# Computing Approximate Equilibria in Sequential Adversarial Games by Exploitability Descent

**Edward Lockhart**[1] , **Marc Lanctot**[1] , **Julien Pérolat**[1] ,
**Jean-Baptiste Lespiau**[1] , **Dustin Morrill**[1,2] , **Finbarr Timbers**[1] , **Karl Tuyls**[1]
[1]DeepMind
[2]University of Alberta, Edmonton, Canada
{locked, lanctot, perolat, jblespiau, dmorrill, finbarrtimbers, karltuyls}@google.com,
morrill@ualberta.ca

## Abstract

In this paper, we present *exploitability descent*, a new algorithm to compute approximate equilibria in two-player zero-sum extensive-form games with imperfect information, by direct policy optimization against worst-case opponents. We prove that when following this optimization, the exploitability of a player's strategy converges asymptotically to zero, and hence when both players employ this optimization, the joint policies converge to a Nash equilibrium. Unlike fictitious play (XFP) and counterfactual regret minimization (CFR), our convergence result pertains to the policies being optimized rather than the average policies. Our experiments demonstrate convergence rates comparable to XFP and CFR in four benchmark games in the tabular case. Using function approximation, we find that our algorithm outperforms the tabular version in two of the games, which, to the best of our knowledge, is the first such result in imperfect information games among this class of algorithms.

## 1 Introduction

Extensive-form games model sequential interactions between multiple agents, each of which maximize their own utility. Classic examples are perfect information games (e.g. chess and Go), which have served as milestones for measuring the progress of artificial intelligence [Campbell *et al.*, 2002; Silver *et al.*, 2016]. When there are simultaneous moves, such as in Markov games, the players may need stochastic policies to guarantee their worst-case expected utility, and must use linear programming at each state for value back-ups. Computing policies for imperfect information games is much more difficult: no Bellman operator exists, so approximate dynamic programming is not applicable; exact equilibrium solutions can be found by sequence-form linear programming [Koller *et al.*, 1994; Shoham and Leyton-Brown, 2009], but these techniques do not scale to very large games.

The challenge domain for imperfect information has been computer Poker, which has driven much of the progress in computational approaches to equilibrium-finding [Rubin and Watson, 2011]. While there are gradient descent techniques that can find an $\epsilon$-Nash equilibrium in $O(\frac{1}{\epsilon})$ iterations [Hoda *et al.*, 2007], the dominant technique has been counterfactual regret minimization (CFR) [Zinkevich *et al.*, 2008]. Based on CFR, recent techniques have solved heads-up limit Texas Hold'em [Bowling *et al.*, 2015] and beat human professionals in no-limit Texas Hold'em [Moravčík *et al.*, 2017; Brown and Sandholm, 2017].

Other techniques have emerged in recent years, based first on fictitious play (XFP) [Heinrich *et al.*, 2015], and generalized to double oracle and any meta-game solver over sets of policies [Lanctot *et al.*, 2017]. Both require a subroutine that computes a best response (an "oracle"). Here, reinforcement learning can be used to compute approximate oracles, and function approximation can be used to generalize over the state space without domain-specific abstraction mechanisms. Hence, deep neural networks can trained from zero knowledge as in AlphaZero [Silver *et al.*, 2018]. Policy gradient techniques are also compatible with function approximation in this setting [Srinivasan *et al.*, 2018], but may require many iterations to converge. Combining data buffers with CFR using regression to predict regrets has also shown promise in medium-sized poker variants [Waugh *et al.*, 2015; Brown *et al.*, 2019].

In this paper, we introduce a new algorithm for computing approximate Nash equilibria. Like XFP, best responses are computed at each iteration. Unlike XFP, players optimize their policies directly against their worst-case opponent. When using tabular policies and $\ell_2$ projections after policy updates, the sequence of policies will contain an $\epsilon$-Nash equilibrium, unlike CFR and XFP that only converge-in-average. Our algorithm works well with function approximation, as the problem can be expressed directly as a policy gradient optimization. Our experiments show convergence rates comparable to XFP and CFR in the tabular setting, exhibit generalization over the state space using neural networks in four different games.

At the time of original submission, we were unaware of a similar algorithm recently presented at the Deep RL Workshop NeurIPS 2018: Self-Play Against a Best Response (SPAR) [Tang *et al.*, 2018]. The work we present in this paper was done independently. In this paper, we provide convergence guarantees, as well as results in both the tabular and neural network cases. We do so on four benchmark games (both commonly used poker benchmarks used in [Tang *et al.*,

2018] and two additional games), whereas results of SPAR focus on the sample-based setting not covered in this paper.

## 2 Background and Terminology

An **extensive-form game** describes a sequential interaction between **players** $i \in \{1, 2, \cdots, n\} \cup \{c\}$, where $c$ is considered a special player called **chance** with a fixed stochastic policy that determines the transition probabilities given states and actions. We will often use $-i$ to refer to all the opponents of $i$. In this paper, we focus on the $n = 2$ player setting.

The game starts in the empty history $h = \emptyset$. On each turn, a player $i$ chooses an action $a \in \mathcal{A}_i$, changing the history to $h' = ha$. Here $h$ is called a prefix history of $h'$, denoted $h \sqsubset h'$. The full history is sometimes also called a *ground* state because it uniquely identifies the true state, since chance's actions are included. In poker, for example, a ground state would include all the players' private cards. We define an **information state** $s \in \mathcal{S}$ for player $i$ as the state as perceived by an agent which is consistent with its observations. Formally, each $s$ is a set of histories, specifically $h, h' \in s \Leftrightarrow$ the sequence of of player $i$'s observations along $h$ and $h'$ are equal. In poker, an information state groups together all the histories that differ only in the private cards of $-i$. Denote $\mathcal{Z}$ the set of terminal histories, each corresponding to the end of a game, and a utility to each player $u_i(z)$ for $z \in \mathcal{Z}$. We also define $\tau(s)$ as the player whose turn it is at $s$, and $\mathcal{Z}(h)$ the subset of terminal histories that share $h$ as a prefix.

Since players cannot observe the ground state $h$, policies are defined as $\pi_i : \mathcal{S}_i \to \Delta(\mathcal{A}_i)$, where $\Delta(\mathcal{A}_i)$ is the set of probability distributions over $\mathcal{A}_i$. Each player tries to maximize their expected utility given the initial starting history $\emptyset$. We assume finite games, so every history $h$ is bounded in length. The expected value of a joint policy $\boldsymbol{\pi}$ (all players' policies) for player $i$ is defined as

$$v_{i,\boldsymbol{\pi}} = \mathbb{E}_{z\sim\boldsymbol{\pi}}[u_i(z)], \tag{1}$$

where the terminal histories $z \in \mathcal{Z}$ are composed of actions drawn from the joint policy. We also define state-action values for joint policies. The value $q_{i,\boldsymbol{\pi}}(s, a)$ represents the expected return starting at state $s$, taking action $a$, and playing $\boldsymbol{\pi}$:

$$\begin{aligned} q_{i,\boldsymbol{\pi}}(s, a) =& \mathbb{E}_{z\sim\boldsymbol{\pi}}[u_i(z) \mid h \in s, ha \sqsubseteq z] \\ =& \sum_{h\in s, z\in\mathcal{Z}(h)} Pr(h|s) u_i(z) \\ =& \frac{\sum_{h\in s} \eta_{\boldsymbol{\pi}}(h) q_{i,\boldsymbol{\pi}}(h, a)}{\sum_{h\in s} \eta_{\boldsymbol{\pi}}(h)}, \end{aligned} \tag{2}$$

where $q_{i,\boldsymbol{\pi}}(h, a) = \mathbb{E}_{z\sim\boldsymbol{\pi}}[u_i(z) \mid ha \sqsubseteq z] = \sum_{h\in s, z\in\mathcal{Z}(h)} \eta_{\boldsymbol{\pi}}(h, z) u_i(z)$ is the expected utility of the ground state-action pair $(h, a)$, and $\eta_{\boldsymbol{\pi}}(h)$ is the probability of reaching $h$ under the policy $\boldsymbol{\pi}$. We make the common assumption that players have **perfect recall**, i.e. they do not forget anything they have observed while playing. Under perfect recall, the distribution of the states can be obtained only from the opponents' policies using Bayes' rule (see [Srinivasan *et al.*, 2018, Section 3.2]).

Each player $i$ tries to find a policy that maximizes their own value $v_{i,\boldsymbol{\pi}}$. However, this is difficult to do independently since the value depends on the joint policy, not just player $i$'s policy. A **best response** policy for player $i$ is defined to be $b_i(\boldsymbol{\pi}_{-i}) \in \mathrm{BR}(\boldsymbol{\pi}_{-i}) = \{\pi_i \mid v_{i,(\pi_i,\boldsymbol{\pi}_{-i})} = \max_{\pi'_i} v_{i,(\pi'_i,\boldsymbol{\pi}_{-i})}\}$. Given a joint policy $\boldsymbol{\pi}$, the **exploitability** of a policy $\boldsymbol{\pi}_{-i}$ is how much the other player could gain if they switched to a best response: $\delta_i(\boldsymbol{\pi}) = \max_{\pi'_i} v_{i,(\pi'_i,\boldsymbol{\pi}_{-i})} - v_{i,\boldsymbol{\pi}}$. In two-player zero-sum games, an $\epsilon$-minmax (or $\epsilon$-Nash equilibrium) policy is one where $\max_i \delta_i(\boldsymbol{\pi}) \leq \epsilon$. A Nash equilibrium is achieved when $\epsilon = 0$. A common metric to measure the distance to Nash is $\textsc{NashConv}(\boldsymbol{\pi}) = \sum_i \delta_i(\boldsymbol{\pi})$.

### 2.1 Extensive-Form Fictitious Play (XFP)

Extensive-form fictitious play (XFP) is equivalent to standard fictitious play, except that it operates in the extensive-form representation of the game [Heinrich *et al.*, 2015]. In fictitious play, the joint policy is initialized arbitrarily (e.g. uniform random distribution at each information state), and players learn by aggregating best response policies. The extensive-form

**Algorithm 1:** Fictitious Play

**input** : $\boldsymbol{\pi}^0$ — initial joint policy
**for** $t \in \{1, 2, \cdots\}$ **do**
  **for** $i \in \{1, \ldots, n\}$ **do**
    Compute a best response $\boldsymbol{b}_i^t(\boldsymbol{\pi}_{-i}^{t-1})$
    Update average policy $\boldsymbol{\pi}^t$ to include $\boldsymbol{b}_i^t$

version, XFP, requires game-tree traversals to compute the best responses and specific update rules that account for the reach probabilities to ensure that the updates are equivalent to the classical algorithm, as described in [Heinrich *et al.*, 2015, Section 3]. Fictitious play converges to a Nash equilibrium asymptotically in two-player zero-sum games. Sample-based approximations to the best response step have also been developed [Heinrich *et al.*, 2015] as well as function approximation methods to both steps [Heinrich and Silver, 2016]. Both steps have also been generalized to other best response algorithms and meta-strategy combinations [Lanctot *et al.*, 2017].

### 2.2 Counterfactual Regret Minimization (CFR)

CFR decomposes the full regret computation over the tree into per information-state regret tables and updates [Zinkevich *et al.*, 2008]. Each iteration traverses the tree to compute the local values and regrets, updating cumulative regret and average policy tables, using a local regret minimizer to derive the current policies at each information state.

The quantities of interest are **counterfactual values**, which are similar to $Q$-values, but differ in that they weigh only the opponent's reach probabilities, and are not normalized. Formally, let $\eta_{-i,\boldsymbol{\pi}}(h)$ be *only the opponents' contributions* to the probability of reaching $h$ under $\boldsymbol{\pi}$. Then, similarly to equation 2, we define counterfactual values: $q^c_{i,\boldsymbol{\pi}}(s, a) = \sum_{h\in s} \eta_{-i,\boldsymbol{\pi}}(h) q_{i,\boldsymbol{\pi}}(h, a)$, and $v^c_{i,\boldsymbol{\pi}}(s) = \sum_{a\in\mathcal{A}_i} \pi_i(s, a) q^c_{i,\boldsymbol{\pi}}(s, a)$. On each iteration $k$, with a joint policy $\boldsymbol{\pi}^k$, CFR computes a counterfactual regret $r(s, a) = q^c_{i,\boldsymbol{\pi}^k}(s, a) - v^c_{i,\boldsymbol{\pi}^k}(s)$ for all information states $s$, and a new policy from the cumulative regrets of $(s, a)$ over the iterations using regret-matching [Hart and Mas-Colell, 2000].

The average policies converge to an $\epsilon$-Nash equilibrium in $O(|\mathcal{S}_i|^2|\mathcal{A}_i|/\epsilon^2)$ iterations.

**CFR Versus a Best Response Oracle (CFR-BR)**

Instead of both players employing CFR (CFR-vs-CFR), each player can use CFR versus their worst-case (best response) opponent, i.e. simultaneously running CFR-vs-BR and BR-vs-CFR. This is the main idea behind counterfactual regret minimization against a best response (CFR-BR) algorithm [Johanson *et al.*, 2012]. The combined average policies of the CFR players is also guaranteed to converge to an $\epsilon$-Nash equilibrium. In fact, the current strategies also converge with high probability. Our convergence analyses are based on CFR-BR, showing that a policy gradient versus a best responder also converges to an $\epsilon$-Nash equilibrium.

### 2.3 Policy Gradients in Games

We consider policies $\boldsymbol{\pi_\theta} = (\pi_{i,\theta_i})_i$ each policy are parameterized by a vector of parameter $(\theta_i)_i = \boldsymbol{\theta}$. Using the likelihood ratio method, the gradient of $v_{i,\boldsymbol{\pi_\theta}}$ with respect to the vector of parameters $\theta_i$ is:

$$\nabla_{\theta_i} v_{i,\boldsymbol{\pi_\theta}} = \sum_{s \in S_i} \left( \sum_{h \in s} \eta_{\boldsymbol{\pi}}(h) \right) \sum_a \nabla_{\theta_i} \pi_{i,\theta_i}(s,a) q_{i,\boldsymbol{\pi_\theta}}(s,a) \tag{3}$$

This result can be seen as an extension of the policy gradient Theorem [Sutton *et al.*, 2000; Glynn and L'ecuyer, 1995; Williams, 1992; Baxter and Bartlett, 2001] to imperfect information games and has been used under several forms: for a detailed derivation, see [Srinivasan *et al.*, 2018, Appendix D].

The critic ($q_{i,\boldsymbol{\pi_\theta}}$) can be estimated in many ways (Monte Carlo Return [Williams, 1992] or using a critic for instance in [Srinivasan *et al.*, 2018] in the context of games. Then:

$$\theta_i \quad \leftarrow \quad \theta_i + \alpha \sum_{l=0}^{K} \mathbb{1}_{i=\tau(s_l)} \sum_a \nabla_{\theta_i} \pi_{i,\theta_i}(s_l,a) \hat{q}_{i,\boldsymbol{\pi_\theta}}(s_l,a),$$

where $\alpha$ is the learning rate used by the algorithm and $\hat{q}_{i,\boldsymbol{\pi_\theta}}(s_l,a)$ is the estimation of the return used.

## 3 Exploitability Descent

Exploitability Descent (ED) follows the basic form of the classic convex-concave optimization problem for solving matrix games [Gale *et al.*, 1951; Boyd and Vandenberghe, 2004]. Conceptually, the algorithm is uncomplicated and shares the outline of fictitious play: on each iteration, there are two steps that occur for each player. The first step is identical to fictitious play: compute the best response to each player's policy. The second step then performs gradient ascent on the policy to increase each player's utility against the respective best responder (aiming to decrease each player's exploitability). The change in the second step is important for two reasons. First, it leads to a convergence of the policies that are being optimized without having to compute an explicit average policy, which is complex in the sequential setting. Secondly, the policies can now be easily parameterized (i.e. using e.g. deep neural networks) and trained using policy gradient ascent without storing a large buffer of previous data.

**Algorithm 2:** Exploitability Descent (ED)

```
   input  : π^0 — initial joint policy
1  for t ∈ {1, 2, ···} do
2      for i ∈ {1, ··· , n} do
3          Compute a best response b_i^t(π_{-i}^{t-1})
4      for i ∈ {1, ··· , n}, s ∈ S_i do
5          Define b_{-i}^t = {b_j^t}_{j≠i}
6          Let q^b(s) = ValuesVsBRs(π_i^{t-1}(s), b_{-i}^t)
7          π_i^t(s) = GradAscent(π_i^{t-1}(s), α^t, q^b(s))
```

The general algorithm is outlined in Algorithm 2, where $\alpha^t$ the learning rate on iteration $t$. Two steps (lines 6 and 7) are intentionally unspecified: we will show properties for two specific instantiations of this general ED algorithm. The quantity $\mathbf{q}^{\boldsymbol{b}}$ refers to a set of expected values for player $i = \tau(s)$, one for each action at $s$ using $\boldsymbol{\pi}_i^{t-1}$ against a set of individual best responses. The GradientAscent update step unspecified for now as we will describe several forms, but the main idea is to increase/decrease the probability of higher/lower utility actions via the gradients of the value functions, and project back to the space of policies.

### 3.1 Tabular ED with $q$-Values and $\ell_2$ Projection

For a vector of $|\mathcal{A}|$ real numbers $\boldsymbol{\theta}_s$, define the **simplex** as $\Delta_s = \{\boldsymbol{\theta}_{s,a} \mid \boldsymbol{\theta}_s \geq \mathbf{0}, \sum_a \theta_{s,a} = 1\}$, and the $\ell_2$ projection as $\Pi_{\ell_2}(\boldsymbol{\theta}_s) = \text{argmin}_{\boldsymbol{\theta}'_s \in \Delta_s} ||\boldsymbol{\theta}'_s - \boldsymbol{\theta}_s||_2$.

Let $\boldsymbol{\pi_\theta}$ be a joint policy parameterized by $\boldsymbol{\theta}$, and $\boldsymbol{\pi}_{\boldsymbol{\theta}_i}$ refer to the portion of player $i$'s parameters (*i.e.* in tabular form $\{\boldsymbol{\theta}_s \mid \tau(s) = i\}$). Here each parameter is a probability of an action at a particular state: $\theta_{s,a} = \pi_{\boldsymbol{\theta}}(s,a)$. We refer to TabularED($q$, $\ell_2$) as an instance of exploitability descent with

$$\mathbf{q}^{\boldsymbol{b}}(s) = \{q_{i,(\boldsymbol{\pi}_\theta^{t-1}, \boldsymbol{b}_{-i}^t)}(s,a)\}_{a \in \mathcal{A}}, \tag{4}$$

and the policy gradient ascent update defined to be

$$\begin{aligned} \theta_s^t &= \Pi_{\ell_2}\left(\boldsymbol{\theta}_s^{t-1} + \alpha^t \langle \nabla_{\boldsymbol{\theta}_s} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{t-1}(s), \mathbf{q}^{\boldsymbol{b}}(s) \rangle\right) \\ &= \Pi_{\ell_2}\left(\boldsymbol{\theta}_s^{t-1} + \alpha^t \mathbf{q}^{\boldsymbol{b}}(s)\right), \end{aligned} \tag{5}$$

where the Jacobian $\nabla_{\boldsymbol{\theta}_s} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{t-1}(s)$ is an identity matrix because each parameter $\theta_{s,a}$ corresponds directly to the probability $\pi_{\boldsymbol{\theta}}(s,a)$, and $\langle \cdot, \cdot \rangle$ is the usual matrix inner product.

### 3.2 Tabular ED with Counterfactual Values and Softmax Transfer Function

For some vector of real numbers, $\boldsymbol{\theta}_s$, define $\text{softmax}(\boldsymbol{\theta}_s) = \{\Pi_{\text{sm}}(\boldsymbol{\theta}_s)\}_a = \{\exp(\theta_{s,a}) / \sum_{a'} \exp(\theta_{s,a'})\}_a$. Re-using the tabular policy notation from the previous section, we now define a different instance of exploitability descent. We refer to TabularED($q^c$, softmax) as the algorithm that specifies $\boldsymbol{\pi_\theta}(s) = \Pi_{\text{sm}}(\boldsymbol{\theta}_s)$,

$$\mathbf{q}^{\boldsymbol{b}}(s) = \{q^c_{i,\boldsymbol{\pi}}((\boldsymbol{\pi}_{\boldsymbol{\theta}}^{t-1}, \boldsymbol{b}_{-i}^t), s, a)\}_{a \in \mathcal{A}}, \tag{6}$$

and the policy parameter update as

$$\boldsymbol{\theta}_s^t = \boldsymbol{\theta}_s^{t-1} + \alpha^t \langle \nabla_{\boldsymbol{\theta}_s} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{t-1}(s), \mathbf{q}^{\boldsymbol{b}}(s) \rangle, \tag{7}$$

where $\nabla_{\boldsymbol{\theta}_s} \boldsymbol{\pi}_{\boldsymbol{\theta}}^{t-1}(s)$ represents the Jacobian of softmax.

### 3.3 Convergence Analyses

We now analyze the convergence guarantees of ED. We give results for two cases: first, in cyclical perfect information games and Markov games, and secondly imperfect information games. All the proofs are found in the Appendix **??** of the technical report version of the paper [Lockhart *et al.*, 2019].

**Cyclical Perfect Information Games and Markov Games**

The following result extends the policy gradient theorem [Sutton *et al.*, 2000; Glynn and L'ecuyer, 1995; Williams, 1992; Baxter and Bartlett, 2001] to the zero-sum two-player case. It proves that a generalized gradient of the worst-case value function can be estimated from experience as in the single player case.

**Theorem 1** (Policy Gradient in the Worst Case)**.** *The gradient of policy $\boldsymbol{\pi_{\theta_i}}$'s value, $v_{i,(\boldsymbol{\pi_{\theta_i}},\boldsymbol{b})}$, against a best response, $\boldsymbol{\beta} \doteq \boldsymbol{b_{-i}}(\boldsymbol{\pi_{\theta_i}}) \in \mathrm{BR}(\pi_{\theta_i})$ is a generalized gradient (see [Clarke, 1975]) of $\boldsymbol{\pi_{\theta_i}}$'s worst-case value function,*

$$\nabla_{\theta_i} v_{i,(\pi_{\theta_i}, b_{-i}(\pi_{\theta_i}))} \in \partial \min_{\pi_{-i}} v_{i,(\pi_{\theta_i},\pi_{-i})}.$$

All of the proofs are found in Appendix **??** of the technical report version of the paper [Lockhart *et al.*, 2019].

This theorem is a natural extension of the policy gradient theorem to the zero-sum two-player case. As in policy gradient, this process is only guaranteed to converge to a local maximum of the worst case value $\min_{\pi_{-i}} v_{i,(\pi_{\theta_i},\pi_{-i})}$ of the game but not necessarily to an equilibrium of the game. An equilibrium of the game is reached when the two following conditions are met simultaneously: (1) if the policy is tabular and (2) if all states are visited with at least some probability for all policies. This statement is proven in Appendix **??**.

The method is called exploitability descent because policy gradient in the worst case minimizes exploitability. In a two-player, zero-sum game, if both players independently run ED, NASHCONV is locally minimized.

**Lemma 1.** *In the two-player zero-sum case, simultaneous policy gradient in the worst case locally minimizes* NASHCONV.

**Imperfect Information Games**

We now examine convergence guarantees in the imperfect information setting. There have two main techniques used to solve adversarial games in this case: the first is to rely on the sequence-form representation of policies which makes the optimization problem convex [Koller *et al.*, 1994; Hoda *et al.*, 2007]. The second is to weight the values by the appropriate reach probabilities, and employ local optimizers [Zinkevich *et al.*, 2008; Johanson *et al.*, 2012]. Both take into account the probability of reaching information states, but the latter allows a convenient tabular policy representation.

We prove finite time exploitability bounds for TabularED($q$, $\ell_2$), and we relate TabularED($q^c$, softmax) to a similar algorithm that also has finite time bounds.

The convergence analysis is built upon two previous results: the first is CFR-BR [Johanson *et al.*, 2012]. The second is a very recent result that relates policy gradient optimization in imperfect information games to CFR [Srinivasan *et al.*, 2018]. The result here is also closely related to the optimization against a worst-case opponent [Waugh and Bagnell, 2014, Theorem 4], except our policies are expressed in tabular (*i.e.* behavioral) form rather than the sequence form.

**Case: TabularED($q$, $\ell_2$).** Recall that the parameters $\boldsymbol{\theta} = \{\theta_{s,a}\}_{s\in\mathcal{S},a\in\mathcal{A}}$ correspond to the tabular policy. For convenience, let $\boldsymbol{\theta}_s = \{\theta_{s,a}\}_{a\in\mathcal{A}}$.

We now present the main theorem, which states that if both players optimize their policies using TabularED($q$, $\ell_2$), it will generate policies with decreasing regret, which combined form an approximate Nash equilibrium.

**Theorem 2.** *Let TabularED($q$, $\ell_2$) be described as in Section 3.1 using tabular policies and the update rule in Definition **??**. In a two-player zero-sum game, if each player updates their policy simultaneously using TabularED($q$, $\ell_2$), if $\forall s, a \in \mathcal{S}\times\mathcal{A} : \theta_{s,a} > 0$ and $\alpha^t = t^{-\frac{1}{2}}$, then for each player $i$: after $T$ iterations, a policy $\boldsymbol{\pi}_i^* \in \{\boldsymbol{\pi}_i^1, \cdots, \boldsymbol{\pi}_i^T\}$ will have been generated such that $\boldsymbol{\pi}_i^*$ is $i$'s part of a $\frac{2\epsilon}{T}$-Nash equilibrium, where $\epsilon = |\mathcal{S}_i|\left(\sqrt{T} + (\sqrt{T} - \frac{1}{2})|\mathcal{A}_i|(\Delta_{u_i})^2\right)$, and $\Delta_u = \max_{z,z'\in\mathcal{Z}}(u_i(z) - u_i(z'))$.*

ED is computing best responses each round already, so it is easy to track the best iterate: it will simply be the one with the highest expected value versus the opponent's best response.

The proof can also be applied to the original CFR-BR theorem, so we now present an improved guarantee, whereas the original CFR-BR theorem made a probabilistic guarantee.

**Corollary 1.** *(Improved [Johanson* et al.*, 2012, Theorem 4]) If player $i$ plays $T$ iterations of CFR-BR, then it will have generated a $\boldsymbol{\pi}_i^* \in \{\boldsymbol{\pi}^1, \boldsymbol{\pi}^2, \cdots, \boldsymbol{\pi}^T\}$, where $\boldsymbol{\pi}_i^*$ is a $2\epsilon$-equilibrium, where $\epsilon$ is defined as in [Johanson* et al.*, 2012, Theorem 3].*

The best iterate can be tracked in the same way as ED, and the convergence is guaranteed.

**Remark 1.** *When using q-values, the values are normalized by a quantity, $\mathcal{B}_{-i}(\boldsymbol{\pi}, s)$, that depends on the opponents' policies [Srinivasan* et al.*, 2018, Section 3.2]. The convergence guarantee of TabularED($q$, $\ell_2$) relies on [Srinivasan* et al.*, 2018, Theorem 2], whose proof includes a division by $\mathcal{B}_{-i}(\boldsymbol{\pi}, s)$ [Srinivasan* et al.*, 2018, Appendix E.2]. Therefore, the regret bound is undefined when $\mathcal{B}_{-i}(\boldsymbol{\pi}, s) = 0$, which can happen when an opponent no longer plays to reach $s$.*

**Case: TabularED($q^c$, $\ell_2$).** Instead of using q-values, we can implement ED with counterfactual values. In this case, TabularED with the $\ell_2$ projection becomes CFR-BR(GIGA), which then avoids the issued discussed in Remark 1.

**Theorem 3.** *Let TabularED($q^c$, $\ell_2$) be described as in Section 3.1 using tabular policies and the following update rule:*

$$\boldsymbol{\pi}_i^t(s) = \Pi_{\ell_2}\left(\boldsymbol{\pi}_i^{t-1}(s) + \alpha^t \boldsymbol{q}^{c,\boldsymbol{b}}(s)\right).$$

*Then, Theorem 2 also holds for TabularED($q^c$, $\ell_2$).*

**Case: TabularED($q^c$, softmax)** We now relate TabularED with counterfactual values and softmax policies closely to an algorithm with known finite time convergence bounds. For details, see Appendix **??**.

TabularED($q^c$, softmax) is still a policy gradient algorithm: it differentiates the policy (*i.e.* softmax function) with respect to its parameters, and updates in the direction of higher value. With two subtle changes to the overall process, we can show that the algorithm would become CFR-BR using hedge [Freund and Schapire, 1997] as a local regret minimizer. CFR with hedge is known to have a better bound, but has typically not performed as well as regret matching in practice, though it has been shown to work better when combined with pruning based on dynamic probability thresholding [Brown *et al.*, 2017].

Instead of policy gradient, one can use a softmax transfer function over the the sum of action values (or regrets) over time, which are the gradients of the *value function* with respect to the policy. Accumulating the gradients in this way, the algorithm can be recognized as Mirror Descent [Nemirovsky and Yudin, 1983], which also coincides with hedge given the softmax transfer [Beck and Teboulle, 2003]. When using the counterfactual values, ED then turns into CFR-BR(hedge), which converges for the same reasons as CFR-BR(regret-matching).

We do not have a finite time bound of the exploitability of TabularED($q^c$, softmax) as we do for the same algorithm with an $\ell_2$ projection or CFR-BR(hedge). But since TabularED($q^c$, softmax) is a policy gradient algorithm, its policy will be adjusted toward a local optimum upon each update and will converge at that point when the gradient is zero. We use this algorithm because the policy gradient formulation allows for easily-applicable general function approximation.

# 4 Experimental Results

We now present our experimental results. We start by comparing empirical convergence rates to XFP and CFR in the tabular setting, following by convergence behavior when training neural network functions to approximate the policy.

In our initial experiments, we found that using $q$-values led to plateaus in convergence in some cases, possibly due to numerical instability caused by the problem outlined in Remark 1. Therefore, we present results only using TabularED($q^c$, softmax), which for simplicity we refer to as TabularED for the remainder of this section. We also found that the algorithm converged faster with slightly higher learning rates than the ones suggested by Section 3.3.

## 4.1 Experiment Domains

Our experiments are run across four different imperfect information games. We provide very brief descriptions here; see Appendix **??** as well as [Kuhn, 1950; Southey *et al.*, 2005] and [Lanctot, 2013, Chapter 3] for more detail.

**Kuhn poker** is a simplified poker game first proposed by Harold Kuhn [Kuhn, 1950] **Leduc poker** is significantly larger game with two rounds and a 6-card deck in two suits, e.g. {JS,QS,KS, JH,QH,KH}. **Liar's Dice**(1,1) is dice game where each player gets a single private die, rolled at the start of the game, and players proceed to bid on the outcomes of all dice in the game. **Goofspiel** is a card game where players try to obtain point cards by bidding simultaneously. We use an imperfect information variant where bid cards are unrevealed.

## 4.2 Convergence Results

We now present empirical convergence rates to $\epsilon$-Nash equilibria. The main results are depicted in Figure 1.

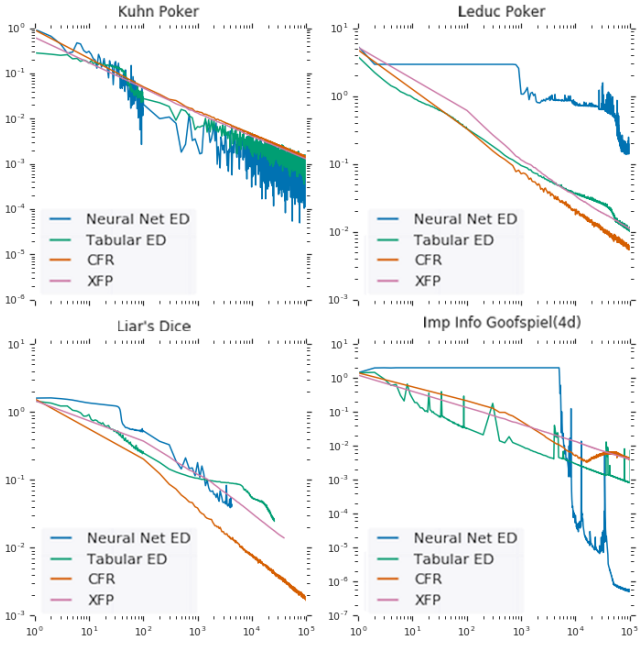


Figure 1: Extensive-form fictitious play (XFP), CFR, tabular and neural-net ED. The y-axis is NASHCONV defined in Section 2, and the x-axis is number of iterations, both in log-scale.

For the neural network experiments, we use a single policy network for both players, which takes as input the current state of the game and whose output is a softmax distribution over the actions of the game. The state of the game is represented in a game-dependent fashion as a fixed-size vector of between 11 and 52 binary bits, encoding public information, private information, and the game history.

The neural network consists of a number of fully-connected hidden layers, each with the same number of units and with rectified linear activation functions after each layer. A linear output layer maps from the final hidden layer to a value per action. The values for the legal actions are selected and mapped to a policy using the softmax function.

At each step, we evaluate the policy for every state of the game, compute a best response to it, and evaluate each action against the best response. We then perform a single gradient descent step on the loss function: $-\sum_s \boldsymbol{\pi}_i(s) \cdot \left(\mathbf{q}^b(s) - B(s)\right) + w_r \frac{1}{n} \sum_i \boldsymbol{\theta}_i^2$, where the final term is a regularization for all the neural network weights, and the baseline $B(s)$ is a computed constant (i.e. it does not contribute to the gradient calculation) with $B(s) = \boldsymbol{\pi}_i(s) \cdot \mathbf{q}^{\boldsymbol{b}}(s)$. We performed a sweep over the number of hidden layers (from 1 to 5), the number of hidden units (64, 128 or 256), the regularization weight ($10^{-7}, 10^{-6}, 10^{-5}, 10^{-4}$), and the initial learning rate (powers of 2). The plotted results show the best values from this sweep for each game.

### 4.3 Discussion

There are several interesting observations to make about the results. First, the convergence of the neural network policies is more erratic than the tabular counterparts. However, in two games the neural network policies have learned *more accurate* approximate equilibria than any of the tabular algorithms for the same number of iterations. The network could be generalizing across the state space (discovering patterns) in a way that is not possible in the tabular case, despite raw input features.

Although Tabular ED and XFP have roughly the same convergence rate, the respective function approximation versions have an order of magnitude difference in speed, with Neural ED reaching an exploitability of $0.08$ in Leduc Poker after $10^5$ iterations, a level which NFSP reaches after approximately $10^6$ iterations [Heinrich and Silver, 2016]. Neural ED and NFSP are not directly comparable as NFSP is computing an approximate equilibrium using sampling and RL while ED uses true best response. However, NFSP uses a reservoir buffer dataset of 2 million entries, whereas this is not required in ED.

## 5 Conclusion

We introduce Exploitability Descent (ED) that optimizes its policy directly against worst-case opponents. In cyclical perfect information and Markov games, we prove that ED policies converge to strong policies that are unexploitable in the tabular case. In imperfect information games, we also present finite time exploitability bounds for tabular policies. While the empirical convergence rates using tabular policies are comparable to previous algorithms, the policies themselves provably converge. So, unlike XFP and CFR, there is no need to compute the average policy. Neural network function approximation is applicable via direct policy gradient ascent, also avoiding domain-specific abstractions, or the need to store large replay buffers of past experience, as in neural fictitious self-play [Heinrich and Silver, 2016], or a set of past networks, as in PSRO [Lanctot *et al.*, 2017].

In some of our experiments, neural networks learned lower-exploitability policies than the tabular counterparts, which could be an indication of strong generalization potential by recognizing similar patterns across states. There are interesting directions for future work: using approximate best responses and sampling trajectories for the policy optimization in larger games where enumerating the trajectories is not feasible.

## Acknowledgments

We would like to thank Neil Burch, Johannes Heinrich, and Martin Schmid for feedback. Dustin Morrill was supported by The Alberta Machine Intelligence Institute (Amii) and Alberta Treasury Branch (ATB) during the course of this research.

## References

[Baxter and Bartlett, 2001] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

[Beck and Teboulle, 2003] Amir Beck and Marc Teboulle. Mirror descent and nonlinear projected subgradient methods for convex optimization. *Operations Research Letters*, 31(3):167–175, May 2003.

[Bowling *et al.*, 2015] Michael Bowling, Neil Burch, Michael Johanson, and Oskari Tammelin. Heads-up Limit Hold'em Poker is solved. *Science*, 347(6218):145–149, January 2015.

[Boyd and Vandenberghe, 2004] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[Brown and Sandholm, 2017] Noam Brown and Tuomas Sandholm. Superhuman AI for heads-up no-limit poker: Libratus beats top professionals. *Science*, 360(6385), December 2017.

[Brown *et al.*, 2017] Noam Brown, Christian Kroer, and Tuomas Sandholm. Dynamic thresholding and pruning for regret minimization. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[Brown *et al.*, 2019] Noam Brown, Adam Lerer, Sam Gross, and Tuomas Sandholm. Deep counterfactual regret minimization. In *Proceedings of the Thirty-sixth International Conference on Machine Learning (ICML)*, 2019. Full technical report available at http://arxiv.org/abs/1811.00164.

[Campbell *et al.*, 2002] M. Campbell, A. J. Hoane, and F. Hsu. Deep blue. *Artificial Intelligence*, 134:57–83, 2002.

[Clarke, 1975] Frank H Clarke. Generalized gradients and applications. *Transactions of the American Mathematical Society*, 205:247–262, 1975.

[Freund and Schapire, 1997] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[Gale *et al.*, 1951] D. Gale, H.W. Kuhn, and A.W. Tucker. Linear programming and the theory of games. In T.C. Koopmans et al., editor, *Activity Analysis of Production and Allocation*, pages 317–329. Wiley: New York, 1951.

[Glynn and L'ecuyer, 1995] Peter W Glynn and Pierre L'ecuyer. Likelihood ratio gradient estimation for stochastic recursions. *Advances in applied probability*, 1995.

[Hart and Mas-Colell, 2000] S. Hart and A. Mas-Colell. A simple adaptive procedure leading to correlated equilibrium. *Econometrica*, 68(5):1127–1150, 2000.

[Heinrich and Silver, 2016] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *CoRR*, abs/1603.01121, 2016.

[Heinrich *et al.*, 2015] Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *ICML 2015*, 2015.

[Hoda *et al.*, 2007] S. Hoda, A. Gilpin, and J. Pe na. A gradient-based approach for computing Nash equilibria of large sequential games. *Optimization Online*, July 2007. http://www.optimization-online.org/DB_HTML/2007/07/1719.html.

[Johanson *et al.*, 2012] M. Johanson, N. Bard, N. Burch, and M. Bowling. Finding optimal abstract strategies in extensive form games. In *Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI)*, pages 1371–1379, 2012.

[Koller *et al.*, 1994] D. Koller, N. Megiddo, and B. von Stengel. Fast algorithms for finding randomized strategies in game trees. In *Proceedings of the 26th ACM Symposium on Theory of Computing (STOC '94)*, pages 750–759, 1994.

[Kuhn, 1950] H. W. Kuhn. Simplified two-person Poker. *Contributions to the Theory of Games*, 1:97–103, 1950.

[Lanctot *et al.*, 2017] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Perolat, David Silver, and Thore Graepel. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

[Lanctot, 2013] Marc Lanctot. *Monte Carlo Sampling and Regret Minimization for Equilibrium Computation and Decision-Making in Large Extensive Form Games*. PhD thesis, Department of Computing Science, University of Alberta, Edmonton, Alberta, Canada, June 2013.

[Lockhart *et al.*, 2019] Edward Lockhart, Marc Lanctot, Julien Pérolat, Jean-Baptiste Lespiau, Dustin Morrill, Finbarr Timbers, and Karl Tuyls. Computing approximate equilibria in sequential adversarial games by exploitability descent. *CoRR*, abs/1903.05614, 2019.

[Moravčík *et al.*, 2017] Matej Moravčík, Martin Schmid, Neil Burch, Viliam Lisý, Dustin Morrill, Nolan Bard, Trevor Davis, Kevin Waugh, Michael Johanson, and Michael Bowling. Deepstack: Expert-level artificial intelligence in heads-up no-limit poker. *Science*, 358(6362), October 2017.

[Nemirovsky and Yudin, 1983] Arkadii Semenovich Nemirovsky and David Borisovich Yudin. Problem complexity and method efficiency in optimization. 1983.

[Rubin and Watson, 2011] J. Rubin and I. Watson. Computer poker: A review. *Artificial Intelligence*, 175(5–6):958–987, 2011.

[Shoham and Leyton-Brown, 2009] Y. Shoham and K. Leyton-Brown. *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge University Press, 2009.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484—-489, 2016.

[Silver *et al.*, 2018] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science*, 632(6419):1140–1144, 2018.

[Southey *et al.*, 2005] Finnegan Southey, Michael Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and Chris Rayner. Bayes' bluff: Opponent modelling in poker. In *Proceedings of the Twenty-First Conference on Uncertaintyin Artificial Intelligence (UAI)*, pages 550–558, 2005.

[Srinivasan *et al.*, 2018] Sriram Srinivasan, Marc Lanctot, Vinicius Zambaldi, Julien Pérolat, Karl Tuyls, Rémi Munos, and Michael Bowling. Actor-critic policy optimization in partially observable multiagent environments. In *Advances in Neural Information Processing Systems*, 2018.

[Sutton *et al.*, 2000] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12*, pages 1057–1063. MIT Press, 2000.

[Tang *et al.*, 2018] Jie Tang, Keiran Paster, and Pieter Abbeel. Equilibrium finding via asymmetric self-play reinforcement learning. In *Deep Reinforcement Learning Workshop NeurIPS 2018*, 2018.

[Waugh and Bagnell, 2014] Kevin Waugh and J. Andrew Bagnell. A unified view of large-scale zero-sum equilibrium computation. *CoRR*, abs/1411.5007, 2014.

[Waugh *et al.*, 2015] Kevin Waugh, Dustin Morrill, J. Andrew Bagnell, and Michael Bowling. Solving games with functional regret estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2015.

[Williams, 1992] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

[Zinkevich *et al.*, 2008] M. Zinkevich, M. Johanson, M. Bowling, and C. Piccione. Regret minimization in games with incomplete information. In *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, 2008.