal-form game. Thus our result yields the first regret-minimization algorithm for computing reduced-normal-form quantal response equilibria in EFGs. Our experiments show that this enables QREs to be computed at a rate that is competitive with that of CFR$^+$ for computing Nash equilibria. These are the first algorithms for finding a QRE in large games.

Our experiments show that $\ell_2$-regularized equilibria can be computed at a rate that is orders of magnitude faster than the fastest algorithms for Nash equilibrium. This shows that our approach can be used to compute regularized equilibria in extremely large games such as real poker games.

Finally, we show that our framework enables a new kind of opponent-exploitation approach for large games, by adding a convex regularizer that penalizes the exploiter for being far away from a pre-computed Nash equilibrium. This allows the exploiter to control the tradeoff between exploitation and exploitability.

**Paper Structure**. We start with a brief review of regret minimization. Then, we introduce the domain on which we operate, sequential decision making. In the section "Regret in Sequential Decision Making" we introduce a notion of regret suitable for the SDM domain. Our central result is the laminar regret decomposition (LRD) in Theorem 1, which shows that the regret over the whole SDM polytope can be decomposed into regret at each decision point, and Theorem 2 which shows that overall regret can be minimized by minimizing these local regrets. Thus we combine LRD with a set of local regret minimizers, one for each decision point. The local regret minimizers are each given the modified losses shown in (4). This gives a regret-minimization algorithm for SDM. Finally, we discuss how our results can be applied to zero-sum games and more general convex-concave saddle point problems, and show experimental results.

## Regret Minimization

We work inside of the online learning framework called *online convex optimization* (Zinkevich 2003). In this setting, a decision maker repeatedly plays against an unknown environment by making a sequence of decisions $x^1, x^2, \ldots$. As customary, we assume that the set $X \subseteq \mathbb{R}^n$ of all possible decisions for the decision maker is convex and compact. The outcome of each decision $x^t$ is evaluated as $\ell^t(x^t)$, where $\ell^t$ is a convex function *unknown* to the decision maker until the decision is made. Abstractly, a *regret minimizer* is a device that supports two operations:

- it gives a *recommendation* for the next decision $x^{t+1} \in X$;
- it receives/observes the convex loss function $\ell^t$ used to "evaluate" decision $x^t$.

The learning is *online* in the sense that the decision maker/regret minimizer's next decision, $x^{t+1}$, is based only on the previous decisions $x^1, \ldots, x^t$ and corresponding loss observations $\ell^1, \ldots, \ell^t$.

In this paper, we adopt (external) *regret* as a way to evaluate the quality of the regret minimizer. Formally, the *cumulative regret* at time $T$ is defined as

$$R^T := \sum_{t=1}^{T} \ell^t(x^t) - \min_{\hat{x} \in X} \sum_{t=1}^{T} \ell^t(\hat{x}),$$

It measures the difference between the loss cumulated by the sequence of decisions $x^1, \ldots, x^T$ and the loss that would have been cumulated by playing the best time-independent decision $\hat{x}$ in hindsight. A desirable property of a regret minimizer is *Hannan consistency*: the average regret approaches zero, that is, $R^T$ grows at a *sublinear* rate in $T$.

Our framework relies on having a regret minimizer for the convex sets at each decision point. Each regret minimizer can be chosen independently and any regret minimizer can be used provided it has the Hannan consistency property for the local convex sets to which it is applied. Several regret minimizers are known. Perhaps the most general is *online mirror descent* (OMD). OMD generalizes several other regret minimizers, such as *online gradient descent* (OGD), exponential weights, and regularized follow-the-leader (Zinkevich 2003; Hazan and Kale 2010; Hazan 2016). The regret generally grows as $O(T^{-1/2})$ for these algorithms.

An alternative to our framework is to run an off-the-shelf regret minimizer (such as OMD) over the entire SDM polytope. For example, we could do that by applying the *distance generating function* of Kroer et al. (2018). However, decomposition into local regret minimization at each decision point has been dramatically more effective in practice, possibly because it allows to better leverage the problem structure.

**Linear Losses and Games.** Regret minimization methods for normal-form and extensive-form games usually involve minimizing the regret induced by linear loss functions. When the domain at each decision point $X_j$ is the $n_j$-dimensional simplex $\Delta_{n_j}$, the two most successful regret-minimizers in practice have been *regret matching* (Blackwell 1956) and *regret matching*$^+$ (Tammelin et al. 2015). These regret minimizers also have regret that grows at a rate $T^{-1/2}$ as with OMD, but they have a worse dependence on the dimension $n_j$. Nonetheless, they seem to perform better in practice when coupled with CFR (Brown, Kroer, and Sandholm 2017).

## Sequential Decision Making

It turns out that the results of this paper can be proven in a general setting which we call a sequential decision making. At each stage, the agent chooses a point in a simplex (or a subset of it). The chosen point incurs a convex loss and defines a probability distribution over the actions of the simplex. An action is sampled according to the chosen distribution, and the agent then arrives at a next decision point, potentially randomly selected out of several candidates. The reason the agent chooses points in the convex hull of actions, rather than simply an action, is that this gives us greater flexibility in representing decision points where agents wish to randomize over actions. This is the case for example in game-theoretic equilibria or when solving the decision-making problem with an iterative optimization algorithm.

Formally, we assume that we have a set of decision points $\mathcal{J}$. Each decision point $j \in \mathcal{J}$ has a set of actions $A_j$ of size

$n_j$. The decision space at each decision point $j$ is represented by a convex set $X_j \subseteq \Delta_{n_j}$. A point $x_j \in X_j$ represents a probability distribution over $A_j$. When a point $x_j$ is chosen, an action is sampled randomly according to $x_j$. Given a specific action at $j$, the set of possible decision points that the agent may next face is denoted by $\mathcal{C}_{j,a}$. It can be an empty set if no more actions are taken after $j, a$. We assume that the decision points form a tree, that is, $\mathcal{C}_{j,a} \cap \mathcal{C}_{j',a'} = \emptyset$ for all other convex sets and action choices $j', a'$. This condition is equivalent to the perfect-recall assumption in extensive-form games, and to conditioning on the full sequence of actions and observations in a finite-horizon partially-observable decision process. In our definition, the decision space starts with a root decision point, whereas in practice multiple root decision points may be needed, for example in order to model different starting hands in card games. Multiple root decision points can be modeled in our framework by having a dummy root decision point with only a single action.

The set of possible next decision points after choosing action $a \in A_j$ at $X_j$, denoted $\mathcal{C}_{j,a}$, can be thought of as representing the different decision points that an agent may face after taking action $a$ and then making an observation on which she can condition her next action choice. For example, in a card game an action may be to raise (that is, put money into the pot), and an observation could be the set of actions taken by the other players, as well as any cards dealt out, until the agent acts again. Each specific observation of actions and cards then corresponds to a specific decision point in $\mathcal{C}_{j,a}$.

We will relate the regret over the whole decision space to regret at subtrees in the decision space and individual convex sets. In order to do that we need ways to refer to each of these structures. Given a strategy $x$, $x_j$ is the (sub)vector belonging to the decision space $X_j$ at decision point $j$. Similarly, $x_{j,a}$ is the scalar associated with action $a \in A_j$ at decision point $j$, and in typical applications it is the probability of choosing action $a$ at decision point $j$. Subscript $\triangle_j$ denotes the portion of $x$ containing the decision variables for decision point $j$ and all its descendants. Finally, $x$ refers to the vector for the whole SDM, which corresponds to subscript $\triangle_r$ where $r$ is the root of the tree.

As an illustration, consider the game of Kuhn poker (Kuhn 1950). Kuhn poker consists of a three-card deck: king, queen, and jack. Each player is dealt one of the three cards and a single round of betting occurs. The action space for the first player is shown in Figure 1. For instance, we have: $\mathcal{J} = \{0, 1, 2, 3, 4, 5, 6\}$; $n_0 = 1, X_0 = \Delta_1 = \{1\}$; $n_j = 2, X_j = \Delta_2$ for all $j \in \mathcal{J} \setminus \{0\}$; $A_0 = \{\text{start}\}$, $A_1 = A_2 = A_3 = \{\text{check}, \text{raise}\}$, $A_4 = A_5 = A_6 = \{\text{fold}, \text{call}\}$; $\mathcal{C}_{0,\text{start}} = \{1, 2, 3\}$, $\mathcal{C}_{1,\text{raise}} = \emptyset$, $\mathcal{C}_{3,\text{check}} = \{6\}$; $X_{\triangle_1} = X_1 \times X_4$, $X_{\triangle_4} = X_4$; $x_1 = [x_{1,\text{check}}, x_{1,\text{raise}}]$; $x_{\triangle_1} = [x_1; x_4]$, etc.

In addition to games, our model captures, for example, POMDPs and MDPs where we condition on the entire history of observations and actions.

## Sequence Form for Sequential Decision Processes

So far we have described the decision space as a product of convex sets where the choice of each action is taken from a subset of a simplex $X_j \subseteq \Delta_{n_j}$. This formulation has a drawback: the expected-value function for a given strategy is not linear. Consider taking action $a$ at decision point $j$. In order to compute the expected overall contribution of that decision, its local payoff $g_{j,a}$ has to be weighted by the product of probabilities of all actions on the path to $j$ and by $x_{j,a}$. So, the overall expected utility is nonlinear and non-convex. We now present a well-known alternative representation of this decision space which preserves linearity. While we will mainly be working in the product space $X$, it will occasionally be useful to move to this equivalent representation to preserve linearity.

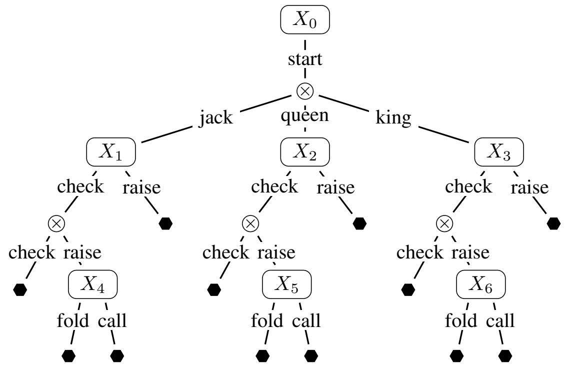

Figure 1: Sequential action space for the first player in the game of Kuhn poker. ⊗ denotes an observation point; ⬢ represents the end of the decision process.

The alternative formulation is called the *sequence form*. In that representation, every convex set $j \in \mathcal{J}$ is scaled by the parent variable leading to $j$. In other words, the sum of values at $j$ now sum to the value of the parent variable. In this formulation, the value of a particular action represents the probability of playing the whole *sequence* of actions from the root to that action. This allows each term in the expected loss to be weighted only by the sequence ending in the corresponding action. The sequence form has been used to instantiate linear programming (von Stengel 1996) and first-order methods (Hoda et al. 2010; Kroer et al. 2015; 2018) for computing Nash equilibria of zero-sum EFGs. There is a straightforward mapping between any $x \in X$ to its corresponding sequence form: simply assign each sequence the product of probabilities in the sequence. Likewise, going from sequence form to $X$ can be done by dividing each $x_{j,a}$ by the value $x_{p_j}$ where $p_j$ is the entry in $x$ corresponding to the parent of $j$. We let $\mu$ be a function that maps each $x \in X$ to its corresponding sequence-form vector. For the reverse direction $\mu^{-1}$, there is ambiguity because $\mu$ is not injective. Nonetheless, an inverse can be computed in linear time.

## Regret in Sequential Decision Making

We assume that we are playing a sequence of $T$ iterations of a sequential decision process. At each iteration $t$ we choose a strategy $x \in X$ and are then given a loss function of the form

$$\ell^t(x) := \sum_{j \in \mathcal{J}} \pi_j(x) \ell_j^t(x_j), \tag{1}$$

where $\ell_j^t : X_j \to \mathbb{R}$ is a convex function for each $j \in \mathcal{J}$ and $\pi_j$ is the probability of the process reaching $j$, computed as the product of the probability of each action in the path from the root of the tree to decision point $j$. We coin loss functions of this form *separable*, and they will play an important role in our results. Our goal is to compute a new strategy vector $x^t$ such that the regret across all $T$ iterations is as low as possible against any sequence of loss functions.

We now summarize definitions for the value and regret associated with convex sets and strategies. First we have the value of convex set $j$ at iteration $t$ when following strategy $\hat{x}$:

$$\hat{V}^t_{\triangle_j}(\hat{x}_{\triangle_j}) := \ell^t_j(\hat{x}^t_j) + \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{j,a}} \hat{x}_{j,a} \hat{V}^t_{\triangle_{j'}}(\hat{x}_{\triangle_{j'}}). \quad (2)$$

This definition denotes the utility associated with starting at convex set $X_j$ rather than at the root. We will be particularly interested in the value of $x^t$, which we denote $V^t_{\triangle_j} := \hat{V}^t_{\triangle_j}(x^t_{\triangle_j})$.

Now we can define the cumulative regret at convex set $j$ across all $T$ iterations as

$$R^T_{\triangle_j} := \sum_{t=1}^T V^t_{\triangle_j} - \min_{\hat{x}_{\triangle_j}} \sum_{t=1}^T \hat{V}^t_{\triangle_j}(\hat{x}_{\triangle_j}),$$

This can equivalently be stated as

$$\min_{\hat{x}_{\triangle_j}} \sum_{t=1}^T \hat{V}^t_{\triangle_j}(\hat{x}_{\triangle_j}) = \sum_{t=1}^T V^t_{\triangle_j} - R^T_{\triangle_j}. \quad (3)$$

Finally, *average regret* is $\bar{R}^T_{\triangle_j} := \frac{1}{T} R^T_{\triangle_j}$.

## Laminar Regret Decomposition

We now define a new parameterized class of loss functions for each subtree $X_j$ which we will show can be used to minimize regret over $X$ by minimizing that loss function independently at each convex set $X_j$. The loss function is

$$\hat{\ell}^t_j : x_j \mapsto \ell^t_j(x_j) + \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{j,a}} x_{j,a} V^t_{\triangle_{j'}}. \quad (4)$$

It is convex since $\ell^t_j$ is convex by hypothesis and we are only adding a linear term to it. Strict convexity is also preserved, and for strongly convex losses, the strong convexity parameter remains unchanged.

We now prove that the regret at information set $j$ decomposes into regret terms depending on $\hat{\ell}^t_j$ and a sum over the regret at child convex sets:

**Theorem 1.** *The cumulative regret at a decision point $j$ can be decomposed as*

$$R^T_{\triangle_j} = \sum_{t=1}^T \hat{\ell}^t_j(x^t) - \min_{\hat{x}_j \in X_j} \left\{ \sum_{t=1}^T \hat{\ell}^t_j(\hat{x}_j) - \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{j,a}} \hat{x}_{j,a} R^T_{\triangle_{j'}} \right\}$$

*Proof.* By definition, the cumulative regret $R^T_{\triangle_j}$ at time $T$ for decision point $j$ is:

$$\sum_{t=1}^T V^t_{\triangle_j} - \min_{\hat{x}_{\triangle_j}} \left\{ \sum_{t=1}^T \ell^t_j(\hat{x}_j) + \sum_{t=1}^T \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{j,a}} \hat{x}_{j,a} \hat{V}^t_{\triangle_{j'}}(\hat{x}_{\triangle_{j'}}) \right\}$$
$$= \sum_{t=1}^T V^t_{\triangle_j} - \min_{\hat{x}_j \in X_j} \left\{ \sum_{t=1}^T \ell^t_j(\hat{x}_j) + \sum_{a \in A_j} \sum_{j' \in \mathcal{C}_{j,a}} \hat{x}_{j,a} \min_{\hat{x}_{\triangle_{j'}}} \sum_{t=1}^T \hat{V}^t_{\triangle_{j'}}(\hat{x}_{\triangle_{j'}}) \right\}, \quad (5)$$

where the equalities follow first from expanding the definitions of $R^T_{\triangle_j}$ and $\hat{V}^t_{\triangle_j}(\hat{x}_{\triangle_j})$, and then using the fact that we can sequentially minimize first over choices at $j$ and then over choices for child information sets.

Now we can use (3) to get that (5) is equal to

$$= \sum_{t=1}^T V^t_{\triangle_j} - \min_{\hat{x}_j \in X_j} \left( \sum_{t=1}^T \hat{\ell}^t_j(\hat{x}_j) - \sum_{a \in A_j} \hat{x}_{j,a} \sum_{j' \in \mathcal{C}_{j,a}} R^t_{\triangle_{j'}} \right). \quad (6)$$

Since $V^t_{\triangle_j}$ already depends on $V^t_{\triangle_{j'}}$ for each child decision point $j'$ we have $V^t_{\triangle_j} = \hat{\ell}^t_j(x^t_j)$, where the equality follows by the definition of $\hat{\ell}^t_j$. Substituting this equality in (6) yields the statement. □

Theorem 1 justifies the introduction of the concept of *laminar regret* at each decision point $j \in \mathcal{J}$:

$$\hat{R}^T_j := \sum_{t=1}^T \hat{\ell}^t_j(x^t_j) - \min_{\hat{x}_j \in X_j} \sum_{t=1}^T \hat{\ell}^t_j(\hat{x}_j).$$

With this, we can write the cumulative subtree regret at decision point $j$ as a sum of laminar regret at $j$ plus a recurrence term for each child decision point. Applying this inductively gives the following theorem which tells us how one can apply regret minimization locally on laminar regrets in order to minimize regret in SDMs:

**Theorem 2.** *The cumulative regret on $X$ satisfies*

$$R^T \leq \max_{\hat{x} \in X} \sum_{j \in \mathcal{J}} \pi_j(\hat{x}) \hat{R}^T_j.$$

**Corollary 1.** *If each individual laminar regret $\hat{R}^T_j$ on each of the convex domains $X_j$ grows sublinearly, overall regret on $X$ grows sublinearly.*

Theorem 2 shows that overall regret can be minimized by minimizing each laminar regret separately. In particular, this means that if we have a regret minimizer for each decision point $j$ that can handle the structure of the convex set $X_j$ and the convex loss from (4), then we can apply those regret minimizers individually at each information set, and Theorem 2 guarantees that overall regret will be bounded by a weighted sum over those local regrets. For example, if each local regret minimizer has regret that grows at a particular sublinear rate, then the overall regret is also guaranteed to grow only at that sublinear rate.

Algorithm 1 gives pseudocode for a sample implementation of a regret minimizer based on LRD. The algorithm assumes that a regret minimizer $\mathcal{R}_j$ has been chosen for each decision point $j \in \mathcal{J}$ and has been initialized via a call to $\mathcal{R}_j$.INITIALIZE() before the algorithm starts. A concrete example of regret local regret minimizer $\mathcal{R}_j$ based on online gradient descent (OGD) is given later, together with its implementation, in Algorithm 2. The algorithm first iterates over all actions and associated child information sets. For each child information set $j'$ the subtree strategy $x^t_{\triangle_{j'}}$ is requested, then the subtree value $V^t_{\triangle_j}$ is computed, and finally it recurses by observing loss at $j'$. By observing loss at $j'$ after requesting $x^t_{\triangle_{j'}}$ we ensure that $V^t_{\triangle_j}$ is computed based on the strategy

**Algorithm 1** Regret minimizer based on LRD

```
1: function SdmObserveLoss($j, \{\ell_k^t\}_{k\in\mathcal{J}}$)
2:     for $a \in A_j$ do
3:         for $j' \in \mathcal{C}_{j,a}$ do
4:             $x^t_{\triangle_{j'}} \leftarrow$ SdmRecommend($j'$)
5:             $V^t_{\triangle_j} \leftarrow$ Value of strategy $x^t_{\triangle_{j'}}$ as defined in (2)
6:             SdmObserveLoss($j', \{\ell_k^t\}_{k\in\mathcal{J}}$)
7:     Construct $\hat{\ell}_j$ as defined in (4) using $\ell_j$, the strategies $x^t_{j'}$
       and the values $V^t_{\triangle_j}$ computed at lines 4 and 5 above.
8:     $\mathcal{R}_j$.ObserveLoss($\hat{\ell}_j$)
```

```
1: function SdmRecommend($j$)
2:     $x_j \leftarrow \mathcal{R}_j$.Recommend()
3:     for $a \in A_j$ do
4:         for $j' \in \mathcal{C}_{j,a}$ do
5:             $x_{\triangle_{j'}} \leftarrow$ SdmRecommend($j'$)
6:     return strategy $x_{\triangle_j}$ defined by $x_j$ at $j$ and $x_{\triangle_{j'}}$ for each
       subtree $\triangle_{j'}$.
```

prior to observing the loss at round $t$. Finally the local regret minimizer $\mathcal{R}_j$ observes the loss.

Our result gives an alternative proof of CFR. This is arguably simpler than existing proofs, because we show directly why regret over a sequential decision-making space decomposes into individual regret terms, as opposed to bounding terms in order to fit the CFR framework. Finally, our result also generalizes CFR to new settings: we show how CFR can be implemented on arbitrary convex subsets of simplexes and with convex losses rather than linear.

## Application Domains

Because we only need a finite sequential tree structure of the decision space, our framework captures a very broad class of SDM problems. In this section we describe how our framework can be applied to a number of prominent applications, such as POMDPs and EFGs. In general, our framework can be applied to any SDM problem where one or more agents are faced with a finite sequence of decisions that form a tree, such that agents always remember all past actions. The specific decision problem at each stage may depend on the past decisions as well as stochasticity. The fact that we require the decision space to be tree structured might seem limiting from the perspective of compactly representing the decision space. However, this has successfully been dealt with in applications by using state- or value-estimation techniques, rather than fully representing the original problem (Moravčík et al. 2017; Jin, Levine, and Keutzer 2018).

One example class of a single-agent decision problems that we can model is finite-horizon POMDPs where the history of states and actions is remembered. In that case, each decision point corresponds to a specific sequence of actions and observations made by the agent. This setting is reminiscent of the POMDP setting considered by Jin, Levine, and Keutzer (2018). This type of model can be used to model sequential medical treatment planning when combined with results on imperfect-recall abstraction (Lanctot et al. 2012; Chen and Bowling 2012; Kroer and Sandholm 2016a), and has potential applications in steering evolutionary adaptation (Sandholm 2015; Kroer and Sandholm 2016b). Our framework allows more general models for such problems via our generalization to convex decision points and convex losses; for example our framework could be used for regularized models. For instance in a medical settings, we may want to regularize the complexity of the treatment plan.

## Extensive-Form Games with Convex-Concave Saddle-Point Structure

In extensive-form game with perfect recall each player faces a sequential decision-making problem, of the type described in the previous section and in Figure 1. The set of next potential decision points $\mathcal{C}_{j,a}$ is based on observations of stochastic outcomes and actions taken by the other players.

Here, we will focus on two-player zero-sum EFGs with perfect recall, but with slightly more general utility structure than is usually considered. In particular, we assume that we are solving a convex-concave saddle-point problem of the following form:

$$\min_{x\in X}\max_{y\in Y}\left\{\mu(x)^\top A\mu(y) + d_1(\mu(x)) - d_2(\mu(y))\right\}, \qquad (7)$$

where $X$ is the SDM polytope for Player 1 and $Y$ is the SDM polytope of Player 2. Each $d_i$ is assumed to be a dilated convex function of the form

$$d_i(\mu(x)) = \sum_{j\in\mathcal{J}} \mu(x)_{p_j} d_j\left(\frac{\mu(x)_j}{\mu(x)_{p_j}}\right) = \sum_{j\in\mathcal{J}} \pi_j(x)\ell_j(x_j),$$

that is, in the form given in (1).

In standard zero-sum EFGs, the loss function for each player at each iteration $t$ is defined to be the negative payoff vector associated with the sequence-form strategy of the other player at that iteration; since we additionally allow a *regularization* term we also get a nonlinear convex term. More formally, at each iteration $t$, the loss functions $\ell^t_X : X \to \mathbb{R}$ and $\ell^t_Y : Y \to \mathbb{R}$ for player 1 and 2 respectively are defined as

$$\ell^t_X : x \mapsto \; -\mu(x)^\top A\mu(y^t) + d_1(x),$$
$$\ell^t_Y : y \mapsto \quad \mu(x^t)^\top A\mu(y) + d_2(y),$$

where $A$ is the sequence-form payoff matrix of the game (von Stengel 1996). Some simple algebra shows that $\ell^t_X$ and $\ell^t_Y$ are indeed separable (that is, they can be written in the form of Equation 1), where each decision-point-level loss $\ell^t_{j,X}$ and $\ell^t_{j,Y}$ is a convex function.

This choice of loss function is justified by the fact that the induced regret-minimizing dynamics for the two players lead to a convex-concave saddle-point problem. Specifically, assume the two players play the game $T$ times, accumulating regret after each iteration as in Figure 2.

A folk theorem explains the tight connection between low-regret strategies and approximate Nash equilibria. We will need a more general variant of that theorem generalized to (7). The convergence criterion we are interested in is the *saddle-point residual (or gap)* $\xi$ of $(\bar{x}, \bar{y})$, defined as

$$\xi = \max_{\hat{y}}\{d_1(\bar{x}) - d_2(\hat{y}) + \bar{x}^\top A\hat{y}\} - \min_{\hat{x}}\{d_1(\hat{x}) - d_2(\bar{y}) + \hat{x}^\top A\bar{y}\}.$$

We show that playing the average of a sequence of regret-minimizing strategies leads to a bounded saddle-point residual. This result is probably known, but it is unclear whether

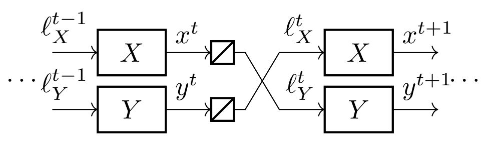

Figure 2: The flow of strategies and losses in regret minimization for games. The symbol ⊠ denotes computation/construction of the loss function.

it has been stated in the form here. A closely related form is used for averaged strategy iterates in a first-order method by Nemirovski (2004).

**Theorem 3.** *If the average regret accumulated on $X$ and $Y$ by the two sets of strategies $\{x_t\}_{t=1}^T$ and $\{y_t\}_{t=1}^T$ is $\epsilon_1$ and $\epsilon_2$, respectively, then any strategy profile $(\bar{x}, \bar{y})$ such that $\mu(\bar{x}) = \frac{1}{T}\sum_{t=1}^T \mu(x^t), \quad \mu(\bar{y}) = \frac{1}{T}\sum_{t=1}^T \mu(y^t)$ has a saddle-point residual bounded by $\epsilon_1 + \epsilon_2$.*

The above averaging is performed in the sequence-form space, which works because that space is also convex. After averaging we can easily compute $\bar{x}$ in linear time. Hence, by applying LRD to the decision spaces $X$ and $Y$, we converge to a small saddle-point residual. The fact that the averaging of the strategies is performed in sequence form explains why the traditional CFR presentation requires averaging with weights based on the player's reaches $\pi_j$ at each decision point $j$.

Theorem 3 shows that a Nash equilibrium can be computed by taking the uniform distribution over sequence-form strategy iterates. However, in the practical EFG-solving literature another approach called *linear averaging* has been popular Tammelin et al. (2015), especially for the CFR$^+$ algorithm. In linear averaging a weighted average strategy is constructed, where each strategy $\mu(x^t)$ is weighted by $t$. Tammelin et al. (2015) show that this is guaranteed to converge specifically when using the RM$^+$ regret minimizer. It would be interesting to prove when this works more generally. Here we make the simple observation that we can compute *both* averages, and simply use the one with better practical performance, even in settings where only the uniform average is guaranteed to converge.

**Quantal Response Equilibrium (QRE)** Ling, Fang, and Kolter (2018) show that a reduced-normal-form logit QRE can be expressed as the convex-concave saddle-point problem (7) where $d_1$ and $d_2$ are the (convex) dilated entropy functions usually used in first-order methods (FOMs) for solving EFGs (Hoda et al. 2010; Kroer et al. 2015; 2018). This saddle-point problem can be solved using FOMs, which would lead to fast convergence rate due to the strongly convex nature of the dilated entropy distance (Kroer et al. 2018). However, until now, no algorithms based on local regret minimization at each decision point have been known for this problem. Because the dilated entropy function separates into a sum over negative entropy terms at each decision point it can be incorporated as a convex loss in LRD. Combined with any regret-minimization algorithm that allows convex functions over the simplex, this leads to the first regret-minimization algorithm for computing (reduced-normal-form) logit QREs.

**Perturbed EFGs and Equilibrium Refinement** Equilibrium refinements are Nash equilibria with additional important rationality properties. Such equilibria have rarely been used in practice due to scalability issues. Recently, fast algorithms for computing approximate refinements were introduced (Kroer, Farina, and Sandholm 2017; Farina, Kroer, and Sandholm 2017). Theorem 1 gives a new tool for constructing such methods: it immediately implies correctness of the method of Farina, Kroer, and Sandholm (2017), while also allowing new types of refinements and regret minimizers.

## Erratum about Alternation in CFR$^+$

Several tweaks to speed up the convergence of CFR have been proposed. The state of the art is CFR$^+$ (Tammelin et al. 2015). CFR$^+$ consists of three tweaks: the RM$^+$ regret minimizer, linear averaging, and alternation. RM$^+$ can be applied in our setting as well; it is simply an alternative regret minimizer for linear losses over a simplex. We described linear averaging earlier in this paper. Finally, *alternation* is the idea that at iteration $t$, we provide Player 2 with the utility vector associated with the *current* iterate of Player 1, rather than that of the previous iteration, as is normally done in regret minimization. Figure 3 illustrates how this works, in contrast with Figure 2 which shows the usual flow. Tammelin et al. (2015) state that they prove convergence of CFR$^+$; however their proof relies on the folk theorem that links Nash equilibrium and regret. That folk theorem is only proven for the case where no alternation is applied. We show below that the theorem does not hold with alternation!

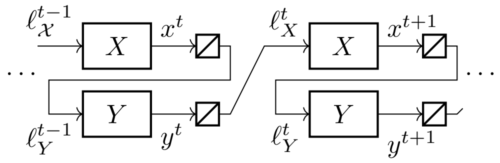

Figure 3: The alternation method for CFR in games. The loss at iteration $t$ for $y$ is computed with $x^t$. The symbol ⊠ denotes computation/construction of the loss function.

**Observation 1.** *Let the action spaces for the players be $X = Y = [0,1]$, and let $\ell_X^t : x \mapsto x \cdot y^t$, $\ell_Y^t : y \mapsto -y \cdot x^{t+1}$ be bilinear loss functions (the superscript $t+1$ comes from the use of alternation—see Figure 3). Consider the sequence of strategies $x^t = t \mod 2, y^t = (t+1) \mod 2$. A simple check reveals that after $2T$ iterations, the average regrets of the two players are both 0. Yet, the average strategies $\bar{x}^{2T} = \bar{y}^{2T} = 0.5$ do not converge to a saddle point of $xy$.*

This observation should be seen as more of a theoretical issue than practical; alternation has been used extensively in practice, and the problem that we show does not seem to come up for nondegenerate iterates (at least for CFR$^+$; it may explain some erratic behavior that we have anecdotally observed with other regret minimization algorithms when using alternation). However, because of this issue, it is now unknown whether the CFR$^+$ algorithm is sound or merely a heuristic that might converge slower or not even converge to an equilibrium at all. By our counterexample, if CFR$^+$ is sound, the proof will need to go through a different theorem than the folk theorem, and may need to leverage specific structure of the regret minimizer, or may even need constraints on the initialization (for example, starting from the interior of the strategy space).

## Experiments

We conducted experiments on two EFG settings. The first game is Leduc 5 poker (Southey et al. 2005), a standard benchmark in imperfect-information game solving. There is a deck consisting of 5 unique cards with 2 copies of each. There are two rounds. In the first round, each player places an ante of 1 in the pot and receives a single private card. A round of betting then takes place with a two-bet maximum, with Player 1 going first. A public shared card is then dealt face up and another round of betting takes place. Again, Player 1 goes first, and there is a two-bet maximum. If one of the players has a pair with the public card, that player wins. Otherwise, the player with the higher card wins. All bets in the first round are 1, while all bets in the second round are 2.

The second game is a variant of Goofspiel (Ross 1971), a bidding game where each player has a hand of cards numbered 1 to $N$. A third stack of $N$ cards is shuffled and used as prizes: each turn a prize card is revealed, and the players each choose a private card to bid on the prize, with the high card winning, the value of the prize card is split evenly on a tie. After $N$ turns all prizes have been dealt out and the payoff to each player is the sum of prize cards that they win. We used $N = 4$ in our experiments.

We conducted three types of experiment, as described in the following subsections, respectively. Our code is parallel and the experiments were conducted on a machine with 8 cores, but the results are not about timing, so they would be identical even on a single core. We used alternation (Figure 3) only in the implementation of CFR$^+$; all other algorithms use the flow of Figure 2.

### Entropic Regularization Applied to Finding a Quantal Response Equilibrium

First we investigated a setting where no previous regret-minimization algorithms based on minimizing regret locally existed: the computation of logit QREs via LRM and our more general convex losses. Ling, Fang, and Kolter (2018) used Newton's method for this setting, but, as with Nash equilibrium, second-order algorithms do not scale to large games (this is why CFR$^+$ has been so successful for creating human-level poker AIs). We compared how quickly we could compute a logit QRE[1] compared to how quickly a Nash equilibrium could be computed, in order to understand the size of games for which we can find a QRE with our approach. To do this, we ran LRM with online gradient descent (OGD) at each decision point. Because OGD is not guaranteed to stay within the simplex at each iteration, we needed to project onto the simplex. There are multiple fast ways of doing so, for example, a binary search (Duchi et al. 2008). The results are in Figure 4. LRM performs extremely well. In Goofspiel it converges vastly faster than CFR$^+$, and in Leduc 5 it converges at a rate comparable to CFR$^+$ and eventually becomes faster. This shows that logit QRE computation via LRM likely scales to extremely large EFGs, such as real-world-sized poker games (since CFR$^+$ is known to scale to such games).

[1]For simplicity, we computed a logit QRE with parameter $\lambda = 1$. However, our construction can easily be extended to handle any positive value of $\lambda$.

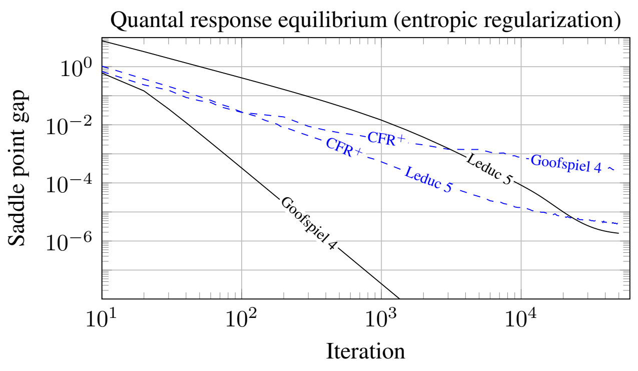

Figure 4: The QRE saddle-point gap as a function of the number of iterations for each game. The convergence rates of CFR$^+$ for Nash equilibrium is shown for reference.

### Quadratic Regularization: Using Strong Convexity to Converge Faster

In the second set of experiments we investigated the speed of convergence for solving $\ell_2$-regularized EFGs. In other words, at each decision point, we added to the loss a term $\alpha\|z\|^2$, where $z$ is the strategy at that decision point. Again we included the convergence rate of standard CFR and CFR$^+$ for Nash equilibrium computation as a benchmark. The results for Leduc 5 are in Figure 5. Solving the regularized game is significantly faster than computing a Nash equilibrium via CFR$^+$, except for extremely small amounts of regularization.

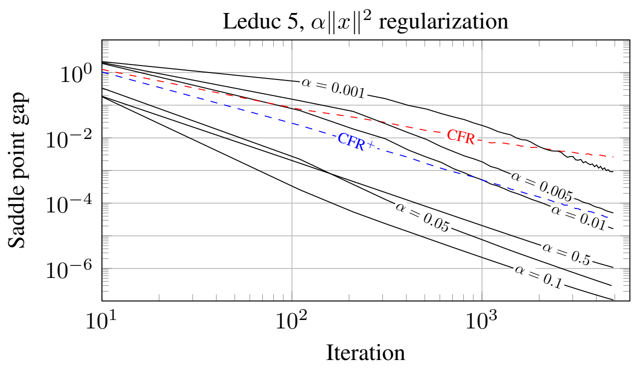

Figure 5: The saddle-point gap as a function of the number of iterations for $\ell_2$-regularized Leduc 5 for varying amounts of regularization $\alpha$. The convergence rates of CFR$^+$ for Nash equilibrium are shown for reference (dashed curves).

The best speed is associated with an interior value of $\alpha$. This is because there is a tradeoff between stronger convexity (achieved by larger $\alpha$) and smaller loss function values (achieved by smaller $\alpha$). Algorithm 2 complements Algorithm 1 by providing pseudocode for the local regret minimizers $\mathcal{R}_j$ that implement the quadratic regularization we just discussed.

**Algorithm 2** Local regret minimizers used in the quadratic regularization experiments

```
1: function $\mathcal{R}_j$.INITIALIZE( )
2:     self.$x \leftarrow$ any strategy in $X_j = \Delta^{n_j}$ (we used $\mathbf{1}/n_j$)

1: function $\mathcal{R}_j$.OBSERVELOSS($\ell$)
2:     $\tilde{x} \leftarrow$ self.$x - \frac{1}{2\alpha\sqrt{t}}\nabla\ell(x^t)$
3:     self.$x \leftarrow$ project $\tilde{x}$ onto $X_j = \Delta^{n_j}$

1: function $\mathcal{R}_j$.RECOMMEND( )
2:     return self.$x$
```

## Opponent Exploitation While Staying Close to a Safe Strategy

In the third set of experiments we investigated the performance of LRM in a single-agent-learning setting: learning how to exploit a static opponent where we observe repeated samples from their strategy. We considered a setting where the exploiter wishes to maximally exploit subject to staying near a pre-computed Nash equilibrium in order to avoid opening herself up to exploitability (Ganzfried and Sandholm 2011). We modeled this in a new way: as a regularized online SDM problem, where the loss for the exploiter is

$$\ell^t : x \mapsto -\mu(x)^\top A\mu(y^t) + \alpha D(x\|x^{NE}) \qquad (8)$$

where $y^t$ is the $t$'th observation from the opponent's strategy and $D(x\|x^{NE})$ is the dilated $\ell_2$-based Bregman divergence between the NE strategy $x^{NE}$ and $x$. The opponent's suboptimal strategy was computed by running CFR$^+$ until a gap of 0.1 was reached. We stopped the training of the exploiter after 5000 iterations or when an average regret of 0.0005 was reached, whichever happened first. Figure 6 shows the results. The "utility increase" line shows how much the agent gains by moving away from the Nash equilibrium and towards an exploitative strategy, while the "exploitability" shows to what extent the agent thereby opens herself up to being exploited by her nemesis, that is, a best-response strategy of the opponent. We see that this model can indeed be used as a scalable proxy for trading off exploitation and exploitability by varying $\alpha$.

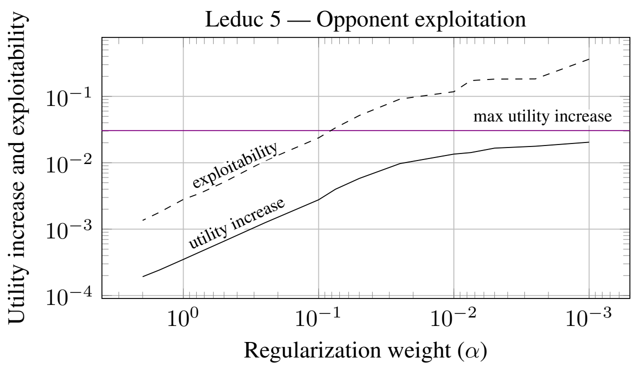

Figure 6: The utility increase from exploitation, and the resulting increase in the agent's exploitability, as a function of decreasing penalization on distance from Nash equilibrium in (8). The horizontal line shows the value of best responding to (that is, maximally exploiting) the opponent strategy.

## Conclusions and Future Research

We presented laminar regret decomposition (LRD), a new decomposition of the regret associated with a sequential action space into regrets associated with individual decision points. We developed our technique for general compact convex sets and convex losses at each decision point, thus providing a generalization of CFR beyond simplex decision points and linear loss. We then showed that our results lead to a new class of regret-minimization algorithms that solve sequential decision making problems by minimizing regret locally at each decision point. Although more general, our proof also provides a new perspective on the CFR algorithm in terms of our regret decomposition, and we explained the need for weighting by reach as a consequence of averaging in sequence form. We then showed that our approach can be used to compute regularized equilibria as well as Nash equilibrium refinements, and gave the first regret-minimization algorithm for computing (reduced-normal-form) quantal response equilibrium (QRE) (based on local regrets). We showed experimentally that our framework, laminar regret minimization (LRM), can be used to compute QREs in extensive-form games (EFGs) at a rate that is comparable to Nash equilibrium finding using CFR$^+$, thus yielding the first algorithm for computing QREs in large EFGs. We also showed that $\ell_2$-regularized equilibrium can be computed very quickly with out method. Finally, we showed how our approach can be used as a new approach to opponent exploitation, and to control the tradeoff between exploitation and exploitability.

Our algorithms that use (quadratic) regularization converge orders of magnitude faster than the fastest equilibrium-finding algorithm, CFR$^+$. This is despite the fact that we simply used an unoptimized off-the-shelf implementation of online gradient descent. The reason is that the approach capitalizes on the strong convexity introduced by the regularization. The algorithms do not converge to a Nash equilibrium, but to a regularized equilibrium. Their speed suggests a promising new family of equilibrium-finding algorithms. One could start with more regularization, and gradually decrease the regularization to achieve convergence to Nash equilibrium. One would need to set the rate of unsmoothing appropriately in order to achieve convergence and to also get the speed benefit from the more smoothed earlier iterations. Equilibrium-finding algorithms based on unsmoothing already exist (Nesterov 2005; Hoda et al. 2010; Kroer et al. 2018), but ours would be quite different. First, ours are based on a regret-minimization framework instead of first-order methods. Second, ours leverage our regret decomposition so they can operate at the decision-point level, thereby having a better understanding of the problem structure and allowing different algorithmic strategies to be used in different parts of the search space, etc.

Another opportunity presented by the strong convexity of the laminar losses is to use accelerated methods. This could lead to a faster theoretical convergence rate than that of CFR$^+$ for Nash equilibrium, and could likely be exploited in practice also. Furthermore, we used online gradient descent for convenience, but one could likely employ a projection-free algorithm and thus have the computational cost at each decision point be the same as for CFR$^+$ while having a faster convergence rate.

Another future direction is to extend the very recent improvements to CFR and CFR$^+$ (Brown and Sandholm 2019), which were developed in parallel with this paper, to our setting. This may involve two directions: 1) generalizing those results to our more general settings, and 2) using those techniques to enhance the local regret minimizers at each decision point in our framework.

It would also be interesting to find further applications where our new types of decision spaces and loss functions can be used.

## Acknowledgments

This material is based on work supported by the National Science Foundation under grants IIS-1718457, IIS-1617590, and CCF-1733556, and the ARO under award W911NF-17-1-0082. Christian Kroer was supported by a Facebook Fellowship.

## References

Blackwell, D. 1956. An analog of the minmax theorem for vector payoffs. *Pacific Journal of Mathematics*.

Bowling, M.; Burch, N.; Johanson, M.; and Tammelin, O. 2015. Heads-up limit hold'em poker is solved. *Science*.

Brown, N., and Sandholm, T. 2017a. Safe and nested subgame solving for imperfect-information games. In *NIPS*.

Brown, N., and Sandholm, T. 2017b. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*.

Brown, N., and Sandholm, T. 2019. Solving imperfect-information games via discounted regret minimization. In *AAAI*.

Brown, N.; Kroer, C.; and Sandholm, T. 2017. Dynamic thresholding and pruning for regret minimization. In *AAAI*.

Chen, K., and Bowling, M. 2012. Tractable objectives for robust policy optimization. In *NIPS*.

Duchi, J.; Shalev-Shwartz, S.; Singer, Y.; and Chandra, T. 2008. Efficient projections onto the l1-ball for learning in high dimensions. In *ICML*. ACM.

Farina, G., and Gatti, N. 2017. Extensive-form perfect equilibrium computation in two-player games. In *AAAI*.

Farina, G.; Kroer, C.; and Sandholm, T. 2017. Regret minimization in behaviorally-constrained zero-sum games. In *ICML*.

Ganzfried, S., and Sandholm, T. 2011. Game theory-based opponent modeling in large imperfect-information games. In *AAMAS*.

Hart, S., and Mas-Colell, A. 2000. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*.

Hazan, E., and Kale, S. 2010. Extracting certainty from uncertainty: Regret bounded by variation in costs. *Machine learning*.

Hazan, E. 2016. Introduction to online convex optimization. *Foundations and Trends in Optimization*.

Hoda, S.; Gilpin, A.; Peña, J.; and Sandholm, T. 2010. Smoothing techniques for computing Nash equilibria of sequential games. *Mathematics of Operations Research*.

Jin, P. H.; Levine, S.; and Keutzer, K. 2018. Regret minimization for partially observable deep reinforcement learning. In *ICML*.

Kroer, C., and Sandholm, T. 2016a. Imperfect-recall abstractions with bounds in games. In *EC*.

Kroer, C., and Sandholm, T. 2016b. Sequential planning for steering immune system adaptation. In *IJCAI*.

Kroer, C.; Waugh, K.; Kılınç-Karzan, F.; and Sandholm, T. 2015. Faster first-order methods for extensive-form game solving. In *EC*.

Kroer, C.; Waugh, K.; Kılınç-Karzan, F.; and Sandholm, T. 2018. Faster algorithms for extensive-form game solving via improved smoothing functions. *Mathematical Programming*.

Kroer, C.; Farina, G.; and Sandholm, T. 2017. Smoothing method for approximate extensive-form perfect equilibrium. In *IJCAI*.

Kuhn, H. W. 1950. A simplified two-person poker. In *Contributions to the Theory of Games*. Princeton University Press.

Lanctot, M.; Gibson, R.; Burch, N.; Zinkevich, M.; and Bowling, M. 2012. No-regret learning in extensive-form games with imperfect recall. In *ICML*.

Ling, C. K.; Fang, F.; and Kolter, J. Z. 2018. What game are we playing? End-to-end learning in normal and extensive form games. In *IJCAI*.

Moravčík, M.; Schmid, M.; Burch, N.; Lisý, V.; Morrill, D.; Bard, N.; Davis, T.; Waugh, K.; Johanson, M.; and Bowling, M. 2017. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*.

Nemirovski, A. 2004. Prox-method with rate of convergence O(1/t) for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*.

Nesterov, Y. 2005. Excessive gap technique in nonsmooth convex minimization. *SIAM Journal of Optimization*.

Ross, S. M. 1971. Goofspiel–the game of pure strategy. *Journal of Applied Probability*.

Sandholm, T. 2015. Steering evolution strategically: Computational game theory and opponent exploitation for treatment planning, drug design, and synthetic biology. In *AAAI*. Senior Member Track.

Southey, F.; Bowling, M.; Larson, B.; Piccione, C.; Burch, N.; Billings, D.; and Rayner, C. 2005. Bayes' bluff: Opponent modelling in poker. In *UAI*.

Tammelin, O.; Burch, N.; Johanson, M.; and Bowling, M. 2015. Solving heads-up limit Texas hold'em. In *IJCAI*.

von Stengel, B. 1996. Efficient computation of behavior strategies. *Games and Economic Behavior*.

Zinkevich, M.; Bowling, M.; Johanson, M.; and Piccione, C. 2007. Regret minimization in games with incomplete information. In *NIPS*.

Zinkevich, M. 2003. Online convex programming and generalized infinitesimal gradient ascent. In *ICML*.