Variance Reduction in Monte Carlo Counterfactual Regret Minimization (VR-MCCFR) for Extensive Form Games using Baselines

Martin Schmid[1], Neil Burch[1], Marc Lanctot[1], Matej Moravcik[1], Rudolf Kadlec[1], Michael Bowling[1,2]

DeepMind[1]

University of Alberta[2]

{mschmid,burchn,lanctot,moravcik,rudolfkadlec,bowlingm}@google.com

Abstract

Learning strategies for imperfect information games from samples of interaction is a challenging problem. A common method for this setting, Monte Carlo Counterfactual Regret Minimization (MCCFR), can have slow long-term convergence rates due to high variance. In this paper, we introduce a variance reduction technique (VR-MCCFR) that applies to any sampling variant of MCCFR. Using this technique, per-iteration estimated values and updates are reformulated as a function of sampled values and state-action baselines, similar to their use in policy gradient reinforcement learning. The new formulation allows estimates to be bootstrapped from other estimates within the same episode, propagating the benefits of baselines along the sampled trajectory; the estimates remain unbiased even when bootstrapping from other estimates. Finally, we show that given a perfect baseline, the variance of the value estimates can be reduced to zero. Experimental evaluation shows that VR-MCCFR brings an order of magnitude speedup, while the empirical variance decreases by three orders of magnitude. The decreased variance allows for the first time CFR+ to be used with sampling, increasing the speedup to two orders of magnitude.

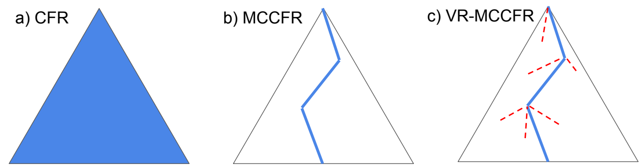


Figure 1: High-level overview of Variance Reduction MCCFR (VR-MCCFR) and related methods. a) CFR traverses the entire tree on every iteration. b) MCCFR samples trajectories and computes the values only for the sampled actions, while the off-trajectory actions are treated as zero-valued. While MCCFR uses importance sampling weight to ensure the values are unbiased, the sampling introduces high variance. c) VR-MCCFR follows the same sampling framework as MCCFR, but uses baseline values for both sampled actions (in blue) as well as the off-trajectory actions (in red). These baselines use control variates and send up bootstrapped estimates to decrease the per-iteration variance thus speeding up the convergence.

Introduction

Policy gradient algorithms have shown remarkable success in single-agent reinforcement learning (RL) (Mnih *et al.* 2016; Schulman *et al.* 2017). While there has been evidence of empirical success in multiagent problems (Foerster *et al.* 2017; Bansal *et al.* 2018), the assumptions made by RL methods generally do not hold in multiagent partially-observable environments. Hence, they are not guaranteed to find an optimal policy, even with tabular representations in two-player zero-sum (competitive) games (Littman 1994). As a result, policy iteration algorithms based on computational game theory and regret minimization have been the preferred formalism in this setting. Counterfactual regret minimization (Zinkevich *et al.* 2008) has been a core component of this progress in Poker AI, leading to solving Heads-Up Limit Texas Hold'em (Bowling *et al.* 2015) and defeating professional poker players in No-Limit (Moravčík *et al.* 2017; Brown and Sandholm 2017).

The two fields of RL and computational game theory have largely grown independently. However, there has been recent work that relates approaches within these two communities. Fictitious self-play uses RL to compute approximate best responses and supervised learning to combine responses (Heinrich *et al.* 2015). This idea is extended to a unified training framework that can produce more general policies by regularizing over generated response oracles (Lanctot *et al.* 2017). RL-style regressors were first used to compress regrets in game theorietic algorithms (Waugh *et al.* 2015). DeepStack introduced deep neural networks as generalized value-function approximators for online planning in imperfect information games (Moravčík *et al.* 2017). These value functions operate on a belief-space over all possible states consistent with the players' observations.

This paper similarly unites concepts from both fields, proposing an unbiased variance reduction technique for Monte Carlo counterfactual regret minimization using an analog of state-action baselines from actor-critic RL methods. While policy gradient methods typically involve Monte Carlo estimates, the analog in imperfect information settings is Monte Carlo Counterfactual Regret Minimization (MCCFR) (Lanctot *et al.* 2009). Policy gradient estimates based on a single sample of an episode suffer significantly from variance. A common technique to decrease the variance is a state or state-action dependent baseline value that is subtracted from the observed return. These methods can drastically improve the convergence speed. However, no such methods are known for MCCFR.

MCCFR is a sample based algorithm in imperfect information settings, which approximates counterfactual regret minimization (CFR) by estimating regret quantities necessary for updating the policy. While MCCFR can offer faster short-term convergence than original CFR in large games, it suffers from high variance which leads to slower long-term convergence.

CFR+ provides significantly faster empirical performance and made solving Heads-Up Limit Texas Hold'em possible (Bowling *et al.* 2015). Unfortunately, CFR+ has so far did not outperform CFR in Monte Carlo settings (Burch 2017) (also see Figure (7) in the appendix for an experiment).

In this work, we reformulate the value estimates using a control variate and a state-action baseline. The new formulation includes any approximation of the counterfactual values, which allows for a range of different ways to insert domain-specific knowledge (if available) but also to design values that are learned online.

Our experiments show two orders of magnitude improvement over MCCFR. For the common testbed imperfect information game – Leduc Poker – VR-MCCFR with a state-action baseline needs 250 times fewer iterations than MCCFR to reach the same solution quality. In contrast to RL algorithms in perfect information settings, where state-action baselines bring little to no improvement over state baselines (Tucker *et al.* 2018), state-action baselines lead to significant improvement over state baselines in multiagent partially-observable settings. We suspect this is due to variance from the environment and different dynamics of the policies during the computation.

Related Work

There are standard variance reduction techniques for Monte Carlo sampling methods (Owen 2013) and the use of control variates in these settings has a long history (Boyle 1977). Reducing variance is particularly important when estimating gradients from sample trajectories. Consequentially, the use of a control variates using baseline has become standard practice in policy gradient methods (Williams 1992; Sutton and Barto 2017). In RL, action-dependent baselines have recently shown promise (Wu *et al.* 2018; Liu *et al.* 2018) but the degree to which variance is indeed reduced remains unclear (Tucker *et al.* 2018). We show that in our setting of MCCFR in imperfect information multiplayer games, action-dependent baselines necessarily influence the variance of the estimates, and we confirm the reduction empirically. This is important because lower-variance estimates lead to better regret bounds (Gibson *et al.* 2012).

There have been a few uses of variance reduction techniques in multiplayer games, within Monte Carlo tree search (MCTS). In MCTS, control variates have used to augment the reward along a trajectory using a property of the state before and after a transition (Veness *et al.* 2011) and to augment the outcome of a rollout from its length or some predetermined quality of the states visited (Pepels *et al.* 2014).

Our baseline-improved estimates are similar to the ones used in AIVAT (Burch *et al.* 2018). AIVAT defines estimates of expected values using heuristic values of states as baselines in practice. Unlike this work, AIVAT was only used for evaluation of strategies.

To the best of our knowledge, there has been two applications of variance reduction in Monte Carlo CFR: by manipulating the chance node distribution (Lanctot 2013, Section 7.5) and by sampling ("probing") more trajectories for more estimates of the underlying values (Gibson *et al.* 2012). The variance reduction (and resulting drop in convergence rate) is modest in both cases, whereas we show more than a two order of magnitude speed-up in convergence using our method.

Background

We start with the formal background necessary to understand our method. For details, see (Shoham and Leyton-Brown 2009; Sutton and Barto 2017).

A two player **extensive-form game** is tuple $(\mathcal{N}, \mathcal{A}, \mathcal{H}, \mathcal{Z}, \tau, u, \mathcal{I})$.

$\mathcal{N} = \{1, 2, c\}$ is a finite set of players, where c is a special player called chance. $\mathcal{A}$ is a finite set of actions. Players take turns choosing actions, which are composed into sequences called *histories*; the set of all valid histories is $\mathcal{H}$, and the set of all terminal histories (games) is $\mathcal{Z} \subseteq \mathcal{H}$. We use the notation $h' \sqsubseteq h$ to mean that h' is a prefix sequence or equal to h. Given a nonterminal history h, the player function $\tau : \mathcal{H} \setminus \mathcal{Z} \to \mathcal{N}$ determines who acts at h. The utility function $u : (\mathcal{N} \setminus \{c\}) \times \mathcal{Z} \to [u_{\min}, u_{\max}] \subset \mathbb{R}$ assigns a payoff to each player for each terminal history $z \in \mathcal{Z}$.

The notion of a state in imperfect information games requires groupings of histories: $\mathcal{I}_i$ for some player $i \in \mathcal{N}$ is a partition of $\{h \in \mathcal{H} \mid \tau(h) = i\}$ into parts $I \in \mathcal{I}_i$ such that $h, h' \in I$ if player i cannot distinguish h from h' given the information known to player i at the two histories. We call these **information sets**. For example, in Texas Hold'em poker, for all $I \in \mathcal{I}_i$, the (public) actions are the same for all $h, h' \in I$, and h only differs from h' in cards dealt to the opponents (actions chosen by chance). For convenience, we refer to $I(h)$ as the information state that contains h.

At any I, there is a subset of legal actions $A(I) \subseteq \mathcal{A}$. To choose actions, each player i uses a **strategy** $\sigma_i : I \to \Delta(A(I))$, where $\Delta(X)$ refers to the set of probability distributions over X. We use the shorthand $\sigma(h, a)$ to refer to $\sigma(I(h), a)$. Given some history h, we define the **reach probability** $\pi^\sigma(h) = \Pi_{h'a \sqsubseteq h} \sigma_{\tau(h')}(I(h'), a)$ to be the product of all action probabilities leading up to h. This reach probability contains all players' actions, but can be separated $\pi^\sigma(h) = \pi^\sigma_i(h)\pi^\sigma_{-i}(h)$ into player i's actions' contribution and the contribution of the opponents' of player i (including chance).

Finally, it is often useful to consider the **augmented information sets** (Burch *et al.* 2014). While an information set I groups histories h that player $i = \tau(h)$ cannot distinguish, an augmented information set groups histories that player i can not distinguish, including these where $\tau(h) \neq i$. For a history h, we denote an augmented information set of player i as $I_i(h)$. Note that the if $\tau(h) = i$ then $I_i(h) = I(h)$ and $I(h) = I_{\tau(h)}(h)$.

Counterfactual Regret Minimization

Counterfactual Regret (CFR) Minimization is an iterative algorithm that produces a sequence of strategies $\sigma^0, \sigma^1, \dots, \sigma^T$, whose average strategy $\bar{\sigma}^T$ converges to an approximate Nash equilibrium as $T \to \infty$ in two-player zero-sum games (Zinkevich *et al.* 2008). Specifically, on iteration t, for each I, it computes **counterfactual values**. Define $\mathcal{Z}_I = \{(h,z) \in \mathcal{H} \times \mathcal{Z} \mid h \in I, h \sqsubseteq z\}$, and $u_i^{\sigma^t}(h,z) = \pi^{\sigma^t}(h,z)u_i(z)$. We will also sometimes use the short form $u_i^{\sigma}(h) = \sum_{z \in \mathcal{Z}, h \sqsubseteq z} u_i^{\sigma}(h,z)$. A counterfactual value is:

$$v_i(\sigma^t, I) = \sum_{(h,z) \in \mathcal{Z}_I} \pi_{-i}^{\sigma^t}(h) u_i^{\sigma^t}(h,z). \quad (1)$$

We also define an action-dependent counterfactual value,

$$v_i(\sigma, I, a) = \sum_{(h,z) \in \mathcal{Z}_I} \pi_{-i}^{\sigma}(ha) u^{\sigma}(ha,z), \quad (2)$$

where ha is the sequence h followed by the action a. The values are analogous to the difference in Q-values and V-values in RL, and indeed we have $v_i(\sigma, I) = \sum_a \sigma(I,a) v_i(\sigma, I, a)$. CFR then computes a **counterfactual regret** for *not taking* a at I:

$$r^t(I,a) = v_i(\sigma^t, I, a) - v_i(\sigma^t, I), \quad (3)$$

This regret is then accumulated $R^T(I,a) = \sum_{t=1}^{T} r^t(I,a)$, which is used to update the strategies using **regret-matching** (Hart and Mas-Colell 2000):

$$\sigma^{T+1}(I,a) = \frac{(R^T(I,a))^+}{\sum_{a \in A(I)} (R^T(I,a))^+}, \quad (4)$$

where $(x)^+ = \max(x, 0)$, or to the uniform strategy if $\sum_a (R^T(I,a))^+ = 0$. CFR+ works by thresholding the quantity at each round (Tammelin *et al.* 2015): define $Q^0(I,a) = 0$ and $Q^T(I,a) = (Q^{T-1} + r^T(I,a))^+$; CFR+ updates the policy by replacing R^T by Q^T in equation 4. In addition, it always alternates the regret updates of the players (whereas some variants of CFR update both players), and the average strategy places more (linearly increasing) weight on more recent iterations.

If for player i we denote $u(\sigma) = u_i(\sigma_i, \sigma_{-i})$, and run CFR for T iterations, then we can define the **overall regret** of the strategies produced as:

$$R_i^T = \max_{\sigma_i'} \sum_{t=1}^{T} \left(v_i(\sigma_i', \sigma_{-i}^t) - v_i(\sigma^t) \right).$$

CFR ensures that $R_i^T / T \to 0$ as $T \to \infty$. When two players minimize regret, the folk theorem then guarantees a bound on the distance to a Nash equilibrium as a function of R_i^T/T.

To compute v_i precisely, each iteration requires traversing over subtrees under each $a \in A(I)$ at each I. Next, we describe variants that allow sampling parts of the trees and using estimates of these quantities.

Monte Carlo CFR

Monte Carlo CFR (MCCFR) introduces sample estimates of the counterfactual values, by visiting and updating quantities over only part of the entire tree. MCCFR is a general family of algorithms: each instance defined by a specific sampling policy. For ease of exposition and to show the similarity to RL, we focus on **outcome sampling** (Lanctot *et al.* 2009); however, our baseline-enhanced estimates can be used in all MCCFR variants. A **sampling policy** ξ is defined in the same way as a strategy (a distribution over $A(I)$ for all I) with a restriction that $\xi(h,a) > 0$ for all histories and actions. Given a terminal history sampled with probability $q(z) = \pi^{\xi}(z)$, a **sampled counterfactual value** $\tilde{v}_i(\sigma, I|z)$

$$= \tilde{v}_i(\sigma, h|z) = \frac{\pi_{-i}^{\sigma}(h) u_i^{\sigma}(h,z)}{q(z)}, \text{ for } h \in I, h \sqsubseteq z, \quad (5)$$

and 0 for histories that were not played, $h \not\sqsubseteq z$. The estimate is unbiased: $\mathbb{E}_{z \sim \xi}[\tilde{v}_i(\sigma, I|z)] = v_i(\sigma, I)$, by (Lanctot *et al.* 2009, Lemma 1). As a result, $\tilde{v}_i$ can be used in Equation 3 to accumulate estimated regrets $\tilde{r}^t(I,a) = \tilde{v}_i(\sigma^t, I, a) - \tilde{v}_i(\sigma^t, I)$ instead. The regret bound requires an additional term $\frac{1}{\min_{z \in \mathcal{Z}} q(z)}$, which is exponential in the length of z and similar observations have been made in RL (Arjona-Medina *et al.* 2018). The main problem with the sampling variants is that they introduce variance that can have a significant effect on long-term convergence (Gibson *et al.* 2012).

Control Variates

Suppose one is trying to estimate a statistic of a random variable, X, such as its mean, from samples $\mathbf{X} = (X_1, X_2, \cdots, X_n)$. A crude Monte Carlo estimator is defined to be $\hat{X}^{mc} = \frac{1}{n} \sum_{i=1}^{n} X_i$. A **control variate** is a random variable Y with a known mean $\mu_Y = \mathbb{E}[Y]$, that is paired with the original variable, such that samples are instead of the form $(\mathbf{X}, \mathbf{Y})$ (Owen 2013). A new random variable is then defined, $Z_i = X_i + c(Y_i - \mu_Y)$. An estimator $\hat{Z}^{cv} = \frac{1}{n} \sum_{i=1}^{n} Z_i$. Since $\mathbb{E}[Z_i] = \mathbb{E}[X_i]$ for any value of c, $\hat{Z}^{cv}$ can be used in place of $\hat{X}^{mc}$. with variance $\mathbb{V}\text{ar}[Z_i] = \mathbb{V}\text{ar}[X_i] + c^2 \mathbb{V}\text{ar}[Y_i] + 2c\mathbb{C}\text{ov}[X_i, Y_i]$. So when X and Y are positively correlated and $c < 0$, variance is reduced when $\mathbb{C}\text{ov}[X,Y] > \frac{c^2}{2} \mathbb{V}\text{ar}[Y]$.

Reinforcement Learning Mapping

There are several analogies to make between Monte Carlo CFR in imperfect information games and reinforcement learning. Since our technique builds on ideas that have been widely used in RL, we end the background by providing a small discussion of the links.

First, dynamics of an imperfect information game are similar to a partially-observable episodic MDP without any cycles. Policies and strategies are identically defined, but in imperfect information games a deterministic optimal (Nash) strategy may not exist causing most of the RL methods to fail to converge. The search for a minmax-optimal strategy with several players is the main reason CFR is used instead of, for example, value iteration. However, both operate by defining values of states which are analogous (counterfactual values versus expected values) since they are both functions of the strategy/policy; therefore, can be viewed as a kind of policy iteration which computes the values and from which a policy is derived. However, the iterates σ^t are not guaranteed to converge, only the average strategy $\bar{\sigma}^t$ does.

Monte Carlo CFR is an off-policy Monte Carlo analog. The value estimates are unbiased specifically because they are corrected by importance sampling. Most applications of MCCFR have operated with tabular representations, but this is mostly due to the differences in objectives. Function approximation methods have been proposed for CFR (Waugh *et al.* 2015) but the variance from pure Monte Carlo methods may prevent such techniques in MCCFR. The use of baselines has been widely successful in policy gradient methods, so reducing the variance could enable the practical use of function approximation in MCCFR.

It was recently shown that policy gradient and actor-critic algorithms implement a form of *on-policy* Monte Carlo CFR in zero-sum games, inspiring regret-based policy update rules (Srinivasan *et al.* 2018); we suspect that the variance reduction techniques proposed in this paper could apply to these model-free RL algorithms as well.

Monte Carlo CFR with Baselines

We now introduce our technique: MCCFR with baselines. While the baselines are analogous to those from policy gradient methods (using counterfactual values), there are slight differences in their construction.

Our technique constructs value estimates using control variates. Note that MCCFR is using sampled estimates of counterfactual values $\tilde{v}_i(\sigma, I)$ whose expected value is the counterfactual value $v_i(\sigma, I)$. First, we introduce an **estimated counterfactual** value $\hat{v}_i(\sigma, I)$ to be any estimator of the counterfactual value (not necessarily $\tilde{v}_i$ as defined above, but this is one possibility).

We now define an action-dependent **baseline** $b_i(I, a)$ that, as in RL, serves as a basis for the sampled values. The intent is to define a baseline function to approximate or be correlated with $\mathbb{E}[\hat{v}_i(\sigma, I, a)]$. We also define a sampled baseline $\hat{b}_i(I, a)$ as an estimator such that $\mathbb{E}[\hat{b}_i(I, a)] = b_i(I, a)$. From this, we construct a new baseline-enhanced estimate for the counterfactual values:

$$\widehat{v}_i^b(\sigma, I, a) = \widehat{v}_i(\sigma, I, a) - \hat{b}_i(\sigma, I, a) + b_i(\sigma, I, a) \qquad (6)$$

First, note that $\hat{b}_i$ is a control variate with $c = -1$. Therefore, it is important that $\hat{b}_i$ be correlated with $\hat{v}_i$. The main idea of our technique is to replace $\tilde{v}_i(\sigma, I, a)$ with $\hat{v}_i^b(\sigma, I, a)$. A key property is that by doing so, the expectation remains unchanged.

Lemma 1. *For any* $i \in \mathcal{N} - \{c\}, \sigma_i, I \in \mathcal{I}, a \in A(I)$, *if* $\mathbb{E}[\hat{b}_i(I, a)] = b_i(I, a)$ *and* $\mathbb{E}[\hat{v}_i(\sigma, I, a)] = v_i(\sigma, I, a)$, *then* $\mathbb{E}[\hat{v}_i^b(\sigma, I, a)] = v_i(\sigma, I, a)$.

The proof is in the appendix. As a result, any baseline whose expectation is known can be used and the baseline-enhanced estimates are consistent. However, not all baselines will decrease variance. For example, if $\mathbb{C}\text{ov}[\hat{v}_i, \hat{b}_i]$ is too low, then the $\mathbb{V}\text{ar}[\hat{b}_i]$ term could dominate and actually increase the variance.

Recursive Bootstrapping

Consider the individual computation (1) for all the information sets on the path to a sampled terminal history z. Given that the counterfactual values up the tree can be computed from the counterfactual values down the tree, it is natural to consider propagating the already baseline-enhanced counterfactual values (6) rather than the original noisy sampled values - thus propagating the benefits up the tree. The Lemma (2) then shows that by doing so, the updates remain unbiased. Our experimental section shows that such bootstrapping a crucial component for the proper performance of the method.

To properly formalize this bootstrapping computation, we must first recursively define the **expected value**:

$$\hat{u}_i(\sigma, h, a|z) = \begin{cases} \hat{u}_i(\sigma, ha|z)/\xi(h, a) & \text{if } ha \sqsubseteq z \\ 0 & \text{otherwise} \end{cases}, \qquad (7)$$

and

$$\hat{u}_i(\sigma, h|z) = \begin{cases} u_i(h) & \text{if } h = z \\ \sum_a \sigma(h, a)\hat{u}_i(\sigma, h, a|z) & \text{if } h \sqsubset z \\ 0 & \text{otherwise} \end{cases}. \qquad (8)$$

Next, we define a baseline-enhanced version of the expected value. Note that the baseline $b_i(I, a)$ can be arbitrary, but we discuss a particular choice and update of the baseline in the later section. For every action, given a specific sampled trajectory z, then $\hat{u}_i^b(\sigma, h, a|z) =$

$$\begin{cases} b_i(I_i(h), a) + \frac{\hat{u}_i^b(\sigma, ha|z) - b_i(I_i(h), a)}{\xi(h, a)} & \text{if } ha \sqsubseteq z \\ b_i(I_i(h), a) & \text{if } h \sqsubset z, ha \not\sqsubseteq z \\ 0 & \text{otherwise} \end{cases} \qquad (9)$$

and

$$\hat{u}_i^b(\sigma, h|z) = \begin{cases} u_i(h) & \text{if } h = z \\ \sum_a \sigma(h, a)\hat{u}_i^b(\sigma, h, a|z) & \text{if } h \sqsubset z \\ 0 & \text{otherwise} \end{cases}. \qquad (10)$$

These are the values that are bootstrapped. We estimate counterfactual values needed for the regret updates using these values as:

$$\hat{v}_i^b(\sigma, I(h), a|z) = \hat{v}_i^b(\sigma, h, a|z) = \frac{\pi_{-i}^{\sigma}(h)}{q(h)}\hat{u}_i^b(\sigma, h, a|z). \qquad (11)$$

We can now formally state that the bootstrapping keeps the counterfactual values unbiased:

Lemma 2. *Let* $\hat{v}_i^b$ *be defined as in Equation 11. Then, for any* $i \in \mathcal{N} - \{c\}, \sigma_i, I \in \mathcal{I}, a \in A(I)$, *it holds that* $\mathbb{E}_z[\hat{v}_i^b(\sigma, I, a|z)] = v_i(\sigma, I, a)$.

The proof is in the appendix. Since each estimate builds on other estimates, the benefit of the reduction in variance can be propagated up through the tree.

Another key result is that there exists a perfect baseline that leads to zero-variance estimates at the updated information sets.

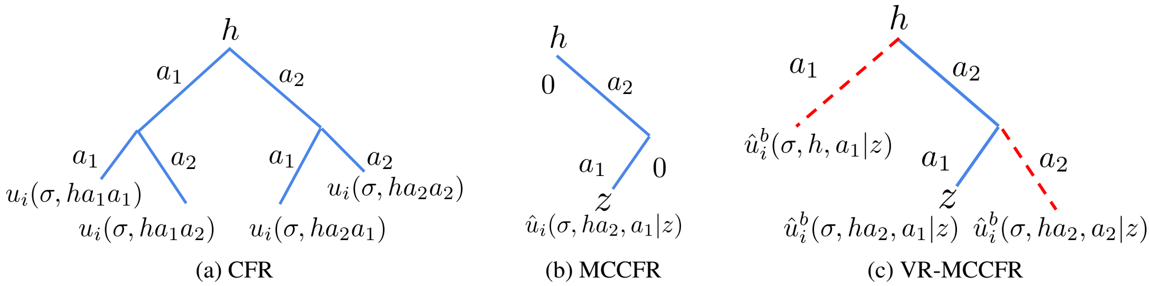


Figure 2: Values and updates for the discussed methods: (a) CFR udpates the full tree and thus uses the exact values for all the actions, (b) MCCFR updates only a single path, and uses the sampled values for the sampled actions and zero values for the off-trajectory actions, (c) VR-MCCFR also updates only a single path, but uses the bootstrapped baseline-enhanced values for the sampled actions and baseline-enhanced values for the off-trajectory actions.

Lemma 3. *There exists a perfect baseline b^* and optimal unbiased estimator $\hat{v}^*_i(\sigma, h, a)$ such that under a specific update scheme: $\mathbb{V}ar_{h,z\sim\xi,h\in I,h\sqsubseteq z}[\hat{v}^*_i(\sigma, h, a|z)] = 0$.*

The proof and description of the update scheme are in the appendix. We will refer to b^* as the **oracle baseline**. Note that even when using the oracle baseline, the convergence rate of MCCFR is still not identical to CFR because each iteration applies regret updates to a portion of the tree, whereas CFR updates the entire tree.

Finally, using unbiased estimates to tabulate regrets $\hat{r}(I, a)$ for each I and a leads to a probabilistic regret bound:

Theorem 1. *(Gibson* et al. *2012, Theorem 2) For some unbiased estimator of the counterfactual values $\hat{v}_i$ and a bound on the difference in its value $\hat{\Delta}_i = |\hat{v}_i(\sigma, I, a) - \hat{v}_i(\sigma, I, a')|$, with probability 1-p, $\frac{R^T_i}{T}$*

$$\leq \left(\hat{\Delta}_i + \frac{\sqrt{\max_{t,I,a} \mathbb{V}ar[r^t_i(I, a) - \hat{r}^t_i(I, a)]}}{\sqrt{p}} \right) \frac{|\mathcal{I}_i||\mathcal{A}_i|}{\sqrt{T}}.$$

Choice of Baselines

How does one choose a baseline, given that we want these to be good estimates of the individual counterfactual values? A common choice of the baseline in policy gradient algorithms is the mean value of the state, which is learned online (Mnih *et al.* 2016). Inspired by this, we choose a similar quantity: the average expected value $\bar{\hat{u}}_i(I_i, a)$. That is, in addition to accumulating regret for each I, average expected values are also tracked.

While a direct average can be tracked, we found that an exponentially-decaying average that places heavier weight on more recent samples to be more effective in practice. On the k^{th} visit to I at iteration t,

$$\bar{\hat{u}}^k_i(I_i, a) = \begin{cases} 0 & \text{if } k = 0 \\ (1-\alpha)\bar{\hat{u}}^{k-1}_i(I_i, a) + \alpha \hat{u}^b_i(\sigma^t, I_i, a) & \text{if } k > 0 \end{cases}$$

We then define the baseline $b_i(I_i, a) = \bar{\hat{u}}_i(I_i, a)$, and

$$\hat{b}_i(I_i, a|z) = \begin{cases} b_i(I_i, a)/\xi(I_i, a) & \text{if } ha \sqsubseteq z, h \in I_i \\ 0 & \text{otherwise.} \end{cases}$$

The baseline can therefore be thought as *local* to I_i since it depends only on quantities defined and tracked at I_i. Note that $\mathbb{E}_{a\sim\xi(I_i)}[\hat{b}_i(I_i, a|z)] = b_i(I_i, a)$ as required.

Summary of the Full Algorithm

We now summarize the technique developed above. One iteration of the algorithm consists of:

1. Repeat the steps below for each $i \in \mathcal{N} - \{c\}$.
2. Sample a trajectory $z \sim \xi$.
3. For each history $h \sqsubseteq z$ in reverse order (longest first):
   (a) If h is terminal, simply return $u_i(h)$
   (b) Obtain current strategy $\sigma(I)$ from Eq. 4 using cumulative regrets $R(I, a)$ where $h \in I$.
   (c) Use the child value $\hat{u}^b_i(\sigma, ha)$ to compute $\hat{u}^b_i(\sigma, h)$ as in Eq. 9.
   (d) If $\tau(h) = i$ then for $a \in A(I)$, compute $\hat{v}^b_i(\sigma, I, a) = \frac{\pi_{-i}(h)}{q(h)} \hat{u}^b_i(\sigma, ha)$ and accumulate regrets $R(I, a) \leftarrow R(I, a) + \hat{v}^b_i(\sigma, I, a) - \hat{v}^b_i(\sigma, I)$.
   (e) Update $\bar{\hat{u}}(\sigma, I_i, a)$.
   (f) Finally, return $\hat{u}^b_i(\sigma, h)$.

Note that the original outcome sampling is an instance of this algorithm. Specifically, when $b_i(I_i, a) = 0$, then $\hat{v}^b_i(\sigma, I, a) = \tilde{v}_i(\sigma, I, a)$. Step by step example of the computation is in the appendix.

Experimental Results

We evaluate the performance of our method on **Leduc poker** (Southey *et al.* 2005), a commonly used benchmark poker game. Players have an unlimited number of chips, and the deck has six cards, divided into two suits of three identically-ranked cards. There are two rounds of betting; after the first round a single public card is revealed from the deck. Each player antes 1 chip to play, receiving one private card. There are at most two bet or raise actions per round, with a fixed size of 2 chips in the first round, and 4 chips in the second round.

For the experiments, we use a vectorized form of CFR that applies regret updates to each information set consistent with the public information. The first vector variants were introduced in (Johanson *et al.* 2012), and have been used in DeepStack and Libratus (Moravčík *et al.* 2017; Brown and Sandholm 2017). See the appendix for more detail on the implementation. Baseline average values $\bar{\hat{u}}^b_i(I, a)$ used a decay factor of $\alpha = 0.5$. We used a uniform sampling in all our experiments, $\xi(I, a) = \frac{1}{|A(I)|}$.

We also consider the best case performance of our algorithm by using the oracle baseline. It uses baseline values of the true counterfactual values. We also experiment with and without CFR+, demonstrating that our technique allows the CFR+ to be for the first time efficiently used with sampling.

Convergence

We compared MCCFR, MCCFR+, VR-MCCFR, VR-MCCFR+, and VR-MCCFR+ with the oracle baseline, see Fig. 3. The variance-reduced VR-MCCFR and VR-MCCFR+ variants converge significantly faster than plain MCCFR. Moreover, the speedup grows as the baseline improves during the computation. A similar trend is shown by both VR-MCCFR and VR-MCCFR+, see Fig. 4. MCCFR needs hundreds of millions of iterations to reach the same exploitability as VR-MCCFR+ achieves in one million iterations: a 250-times speedup. VR-MCCFR+ with the oracle baseline significantly outperforms VR-MCCFR+ at the start of the computation, but as time progresses and the learned baseline improves, the difference shrinks. After one million iterations, exploitability of VR-MCCFR+ with a learned baseline approaches the exploitability of VR-MCCFR+ with the oracle baseline. This oracle baseline result gives a bound on the gains we can get by constructing better learned baselines.

Observed Variance

To verify that the observed speedup of the technique is due to variance reduction, we experimentally observed variance of counterfactual value estimates for MCCFR+ and MCCFR, see Fig. 5. We did that by sampling 1000 alternative trajectories for all visited information sets, with each trajectory sampling a different estimate of the counterfactual value. While the variance of value estimates in the plain algorithm seems to be more or less constant, the variance of VR-MCCFR and VR-MCCFR+ value estimates is lower, and continues to decrease as more iterations are run. This confirms that the combination of baseline and bootstrapping is reducing variance, which implies better performance given the connection between variance and MCCFR's performance (Theorem 1).

Evaluation of Bootstrapping and Baseline Dependence on Actions

Recent work that evaluates action-dependent baselines in RL (Tucker *et al.* 2018), shows that there is often no real advantage compared to baselines that depend just on the state. It is also not common to bootstrap the value estimates in RL. Since VR-MCCFR uses both of these techniques it is natural to explore the contribution of each idea. We compared four VR-MCCFR+ variants: with or without bootstrapping and with baseline that is state or state-action dependant, see Fig. 6. The conclusion is that the improvement in the performance is very small unless we use both bootstrapping and an action-dependant baseline.

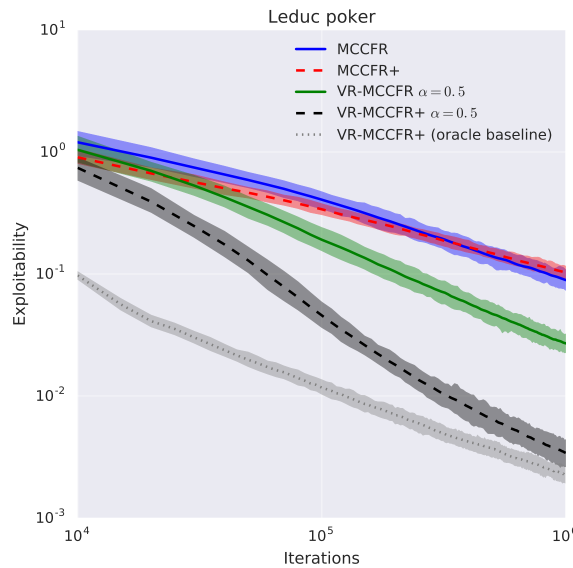


Figure 3: Convergence of exploitability for different MCCFR variants on logarithmic scale. VR-MCCFR converges substantially faster than plain MCCFR. VR-MCCFR+ bring roughly two orders of magnitude speedup. VR-MCCFR+ with oracle baseline (actual true values are used as baselines) is used as a bound for VR-MCCFR's performace to show possible room for improvement. When run for 10^6 iterations VR-MCCFR+ approaches performance of the oracle version. The ribbons show 5th and 95th percentile over 100 runs.

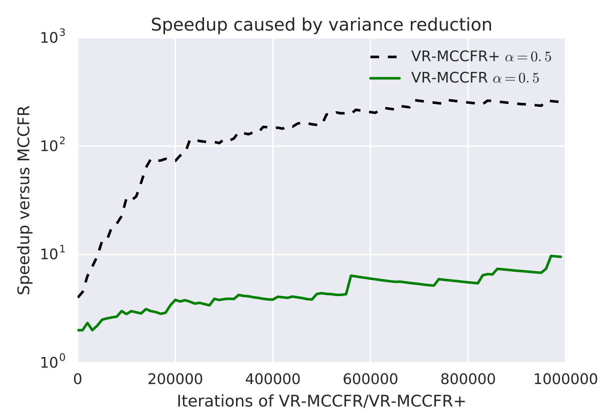


Figure 4: Speedup of VR-MCCFR and VR-MCCFR+ compared to plain MCCFR. Y-axis show how many times more iterations are required by MCCFR to reach the same exploitability as VR-MCCFR or VR-MCCFR+.

Conclusions

We have presented a new technique for variance reduction for Monte Carlo counterfactual regret minimization. This technique has close connections to existing RL methods of

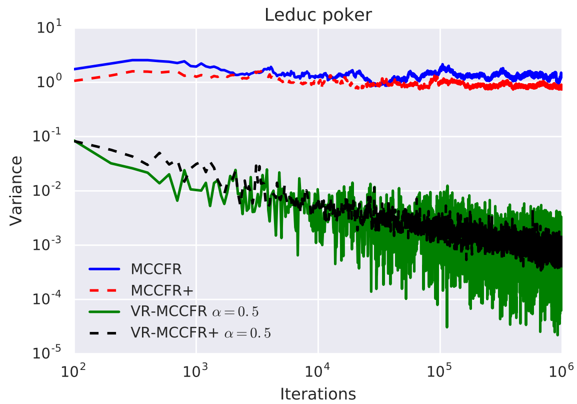


Figure 5: Variance of counterfactual values in VR-MCCFR and plain MCCFR with both regret matching and regret matching+. The curves were smoothed by computing moving average over a sliding window of 100 iterations.

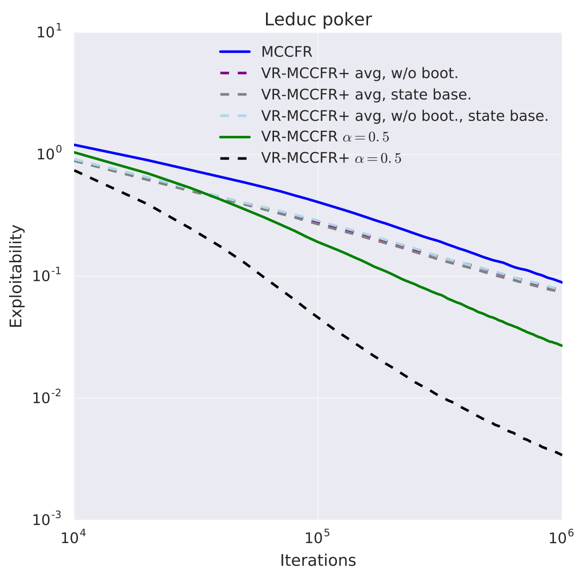


Figure 6: Detailed comparison of different VR-MCCFR variants on logarithmic scale. The curves for MCCFR, VR-MCCFR and VR-MCCFR+ are the same as in the previous plot, the other lines show how the algorithm performs when using state baselines instead of state-action baselines, and without bootstrapping. All of these reduced variants perform better than plain MCCFR, however they are worse than full VR-MCCFR. This ablation study shows that the combination of all VR-MCCFR features is important for final performance.

state and state-action baselines. In contrast to RL environments, our experiments in imperfect information games suggest that state-action baselines are superior to state baselines. Using this technique, we show that empirical variance is indeed reduced, speeding up the convergence by an order of magnitude. The decreased variance allows for the first time CFR+ to be used with sampling, bringing the speedup to two orders of magnitude. Finally, the technique requires only a relatively small computational overhead. In the experiments on Leduc using our non-optimized implementation, we observed a factor of 2 per-iteration slowdown.

References

[Arjona-Medina *et al.* 2018] Jose A. Arjona-Medina, Michael Gillhofer, Michael Widrich, Thomas Unterthiner, and Sepp Hochreiter. Rudder: Return decomposition for delayed rewards. *CoRR*, abs/1806.07857, 2018.

[Bansal *et al.* 2018] Trapit Bansal, Jakub Pachocki, Szymon Sidor, Ilya Sutskever, and Igor Mordatch. Emergent complexity via multi-agent competition. In *Proceedings of the Sixth International Conference on Learning Representations*, 2018.

[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.

[Boyle 1977] Phelim P Boyle. Options: A monte carlo approach. *Journal of financial economics*, 4(3):323–338, 1977.

[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.

[Burch *et al.* 2014] Neil Burch, Michael Johanson, and Michael Bowling. Solving imperfect information games using decomposition. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

[Burch *et al.* 2018] Neil Burch, Martin Schmid, Matej Moravcik, Dustin Morill, and Michael Bowling. Aivat: A new variance reduction technique for agent evaluation in imperfect information games, 2018.

[Burch 2017] Neil Burch. *Time and Space: Why Imperfect Information Games are Hard*. PhD thesis, University of Alberta, 2017.

[Foerster *et al.* 2017] Jakob N. Foerster, Richard Y. Chen, Maruan Al-Shedivat, Shimon Whiteson, Pieter Abbeel, and Igor Mordatch. Learning with opponent-learning awareness. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, 2017.

[Gibson *et al.* 2012] Richard Gibson, Marc Lanctot, Neil Burch, Duane Szafron, and Michael Bowling. Generalized sampling and variance in counterfactual regret minimization. In *Proceedings of the Twenty-Sixth Conference on Artificial Intelligence (AAAI-12).*, pages 1355–1361, 2012.

[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 *et al.* 2015] Johannes Heinrich, Marc Lanctot, and David Silver. Fictitious self-play in extensive-form games. In *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*, 2015.

[Johanson *et al.* 2011] Michael Johanson, Michael Bowling, Kevin Waugh, and Martin Zinkevich. Accelerating best response calculation in large extensive games. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI)*, pages 258–265, 2011.

[Johanson *et al.* 2012] Michael Johanson, Nolan Bard, Marc Lanctot, Richard Gibson, and Michael Bowling. Efficient nash equilibrium approximation through Monte Carlo counterfactual regret minimization. In *Proceedings of the Eleventh International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, 2012.

[Kuhn poker 2018] Kuhn poker. Kuhn poker — Wikipedia, the free encyclopedia, 2018. [Online; accessed 28-August-2018].

[Lanctot *et al.* 2009] Marc Lanctot, Kevin Waugh, Martin Zinkevich, and Michael Bowling. Monte Carlo sampling for regret minimization in extensive games. In Y. Bengio, D. Schuurmans, J. Lafferty, C. K. I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 1078–1086, 2009.

[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, University of Alberta, University of Alberta, Computing Science, 116 St. and 85 Ave., Edmonton, Alberta T6G 2R3, June 2013.

[Littman 1994] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *In Proceedings of the Eleventh International Conference on Machine Learning*, pages 157–163. Morgan Kaufmann, 1994.

[Liu *et al.* 2018] Hao Liu, Yihao Feng, Yi Mao, Dengyong Zhou, Jian Peng, and Qiang Liu. Action-dependent control variates for policy optimization via stein identity. 2018.

[Mnih *et al.* 2016] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 1928–1937, 2016.

[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.

[Owen 2013] Art B. Owen. *Monte Carlo theory, methods and examples*. 2013.

[Pepels *et al.* 2014] Tom Pepels, Mandy J.W. Tak, Marc Lanctot, and Mark H.M. Winands. Quality-based rewards for Monte-Carlo tree search simulations. In *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI)*, 2014.

[Schulman *et al.* 2017] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

[Southey *et al.* 2005] Finnegan Southey, Michael H. Bowling, Bryce Larson, Carmelo Piccione, Neil Burch, Darse Billings, and D. Chris Rayner. Bayes' bluff: Opponent modelling in poker. In *UAI '05, Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence*, 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 (NIPS) 31*, 2018.

[Sutton and Barto 2017] R. Sutton and A. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition, 2017. Draft, in progress.

[Tammelin *et al.* 2015] Oskari Tammelin, Neil Burch, Michael Johanson, and Michael Bowling. Solving heads-up limit texas hold'em. In *Proceedings of the 24th International Joint Conference on Artificial Intelligence*, 2015.

[Tucker *et al.* 2018] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. *arXiv preprint arXiv:1802.10031*, 2018.

[Veness *et al.* 2011] Joel Veness, Marc Lanctot, and Michael Bowling. Variance reduction in Monte-Carlo tree search. In J. Shawe-Taylor, R.S. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 1836–1844, 2011.

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

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

[Wu *et al.* 2018] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M. Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. *CoRR*, abs/1803.07246, 2018.

[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.

Appendices

MCCFR and MCCFR+ comparison

While it is known in that MCCFR+ is outperformed by MC-CFR (Burch 2017), we are not aware on any explicit comparison of these two algorithms in literature. Fig. 7 shows experimental evaluation of these two techniques on Leduc poker.

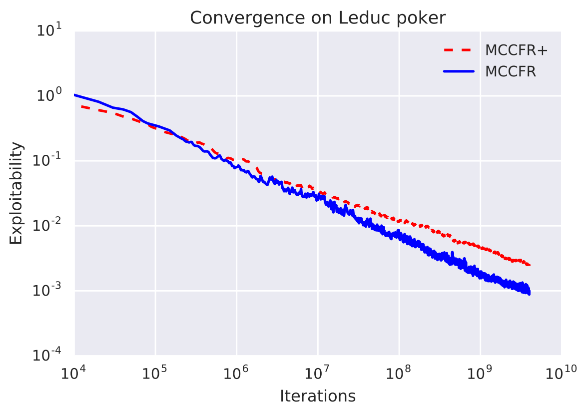


Figure 7: Convergence of MCCFR and MCCFR+ on logarithmic scale. For the first 10^6 iterations, MCCFR+ performs simillary to the MCCFR. After approximately 10^7 iterations, the difference in favor of MCCFR starts to be visible and the gap in exploitability widens as the number of iterations grows.

Vector Form CFR and Augmented Information Sets

The first appearance of the vector form was presented in (Johanson *et al.* 2011). In this paper, the best response computation, needed to compute exploitability, was sped-up by redefining the computation using the notion of a public tree. At the heart of a public tree is the notion of a **public state** which contains a set of information sets whose histories are consistent with the public information revealed so far (Johanson *et al.* 2011, Definition 2). This allowed the method to compute quantities for all information sets consistent with a public state at once (stored in vectors) and operations to compute them could be vectorized during a traversal of the public tree. There are also game-specific optimizations that could be applied at leaf nodes to asymptotically reduce the total computation necessary.

A similar construction was used in several sampling variants introduced in (Johanson *et al.* 2012). Here, instead of computing necessary for best response, counterfactual values were vectorized and stored instead. The paper describes several ways to sample at various types of chance nodes (ones which reveal public information, or private information to each player), but the concept of a vectorized form of CFR was general. In fact, a vector form of vanilla CFR is possible in any game: when traversing down the tree, these vectors store the probability of reaching each information set (called a *range* in (Moravčík *et al.* 2017)) and return vectors of counterfactual values. Both DeepStack and Libratus used vector forms of CFR and CFR+ in No-Limit poker.

Our experiments use a MCCFR variant of the public tree CFR methods above. Given some sampled sequence of public actions, we update all information sets consistent with that sequence. For example in Leduc poker, six trajectories per player are considered, one for every possible private player card, which all share the same sequence of public actions.

In this public tree form, it is often useful to consider the counterfactual values for the player that is not acting. For example, in poker it might be my turn to act, but I want a baseline value estimate of how well my opponent would do with any possible hand, given the publicly visible betting and board cards. Whereas standard information sets only partition states according to the acting player, augmented information sets (Burch *et al.* 2014) partition the states according to the player and that player's information. Two states are said to be in the same player p augmented information set if both states passed through the same sequence of player p information sets and made the same player p actions. In poker, these are states where player p's private card and the public tree node (the betting and board cards) are the same.

In our VR-MCCFR implementation, the baselines values are kept as vectors at each public state, each representing a baseline for the augmented information sets corresponding to the public state. Also, the average values tracked are counterfactual and normalized by the range. So, for example in Leduc, for five information sets in some public state, $(I_1, I_2, \ldots, I_5)$, quantity tracked by the baseline at this public state for action a is:

$$\frac{\hat{v}_i^b(\sigma, I_k, a)}{\sum_{k'} \pi_{opp}^{\sigma}(I_{k'}^{opp})},$$

where π_{opp}^{σ} is the reach probability of the opponent only (excluding chance), and I^{opp} refers to the augmented information set belonging to the opponent at I. Then, when using the baseline values to compute the modified counterfactual values, we need to multiply them by the current $\sum_{k'} \pi_{opp}^{\sigma}(I_{k'}^{opp})$ to get the baseline values under the current strategy σ.

Proofs

Proof of Lemma 1

$$\begin{aligned}\mathbb{E}[\hat{v}^b(\sigma, I, a)] &= \mathbb{E}[\hat{v}_i(\sigma, I, a)] - \mathbb{E}[\hat{b}_i(I, a)] + \mathbb{E}[b_i(I, a)] \\ &= v_i(\sigma, I, a) - b_i(I, a) + b_i(I, a) \\ &= v_i(\sigma, I, a).\end{aligned}$$

□

Proof of Lemma 2

We begin by proving a few supporting lemmas regarding local expectations over actions at specific histories:

Lemma 4. *Given some $h \in \mathcal{H}$, for any $z \in \mathcal{Z}$ generated by sampling $\xi : \mathcal{H} \mapsto \mathcal{A}$ and all actions a, $\mathbb{E}_{z\sim\xi}[\hat{u}_i^b(\sigma, h, a|z)] = \sum_{z, ha \sqsubseteq z} q(z)\hat{u}_i^b(\sigma, ha|z)/\xi(h, a)$:*

Proof. $\hat{u}_i^b(\sigma,h,a|z)$ has three cases, from which we get $\mathbb{E}_{z\sim\xi}[\hat{u}_i^b(\sigma,h,a|z)]$

$$\begin{aligned}
&= \sum_{z,ha\sqsubseteq z} q(z) \\
&\quad \left(b_i(I_i(h),a) + \frac{-b_i(I_i(h),a) + \hat{u}_i^b(\sigma,ha|z)}{\xi(h,a)}\right) \\
&+ \sum_{z,h\sqsubset z,ha\not\sqsubseteq z} q(z)(b_i(I_i(h),a)) \\
&+ \sum_{h\not\sqsubseteq z} 0 \\
&= \sum_{z,ha\sqsubseteq z} q(z)\hat{u}_i^b(\sigma,ha|z)/\xi(h,a) \\
&+ (q(ha) - q(ha)/\xi(h,a))b_i(I_i(h),a) \\
&+ q(h)(1-\xi(h,a))b_i(I_i(h),a) \\
&= \sum_{z,ha\sqsubseteq z} q(z)\hat{u}_i^b(\sigma,ha|z)/\xi(h,a)
\end{aligned}$$

□

Lemma 5. *Given some $h \in \mathcal{H}$, for any $z \in \mathcal{Z}$ generated by sampling $\xi : \mathcal{H} \mapsto \mathcal{A}$, the local baseline-enhanced estimate is an unbiased estimate of expected values for all actions a:*

$$\mathbb{E}_{z\sim\xi}[\hat{u}_i^b(\sigma,h,a|z)] = \mathbb{E}_{z\sim\xi}[\hat{u}_i(\sigma,h,a|z)].$$

Proof. We prove this by induction on the maximum distance from ha to any terminal. The base case is $ha \in \mathcal{Z}$. $\mathbb{E}_{z\sim\xi}[\hat{u}_i^b(\sigma,h,a|z)]$

$$\begin{aligned}
&= \sum_{z,ha\sqsubseteq z} q(z)\hat{u}_i^b(\sigma,ha|z)/\xi(h,a) \quad \text{by Lemma 4} \\
&= \sum_{z,ha\sqsubseteq z} q(z)\hat{u}_i(\sigma,ha|z)/\xi(h,a) \quad \text{by Eq. 10} \\
&= \mathbb{E}_{z\sim\xi}[\hat{u}_i(\sigma,h,a|z)] \quad \text{by Eq. 7, 8}
\end{aligned}$$

Now assume for $i \geq 0$ that the lemma property holds for all $h'a'$ that are at most $j \leq i$ steps from a terminal. Consider history ha being $i+1$ steps from some terminal, which implies that $ha \notin \mathcal{Z}$. We have $\mathbb{E}_{z\sim\xi}[\hat{u}_i^b(\sigma,h,a|z)]$

$$\begin{aligned}
&= \sum_{z,ha\sqsubseteq z} q(z)\hat{u}_i^b(\sigma,ha|z)/\xi(h,a) \quad \text{by Lemma 4} \\
&= \sum_{z,ha\sqsubseteq z} q(z)\sum_{a'}\sigma(ha,a')\hat{u}_i^b(\sigma,ha,a'|z)/\xi(h,a) \\
&\quad \text{by Eq. 10} \\
&= \sum_{z,ha\sqsubseteq z} q(z)\sum_{a'}\sigma(ha,a')\hat{u}_i(\sigma,ha,a'|z)/\xi(h,a) \\
&\quad \text{by assumption} \\
&= \sum_{z,ha\sqsubseteq z} q(z)\hat{u}_i(\sigma,ha|z)/\xi(h,a) \quad \text{by Eq. 8} \\
&= \mathbb{E}_{z\sim\xi}[\hat{u}_i(\sigma,h,a|z)] \quad \text{by Eq. 7}
\end{aligned}$$

The lemma property holds for distance $i+1$, and so by induction the property holds for all h and a. □

Lemma 6. *Given some $h \in \mathcal{H}$, for any $z \in \mathcal{Z}$ generated by sampling $\xi : \mathcal{H} \mapsto \mathcal{A}$ and for all actions a, the local baseline-enhanced estimate is an unbiased estimate of the original sampled counterfactual value: $\mathbb{E}_{z\sim\xi}[\hat{v}_i^b(\sigma,I_i(h),a|z)] = \mathbb{E}_{z\sim\xi}[\tilde{v}_i(\sigma,I_i(h),a|z)]$.*

Proof. First, $\mathbb{E}_{z\sim\xi}[\hat{v}_i^b(\sigma,I_i(h),a|z)]$

$$\begin{aligned}
&= \mathbb{E}_{z\sim\xi}\left[\frac{\pi_{-i}^\sigma(h)}{q(h)}\hat{u}_i^b(\sigma,h,a|z)\right] \quad \text{by Eq. 11} \\
&= \frac{\pi_{-i}^\sigma(h)}{q(h)}\mathbb{E}_{z\sim\xi}[\hat{u}_i^b(\sigma,h,a|z)] \\
&= \frac{\pi_{-i}^\sigma(h)}{q(h)}\mathbb{E}_{z\sim\xi}[\hat{u}_i(\sigma,h,a|z)] \quad \text{by Lemma 5} \\
&= \mathbb{E}_{z\sim\xi}[\tilde{v}_i(\sigma,I_i(h),a|z)] \quad \text{by Eq. 5, 7.}
\end{aligned}$$

□

Proof of Lemma 2. The proof now follows directly: $\mathbb{E}_{z\sim\xi}[\hat{v}_i^b(\sigma,I,a|z)]$

$$\begin{aligned}
&= \mathbb{E}_{z\sim\xi}[\tilde{v}_i(\sigma,I,a|z)] \quad \text{by Lemma 6} \\
&= v_i(\sigma,I,a) \quad \text{by (Lanctot \textit{et al.} 2009, Lemma 1).}
\end{aligned}$$

□

Proof of Lemma 3

We start by proving that given an oracle baseline, the baseline-enhanced expected value is always equal to the true expected value, and therefore has zero variance.

Lemma 7. *Using an oracle baseline defined over histories, $b_i^*(h,a) = u_i^\sigma(ha)$, then for all z such that $h \sqsubseteq z$, $\hat{u}_i^{b^*}(\sigma,h,a|z) = u_i^\sigma(ha)$.*

Proof. Similar to above, we prove this by induction on the maximum distance from ha to z. The base case is $ha \in \mathcal{Z}$. By assumption $h \sqsubseteq z$ so we have $\hat{u}_i^{b^*}(\sigma,h,a|z)$

$$\begin{aligned}
&= \begin{cases} b_i^*(h,a) + \frac{\hat{u}_i^{b^*}(\sigma,ha|z) - b_i^*(h,a)}{\xi(h,a)} & \text{if } ha = z \\ b_i^*(h,a) & \text{otherwise} \end{cases} \\
&\quad \text{by Eq. 9} \\
&= \begin{cases} u_i^\sigma(ha) + \frac{u_i^\sigma(ha) - u_i^\sigma(ha)}{\xi(h,a)} & \text{if } ha = z \\ u_i^\sigma(ha) & \text{otherwise} \end{cases} \\
&\quad \text{by Eq. 10 and definition of } b_i^*(h,a) \\
&= u_i^\sigma(ha)
\end{aligned}$$

Now assume for $i \geq 0$ that the lemma property holds for all $h'a'$ that are at most $j \leq i$ steps from a terminal. Consider history ha being $i+1$ steps from some terminal, which implies $ha \notin \mathcal{Z}$. We have

$$\hat{u}_i^{b^*}(\sigma,ha|z) = u_i^\sigma(ha) \tag{12}$$

because $\hat{u}_i^{b^*}(\sigma,ha|z)$

$$\begin{aligned}
&= \sum_{a'}\sigma(ha,a')\hat{u}_i^{b^*}(\sigma,ha,a'|z) \quad \text{by Eq. 10} \\
&= \sum_{a'}\sigma(ha,a')u_i^\sigma(haa') \quad \text{by assumption} \\
&= u_i^\sigma(ha) \quad \text{by definition of } u_i^\sigma
\end{aligned}$$

We now look at $\hat{u}_i^{b^*}(\sigma, h, a|z)$

$$
\begin{aligned}
&= \begin{cases} u_i^\sigma(ha) + \frac{\hat{u}_i^{b^*}(\sigma,ha|z) - u_i^\sigma(ha)}{\xi(h,a)} & \text{if } ha \sqsubset z \\ u_i^\sigma(ha) & \text{otherwise} \end{cases} \\
&\quad \text{by Eq. 9 and definition of } b_i^*(h,a) \\
&= \begin{cases} u_i^\sigma(ha) + \frac{u_i^\sigma(ha) - u_i^\sigma(ha)}{\xi(h,a)} & \text{if } ha \sqsubset z \\ u_i^\sigma(ha) & \text{otherwise} \end{cases} \\
&\quad \text{by Eq. 12} \\
&= u_i^\sigma(ha)
\end{aligned}
$$

The lemma property holds for distance $i+1$, and so by induction the property holds for all h and a. □

Proof of Lemma 3. Given z such that $h \sqsubseteq z$, we have $\hat{v}_i^*(\sigma, h, a|z)$

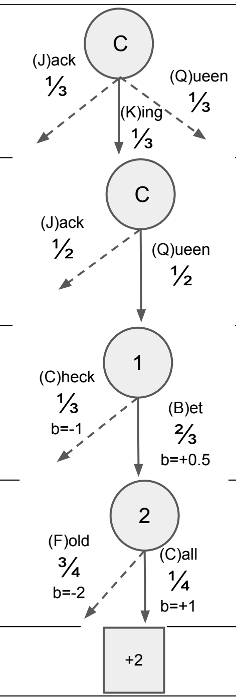
$$
\begin{aligned}
&= \frac{\pi_{-i}^\sigma(h)}{q(h)} \hat{u}_i^{b^*}(\sigma, h, a|z) \quad \text{by Eq. 11} \\
&= \frac{\pi_{-i}^\sigma(h)}{q(h)} u_i^\sigma(ha) \quad \text{by Lemma 7}
\end{aligned}
$$

None of the terms above depend on z, and so we have $\mathbb{V}\text{ar}_{h,z\sim\xi,h\in I,h\sqsubseteq z}[\hat{v}_i^*(\sigma, h, a|z)] = 0$. Note as well that $\pi_{-i}^\sigma(h) u_i^\sigma(ha)$ corresponds to the terms in the summation of Equation 2, so abusing notation, we have $\hat{v}_i^*(\sigma, h, a|z) = v_i(\sigma, h, a)/q(h)$: the counterfactual value of taking action a at h, with an importance sampling weight to correct for the likelihood of reaching h. □

In MCCFR, the optimal baseline b^* is not known, as it would require traversing the entire tree, taking away any advantages of sampling. However, b^* can be approximated (learned online), which motivates the choice for tracking its average value presented in the main part of the paper.

Kuhn Example

In this section, we present a step-by-step example of one iteration of the algorithm on Kuhn poker (Kuhn poker 2018). Kuhn poker is a simplified version of poker with three cards and is therefore suitable for demonstration purposes. Table 1 show forward pass of VR-MCCFR algorithm, Table 2 shows backward pass.



	Forward pass					
	h History	Game tree trajectory	$\pi^{\sigma}_{-1}(h)$ Reach prob.	$q(h)$ Sampling prob.	$I_1 = I_1(h)$ Infoset for Pl1	$I_2 = I_2(h)$ Infoset for Pl2
Forward pass	$\emptyset$		1	1	$\emptyset$	$\emptyset$
	K		$\frac{1}{3}$	$\frac{1}{3}$	K	?
	KQ		$\frac{1}{6}$	$\frac{1}{6}$	K?	?Q
	KQB		$\frac{1}{6}$	$\frac{1}{12}$	K?B	?QB
	KQBC		$\frac{1}{24}$	$\frac{1}{24}$	K?BC	?QBC

Table 1: Detailed example of updates computed for player 1 in Kuhn poker during forward pass of the algorithm. Backward pass that uses these values is shown in Table 2. In our representation history h is a concatenation of all public and private actions. The game tree trajectory column shows the path in the game tree that was sampled. Solid arrows denote sampled actions while dashed arrows show other available actions, all actions have their probability under current strategy σ next to them. The sampled history in this case is: chance deals (K)ing to player 1, chance deals (Q)ueen to player 2, player 1 (B)ets, player 2 (C)alls. We will use shorter notation $KQBC$ to refer to this history. For each history h reach probability $\pi^{\sigma}_{-1}(h)$ shows how likely the history is reached when player 1 plays in a way to get to this history. The sampling probabilities $q(h)$ are computed following sampling policy ξ which is uniform in this case, i.e. for each history all available actions have the same probability that they will be sampled. The last two columns show augmented information sets for each player in each history. For example for player 1 history KQB is represented by information set K?B since he does not know what card was dealt to PLAYER 2. Light gray background marks cells where the values are well defined however they are not used in our example update for player 1.

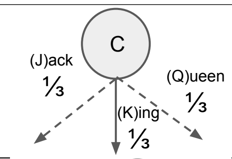
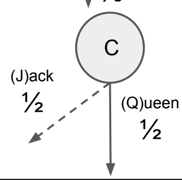
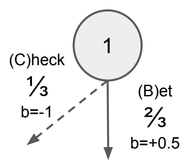
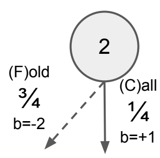
	h History	Game tree trajectory	Backward pass $\hat{u_1}^b(\sigma, h, a \mid z)$ Sampled corrected history-action utility	$\hat{u_1}^b(\sigma, h \mid z)$ Sampled corrected history utility	$\hat{v_1}^b(\sigma, I_1, a \mid z)$ Sampled corrected cf-value
Def.			Eq. 9	Eq. 10	Eq. 11
Backward pass	$\emptyset$				
	K				
	KQ		$\mathbf{\hat{u_1}^b(\sigma, h, B \mid z)} = \frac{\hat{u_1}^b(\sigma, hB \mid z) - b(I_1, B)}{\xi(h, B)} + b(I_1, B) = \frac{-\frac{3}{4} - 0.5}{\frac{1}{2}} + 0.5 = -2$ $\mathbf{\hat{u_1}^b(\sigma, h, C \mid z)} = b(I_1, C) = -1$	$\mathbf{\hat{u}_1^b(\sigma, h \mid z)} = \sum_a \sigma(h, a)\hat{u}_1^b(\sigma, h, a \mid z) = \frac{1}{3} * (-1) + \frac{2}{3} * (-2) = -\frac{5}{3}$	$\mathbf{\hat{v}_1^b(\sigma, I_1, B \mid z)} = \frac{\pi^{\sigma}_{-1}(h)}{q(h)}\hat{u}_1^b(\sigma, h, B \mid z) = \frac{\frac{1}{6}}{\frac{1}{6}} * (-2) = -2$ $\mathbf{\hat{v}_1^b(\sigma, I_1, C \mid z)} = \frac{\pi^{\sigma}_{-1}(h)}{q(h)}\hat{u}_1^b(\sigma, h, C \mid z) = \frac{\frac{1}{6}}{\frac{1}{6}} * (-1) = -1$
	KQB		$\mathbf{\hat{u_1}^b(\sigma, h, C \mid z)} = \frac{\hat{u_1}^b(\sigma, hC \mid z) - b(I_1, c)}{\xi(h, C)} + b(I_1, C) = \frac{2-1}{\frac{1}{2}} + 1 = 3$ $\mathbf{\hat{u_1}^b(\sigma, h, F \mid z)} = b(I_1, F) = -2$	$\mathbf{\hat{u}_1^b(\sigma, h \mid z)} = \sum_a \sigma(h, a)\hat{u}_1^b(\sigma, h, a \mid z) = \frac{3}{4} * (-2) + \frac{1}{4} * 3 = -\frac{3}{4}$	
	KQBC	$\boxed{+2}$		$\mathbf{\hat{u_1}^b(\sigma, h, \mid z)} = u_1(h) = 2$	

Table 2: The backward pass starts by evaluating utility of the terminal history: $\hat{u_1}^b(\sigma, KQBC \mid KQBC) = +2$ since player 1 has (K)ing which is better card than opponent's (Q)ueen. In the next step computation updates values for history KQB. Expected baseline corrected history-action value $\hat{u_1}^b(\sigma, KQB, Call \mid KQBC)$ is computed based on current sample and then used together with $\hat{u_1}^b(\sigma, KQB, Fold \mid KQBC)$ to compute $\hat{u_1}^b(\sigma, KQB \mid KQBC)$. When updating values for history KQ baseline corrected sampled counterfactual values are computed based on just updated $\hat{u_1}^b(\sigma, KQ, Bet \mid KQBC)$ for the sampled Bet action and on a baseline value $\hat{u_1}^b(\sigma, KQ, Check \mid KQBC)$ for Check action that was not sampled. Reach probability $\pi^{\sigma}_{-1}(KQ)$ and sampling probability $q(KQ)$ that are also needed to compute counterfactual-values $\hat{v_1}^b(\sigma, K?, a \mid KQBC)$ were already computed in the forward pass. The counterfactual values are then used to compute actions' regrets (Eq. 3) which is not shown in the table. Values in cell with light gray background are not used in computation of $\hat{v_1}^b(\sigma, K?, a \mid KQBC)$.