Algorithms and Bounds for Rollout Sampling Approximate Policy Iteration[⋆]

Christos Dimitrakakis[1] and Michail G. Lagoudakis[2]

[1] Informatics Institute, University of Amsterdam, Amsterdam, The Netherlands
`dimitrak@science.uva.nl`
[2] Department of ECE, Technical University of Crete, Chania 73100, Greece
`lagoudakis@intelligence.tuc.gr`

Abstract. Several approximate policy iteration schemes without value functions, which focus on policy representation using classifiers and address policy learning as a supervised learning problem, have been proposed recently. Finding good policies with such methods requires not only an appropriate classifier, but also reliable examples of best actions, covering the state space sufficiently. Up to this time, little work has been done on appropriate covering schemes and on methods for reducing the sample complexity of such methods, especially in continuous state spaces. This paper focuses on the simplest possible covering scheme (a discretized grid over the state space) and performs a sample-complexity comparison between the simplest (and previously commonly used) rollout sampling allocation strategy, which allocates samples equally at each state under consideration, and an almost as simple method, which allocates samples only as needed and requires significantly fewer samples.

1 Introduction

Supervised and reinforcement learning are two well-known learning paradigms, which have been researched mostly independently. Recent studies have investigated using mature supervised learning methods for reinforcement learning [9, 6, 10, 7]. Initial results have shown that policies can be approximately represented using multi-class classifiers and therefore it is possible to incorporate classification algorithms within the inner loops of several reinforcement learning algorithms [9, 6, 7]. This viewpoint allows the quantification of the performance of reinforcement learning algorithms in terms of the performance of classification algorithms [10]. While a variety of promising combinations become possible through this synergy, heretofore there have been limited practical results and widely-applicable algorithms.

Herein we consider approximate policy iteration algorithms, such as those proposed by Lagoudakis and Parr [9] as well as Fern et al. [6, 7], which do not explicitly represent a value function. At each iteration, a new policy/classifier is

[⋆] This project was partially supported by the ICIS-IAS project and the Marie Curie International Reintegration Grant MCIRG-CT-2006-044980 awarded to Michail G. Lagoudakis within the 6th European Framework Programme.

produced using training data obtained through extensive simulation (rollouts) of the previous policy on a generative model of the process. These rollouts aim at identifying better action choices over a subset of states in order to form a set of data for training the classifier representing the improved policy. The major limitation of these algorithms, as also indicated by Lagoudakis and Parr [9], is the large amount of rollout sampling employed at each sampled state. It is hinted, however, that great improvement could be achieved with sophisticated management of sampling. We have verified this intuition in a companion paper [4] that experimentally compared the original approach of uninformed uniform sampling with various intelligent sampling techniques. That paper employed heuristic variants of well-known algorithms for bandit problems, such as Upper Confidence Bounds [1] and Successive Elimination [5], for the purpose of managing rollouts (choosing which state to sample from is similar to choosing which lever to pull on a bandit machine). It should be noted, however, that despite the similarity, rollout management has substantial differences to standard bandit problems and thus general bandits results are not directly applicable to our case.

The current paper aims to offer a first theoretical insight into the rollout sampling problem. This is done through the analysis of the two simplest sample allocation methods described in [4]. Firstly, the old method that simply allocates an equal, fixed number of samples at each state and secondly the slightly more sophisticated method of progressively sampling all states where we are not yet reasonably certain of which the policy-improving action would be.

The remainder of the paper is organised as follows. Section 2 provides the necessary background, Section 4 introduces the proposed algorithms, and Section 3 discusses related work. Section 5, which contains an analysis of the proposed algorithms, is the main technical contribution.

2 Preliminaries

A *Markov Decision Process* (MDP) is a 6-tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma, D)$, where $\mathcal{S}$ is the state space of the process, $\mathcal{A}$ is a finite set of actions, P is a Markovian transition model ($P(s, a, s')$ denotes the probability of a transition to state s' when taking action a in state s), R is a reward function ($R(s, a)$ is the expected reward for taking action a in state s), $\gamma \in (0, 1]$ is the discount factor for future rewards, and D is the initial state distribution. A *deterministic policy* π for an MDP is a mapping $\pi : \mathcal{S} \mapsto \mathcal{A}$ from states to actions; $\pi(s)$ denotes the action choice at state s. The value $V^\pi(s)$ of a state s under a policy π is the expected, total, discounted reward when the process begins in state s and all decisions at all steps are made according to π:

$$V^\pi(s) = E\left[\sum_{t=0}^{\infty} \gamma^t R\big(s_t, \pi(s_t)\big) | s_0 = s, s_t \sim P\right] \ . \tag{1}$$

The goal of the decision maker is to find an optimal policy π^* that maximises the expected, total, discounted reward from all states; in other words, $V^{\pi^*}(s) \geq V^\pi(s)$ for all policies π and all states $s \in \mathcal{S}$.

Policy iteration (PI) is an efficient method for deriving an optimal policy. It generates a sequence π_1, π_2, ..., π_k of gradually improving policies, which terminates when there is no change in the policy ($\pi_k = \pi_{k-1}$); π_k is an optimal policy. Improvement is achieved by computing V^{π_i} analytically (solving the linear Bellman equations) and the action values

$$Q^{\pi_i}(s,a) = R(s,a) + \gamma \sum_{s'} P(s,a,s')V^{\pi_i}(s') \ ,$$

and then determining the improved policy as $\pi_{i+1}(s) = \arg\max_a Q^{\pi_i}(s,a)$.

Policy iteration typically terminates in a small number of steps. However, it relies on knowledge of the full MDP model, exact computation and representation of the value function of each policy, and exact representation of each policy. *Approximate policy iteration* (API) is a family of methods, which have been suggested to address the "curse of dimensionality", that is, the huge growth in complexity as the problem grows. In API, value functions and policies are represented approximately in some compact form, but the iterative improvement process remains the same. Apparently, the guarantees for monotonic improvement, optimality, and convergence are compromised. API may never converge, however in practice it reaches good policies in only a few iterations.

2.1 Rollout estimates

Typically, API employs some representation of the MDP model to compute the value function and derive the improved policy. On the other hand, the Monte-Carlo estimation technique of *rollouts* provides a way of accurately estimating Q^π at any given state-action pair (s,a) without requiring an explicit MDP model or representation of the value function. Instead, a generative model of the process (a simulator) is used; such a model takes a state-action pair (s,a) and returns a reward r and a next state s' sampled from $R(s,a)$ and $P(s,a,s')$ respectively.

A rollout for the state-action pair (s,a) amounts to simulating a single trajectory of the process beginning from state s, choosing action a for the first step, and choosing actions according to the policy π thereafter up to a certain horizon T. If we denote the sequence of collected rewards during the i-th simulated trajectory as $r_t^{(i)}$, $t = 0,1,2,\dots,T-1$, then the rollout estimate $\hat{Q}_K^{\pi,T}(s,a)$ of the true state-action value function $Q^\pi(s,a)$ is the observed total discounted reward, averaged over all K trajectories:

$$\hat{Q}_K^{\pi,T}(s,a) \triangleq \frac{1}{K}\sum_{i=1}^{K} \tilde{Q}_{(i)}^{\pi,T}(s,a) \ , \qquad \tilde{Q}_{(i)}^{\pi,T}(s,a) \triangleq \sum_{t=0}^{T-1} \gamma^t r_t^{(i)} \ .$$

Similarly, we define $Q^{\pi,T}(s,a) = \mathbf{E}\left(\sum_{t=0}^{T-1} \gamma^{t-1} r_t \middle| a_0{=}a, s_0{=}s, a_t \sim \pi, s_t \sim P\right)$ to be the actual state-action value function up to horizon T. As will be seen later, with a sufficient amount of rollouts and a long horizon T, we can create an improved policy π' from π at any state s, without requiring a model of the MDP.

3 Related work

Rollout estimates have been used in the Rollout Classification Policy Iteration (RCPI) algorithm [9], which has yielded promising results in several learning domains. However, as stated therein, it is sensitive to the distribution of training states over the state space. For this reason it is suggested to draw states from the discounted future state distribution of the improved policy. This tricky-to-sample distribution, also used by Fern et al. [7], yields better results. One explanation advanced in those studies is the reduction of the potential mismatch between the training and testing distributions of the classifier.

However, in both cases, and irrespectively of the sampling distribution, the main drawback is the excessive computational cost due to the need for lengthy and repeated rollouts to reach a good level of accuracy in the estimation of the value function. In our preliminary experiments with RCPI, it has been observed that most of the effort is spent where the action value differences are either non-existent, or so fine that they require a prohibitive number of rollouts to identify them. In this paper, we propose and analyse sampling methods to remove this performance bottle-neck. By restricting the sampling distribution to the case of a uniform grid, we compare the fixed allocation algorithm (Fixed) [9, 7], whereby a large fixed amount of rollouts is used for estimating the action values in each training state, to a simple incremental sampling scheme based on counting (Count), where the amount of rollouts in each training state varies. We then derive complexity bounds, which show a clear improvement using Count that depends only on the structure of differential value functions.

We note that Fern et al. [7] presented a related analysis. While they go into considerably more depth with respect to the classifier, their results are not applicable to our framework. This is because they assume that there exists some real number $\Delta^* > 0$ which lower-bounds the amount by which the value of an optimal action(s) under any policy exceeds the value of the nearest sub-optimal action in any state s. Furthermore, the algorithm they analyse uses a fixed number of rollouts at each sampled state. For a given minimum Δ^* value over all states, they derive the necessary number of rollouts per state to guarantee an improvement step with high probability, but the algorithm offers no practical way to guarantee a high probability improvement. We instead derive error bounds for the fixed and counting allocation algorithms. Additionally, we are considering continuous, rather than discrete, state spaces. Because of this, technically our analysis is much more closely related to that of Auer et al. [2].

4 Algorithms to reduce sampling cost

The total sampling cost depends on the balance between the number of states sampled and the number of samples per state. In the fixed allocation scheme [9, 7], the same number of $K|\mathcal{A}|$ rollouts is allocated to each state in a subset S of states and all K rollouts dedicated to a single action are exhausted before moving on to the next action. Intuitively, if the desired outcome (superiority of

Algorithm 1 SampleState

```
Input: state $s$, policy $\pi$, horizon $T$, discount factor $\gamma$
for (each $a \in \mathcal{A}$) do
    $(s', r) = $ Simulate$(s, a)$
    $\tilde{Q}^\pi(s, a) = r$
    $x = s'$
    for $t = 1$ to $T - 1$ do
        $(x', r) = $ Simulate$(x, \pi(x))$
        $\tilde{Q}^\pi(s, a) = \tilde{Q}^\pi(s, a) + \gamma^t r$
        $x = x'$
    end for
end for
return $\tilde{Q}^\pi$
```

some action) in some state can be confidently determined early, there is no need to exhaust all $K|\mathcal{A}|$ rollouts available in that state; the training data could be stored and the state could be removed from the pool without further examination. Similarly, if we can confidently determine that all actions are indifferent in some state, we can simply reject it without wasting any more rollouts; such rejected states could be replaced by fresh ones which might yield meaningful results. These ideas lead to the following question: can we examine all states in S collectively in some interleaved manner by selecting each time a single state to focus on and allocating rollouts only as needed?

Selecting states from the state pool could be viewed as a problem akin to a multi-armed bandit problem, where each state corresponds to an arm. Pulling a lever corresponds to sampling the corresponding state once. By *sampling a state* we mean that we perform a single rollout for each action in that state as shown in Algorithm 1. This is the minimum amount of information we can request from a single state.[3] Thus, the problem is transformed to a *variant* of the classic multi-armed bandit problem. Several methods have been proposed for various versions of this problem, which could potentially be used in this context. In this paper, apart from the fixed allocation scheme presented above, we also examine a simple counting scheme.

The algorithms presented here maintain an empirical estimate $\hat{\Delta}^\pi(s)$ of the marginal difference of the apparently maximal and the second best of actions. This can be represented by the marginal difference in Q^π values in state s, defined as

$$\Delta^\pi(s) = Q^\pi(s, a^*_{s,\pi}) - \max_{a \neq a^*_{s,\pi}} Q^\pi(s, a) \ ,$$

where $a^*_{s,\pi}$ is the action that maximises Q^π in state s:

$$a^*_{s,\pi} = \arg\max_{a \in \mathcal{A}} Q^\pi(s, a) \ .$$

[3] It is possible to also manage sampling of the actions, but herein we are only concerned with the effort saved by managing state sampling.

The case of multiple equivalent maximising actions can be easily handled by generalising to sets of actions in the manner of Fern et al. [7], in particular

$$\begin{aligned} A^*_{s,\pi} &\triangleq \{a \in \mathcal{A} : Q^\pi(s,a) \geq Q^\pi(s,a'),\ \forall a' \in \mathcal{A}\} \\ V^\pi_*(s) &= \max_{a \in \mathcal{A}} Q^\pi(s,a) \\ \Delta^\pi(s) &= \begin{cases} V^\pi_*(s) - \max_{a \notin A^*_{s,\pi}} Q^\pi(s,a), & A^*_{s,\pi} \subset \mathcal{A} \\ 0, & A^*_{s,\pi} = \mathcal{A} \end{cases} \end{aligned}$$

However, here we discuss only the single best action case to simplify the exposition. The estimate $\hat{\Delta}^\pi(s)$ is defined using the empirical value function $\hat{Q}^\pi(s,a)$.

5 Complexity of sampling-based policy improvement

Rollout algorithms can be used for policy improvement under certain conditions. Bertsekas [3] gives several theorems for policy iteration using rollouts and an approximate value function that satisfies a consistency property. Specifically, Proposition 3.1. therein states that the one-step look-ahead policy π' computed from the approximate value function $\hat{V}^\pi$, has a value function which is better than the current approximation $\hat{V}^\pi$, if $\max_{a \in \mathcal{A}} \mathbf{E}[r_{t+1} + \gamma \hat{V}^\pi(s_{t+1}) | \pi', s_t = s, a_t = a] \geq \hat{V}^\pi(s)$ for all $s \in \mathcal{S}$. It is easy to see that an approximate value function that uses only sampled trajectories from a fixed policy π satisfies this property if we have an adequate number of samples. While this assures us that we can perform rollouts at any state in order to improve upon the given policy, it does not lend itself directly to policy iteration. That is, with no way to compactly represent the resulting rollout policy we would be limited to performing deeper and deeper tree searches in rollouts.

In this section we shall give conditions that allow policy iteration through compact representation of rollout policies via a grid and a finite number of sampled states and sample trajectories with a finite horizon. Following this, we will analyse the complexity of the fixed sampling allocation scheme employed in [9, 7] and compare it with an oracle that needs only one sample to determine $a^*_{s,\pi}$ for any $s \in \mathcal{S}$ and a simple counting scheme.

5.1 Sufficient conditions

Assumption 1 (Bounded finite-dimension state space) *The state space $\mathcal{S}$ is a compact subset of $[0,1]^d$.*

This assumption can be generalised to other bounded state spaces easily. However, it is necessary to have this assumption in order to be able to place some minimal constraints on the search.

Assumption 2 (Bounded rewards) *$R(s,a) \in [0,1]$ for all $a \in \mathcal{A}$, $s \in \mathcal{S}$.*

This assumption bounds the reward function and can also be generalised easily to other bounding intervals.

Assumption 3 (Hölder Continuity) *For any policy $\pi \in \Pi$, there exists $L, \alpha \in [0, 1]$, such that for all states $s, s' \in \mathcal{S}$*

$$|Q^\pi(s, a) - Q^\pi(s', a)| \leq \frac{L}{2} \|s - s'\|_\infty^\alpha \ .$$

This assumption ensures that the value function Q^π is fairly smooth. It trivially follows in conjunction with Assumptions 1 and 2 that Q^π, Δ^π are bounded *everywhere* in $\mathcal{S}$ if they are bounded for at least one $s \in \mathcal{S}$. Furthermore, the following holds:

Remark 1. Given that, by definition, $Q^\pi(s, a^*_{s,\pi}) \geq \Delta^\pi(s) + Q^\pi(s, a)$ for all $a \neq a^*_{s,\pi}$, it follows from Assumption 3 that

$$Q^\pi(s', a^*_{s,\pi}) \geq Q^\pi(s', a) \ ,$$

for all $s' \in \mathcal{S}$ such that $\|s - s'\|_\infty \leq \sqrt[\alpha]{\Delta^\pi(s)/L}$.

This remark implies that the best action in some state s according to Q^π will also be the best action in a neighbourhood of states around s. This is a reasonable condition as there would be no chance of obtaining a reasonable estimate of the best action in any region from a single point, if Q^π could change arbitrarily fast. We assert that MDPs with a similar smoothness property on their transition distribution will also satisfy this assumption.

Finally, we need an assumption that limits the total number of rollouts that we need to take, as states with a smaller Δ^π will need more rollouts.

Assumption 4 (Measure) *If $\mu\{S\}$ denotes the Lebesgue measure of set S, then, for any $\pi \in \Pi$, there exist $M, \beta > 0$ such that $\mu\{s \in \mathcal{S} : \Delta^\pi(s) < \epsilon\} < M\epsilon^\beta$ for all $\epsilon > 0$.*

This assumption effectively limits the amount of times value-function changes lead to best-action changes, as well as the ratio of states where the action values are close. This assumption, together with the Hölder continuity assumption, imposes a certain structure on the space of value functions. We are thus guaranteed that the value function of any policy results in an improved policy which is not arbitrarily complex. This in turn, implies that an optimal policy cannot be arbitrarily complex either.

A final difficulty is determining whether there exists some sufficient horizon T_0 beyond which it is unnecessary to go. Unfortunately, even though for any state s for which $Q^\pi(s, a') > Q^\pi(s, a)$, there exists $T_0(s)$ such that $Q^{\pi,T}(s, a') > Q^{\pi,T}(s, a)$ for all $T > T_o(s)$, T_0 grows without bound as we approach a point where the best action changes. However, by selecting a fixed, sufficiently large rollout horizon, we can still behave optimally with respect to the true value function in a compact subset of $\mathcal{S}$.

Lemma 1. *For any policy $\pi \in \Pi$, $\epsilon > 0$, there exists a finite $T_\epsilon > 0$ and a compact subset $\mathcal{S}_\epsilon \subset \mathcal{S}$ such that*

$$Q^{\pi,T}(s, a^*_{s,\pi}) \geq Q^{\pi,T}(s, a) \quad \forall a \in \mathcal{A}, s \in \mathcal{S}, T > T_\epsilon$$

where $a^*_{s,\pi} \in \mathcal{A}$ is such that $Q^\pi(s, a^*_{s,\pi}) \geq Q^\pi(s,a)$ for all $a \in \mathcal{A}$.

Proof. From the above assumptions it follows directly that for any $\epsilon > 0$, there exists a compact set of states $\mathcal{S}_\epsilon \subset \mathcal{S}$ such that $Q^\pi(s, a^*_{s,\pi}) \geq Q^\pi(s, a') + \epsilon$ for all $s \in \mathcal{S}_\epsilon$, with $a' = \arg\max_{a \neq a^*_{s,\pi}} Q^\pi(s,a)$. Now let $x_T \triangleq Q^{\pi,T}(s, a^*_{s,\pi}) - Q^{\pi,T}(s,a')$. Then, $x_\infty \triangleq \lim_{T\to\infty} x_T \geq \epsilon$. For any $s \in \mathcal{S}_\epsilon$ the limit exists and thus by definition $\exists T_\epsilon(s)$ such that $x_{T_\epsilon} > 0$ for all $T > T_\epsilon$. Since $\mathcal{S}_\epsilon$ is compact, $T_\epsilon \triangleq \sup_{s\in\mathcal{S}_\epsilon} T_\epsilon(s)$ also exists.[4] $\square$

This ensures that we can identify the best action within ϵ, using a finite rollout horizon, in most of $\mathcal{S}$. Moreover, $\mu\{\mathcal{S}_\epsilon\} \geq 1 - M2\epsilon^\beta$ from Assumption 4.

In standard policy iteration, the improved policy π' over π has the property that the improved action in any state is the action with the highest Q^π value in that state. However, in rollout-based policy iteration, we may only guarantee being within $\epsilon > 0$ of the maximally improved policy.

Definition 1 (ϵ-improved policy). *An ϵ-improved policy π' derived from π satisfies*

$$\max_{a\in\mathcal{A}} Q^\pi(s,a) - \epsilon \leq V^{\pi'}(s), \tag{2}$$

Such a policy will be said to be *improving in S* if $V^\pi(s) \leq V^{\pi'}(s)$ for all $s \in S$. The measure of states for which there can not be improvement is limited by Assumption 4. Finding an improved π' for the whole of $\mathcal{S}$ is in fact not possible in finite time, since this requires determining the boundaries in $\mathcal{S}$ at which the best action changes. [5]

In all cases, we shall attempt to find the improving action $a^*_{s,\pi}$ at each state s on a uniform grid of n states, with the next policy $\pi'(s')$ taking the estimated best action $\hat{a}^*_{s,\pi}$ for the state s closest to s', i.e. it is a nearest-neighbour classifier.

In the remainder, we derive complexity bounds for achieving an ϵ-improved policy π' from π with probability at least $1-\delta$. We shall always assume that we are using a sufficiently deep rollout to cover $\mathcal{S}_\epsilon$ and only consider the number of rollouts performed. First, we shall derive the number of states we need to sample from in order to guarantee an ϵ-improved policy, under the assumption that at each state we have an *oracle* which can give us the exact Q^π values for each state we examine. Later, we shall consider sample complexity bounds for the case where we do not have an oracle, but use empirical estimates $\hat{Q}^{\pi,T}$ at each state.

5.2 The Oracle algorithm

Let $B(s,\rho)$ denote the infinity-norm sphere of radius ρ centred in s and consider Alg. 2 (Oracle) that can instantly obtain the state-action value function for any point in $\mathcal{S}$. The algorithm creates a uniform grid of n states, such that

[4] For a discount factor $\gamma < 1$ we can simply bound T_ϵ with $\log[\epsilon(1-\gamma)]/\log(\gamma)$.

[5] To see this, consider $\mathcal{S} \triangleq [0,1]$, with some $s^* : R(s,a_1) \geq R(s,a_2)\ \forall s \geq s^*$ and $R(s,a_1) < R(s,a_2)\ \forall s < s^*$. Finding s^* requires a binary search, at best.

Algorithm 2 ORACLE

```
Input: $n$, $\pi$
Set $S$ to a uniform grid of $n$ states in $\mathcal{S}$.
for $s \in S$ do
    $\hat{a}^*_{s,\pi} = a^*_{s,\pi}$
end for
return $\hat{A}^*_{S,\pi} \triangleq \{\hat{a}^*_{s,\pi} : s \in S\}$
```

the distance between adjacent states is $2\rho = \frac{1}{n^{1/d}}$ – and so can cover $\mathcal{S}$ with spheres $B(s, \rho)$. Due to Assumption 3, the error in the action values of any state in sphere $B(s, \rho)$ of state s will be bounded by $L\left(\frac{1}{2n^{1/d}}\right)^{\alpha}$. Thus, the resulting policy will be $L\left(\frac{1}{2n^{1/d}}\right)^{\alpha}$-improved, i.e. this will be the maximum regret it will suffer over the maximally improved policy.

To bound this regret by ϵ, it is sufficient to have $n = \left(\frac{1}{2}\sqrt[\alpha]{\frac{L}{\epsilon}}\right)^{d}$ states in the grid. The following proposition follows directly.

Proposition 1. *Algorithm 2 results in regret ϵ for $n = \mathcal{O}\left(L^{d/\alpha}\left[2\epsilon^{1/\alpha}\right]^{-d}\right)$.*

Furthermore, as for all s such that $\Delta^{\pi}(s) > L\rho^{\alpha}$, $a^*_{s,\pi}$ will be the improved action in all of $B(s, \rho)$, then π' will be improving in S with $\mu\{S\} \geq 1 - ML^{\beta}\left(\frac{1}{2n^{1/d}}\right)^{\alpha\beta}$. Both the regret and the lack of complete coverage are due to the fact that we cannot estimate the best-action boundaries with arbitrary precision in finite time. When using rollout sampling, however, even if we restrict ourselves to ϵ improvement, we may still make an error due to both the limited number of rollouts and the finite horizon of the trajectories. In the remainder, we shall derives error bounds for two practical algorithms that employ a fixed grid with a finite number of T-horizon rollouts.

5.3 Error bounds for states

When we estimate the value function at each $s \in S$ using rollouts there is a probability that the estimated best action $\hat{a}^*_{s,\pi}$ is not in fact the best action. For any given state under consideration, we can apply the following well-known lemma to obtain a bound on this error probability

Lemma 2 (Hoeffding inequality). *Let X be a random variable in $[b, b+Z]$ with $\bar{X} \triangleq \mathbf{E}[X]$, observed values $X_1, \ldots, X_n$ of X, and $\hat{X}_n \triangleq \frac{1}{n}\sum_{i=1}^{n} X_i$. Then, $\mathbf{P}(\hat{X}_n \geq \bar{X} + \epsilon) = \mathbf{P}(\hat{X}_n \leq \bar{X} + \epsilon) \leq \exp\left(-2n\epsilon^2/Z^2\right)$ for any $\epsilon > 0$.*

Without loss of generality, consider two random variables $X, Y \in [0, 1]$, with empirical means $\hat{X}_n, \hat{Y}_n$ and empirical difference $\hat{\Delta}_n \triangleq \hat{X}_n - \hat{Y}_n > 0$. Their means and difference will be denoted as $\bar{X}, \bar{Y}, \bar{\Delta} \triangleq \bar{X} - \bar{Y}$ respectively.

Note that if $\bar{X} > \bar{Y}$, $\hat{X}_n > \bar{X} - \bar{\Delta}/2$ and $\hat{Y}_n < \bar{Y} + \bar{\Delta}/2$ then necessarily $\hat{X}_n > \hat{Y}_n$, so $\mathbf{P}(\hat{X}_n > \hat{Y}_n | \bar{X} > \bar{Y}) \geq \mathbf{P}(\hat{X}_n > \bar{X} - \bar{\Delta}/2 \wedge \hat{Y}_n < \bar{Y} + \hat{\Delta}_n/2)$. The

Algorithm 3 FIXED

```
Input: $n$, $\pi$, $c$, $T$, $\delta$
Set $S$ to a uniform grid of $n$ states in $\mathcal{S}$.
for $s \in S$ do
    Estimate $\hat{Q}_c^{\pi,T}(s,a)$ for all $a$.
    if $\hat{\Delta}^\pi(s) > Z\sqrt{\frac{2\log(2n|\mathcal{A}|/\delta)}{c}}$ then
        $\hat{a}^*_{s,\pi} = \arg\max \hat{Q}^\pi$
    else
        $\hat{a}^*_{s,\pi} = \pi(s)$
    end if
end for
return $\hat{A}^*_{S,\pi} \triangleq \{\hat{a}^*_{s,\pi} : s \in S\}$
```

converse is

$$\mathbf{P}\left(\hat{X}_n < \hat{Y}_n \mid \bar{X} > \bar{Y}\right) \leq \mathbf{P}\left(\hat{X}_n < \bar{X} - \bar{\Delta}/2 \vee \hat{Y}_n > \bar{Y} + \bar{\Delta}/2\right) \tag{3a}$$

$$\leq \mathbf{P}\left(\hat{X}_n < \bar{X} - \bar{\Delta}/2\right) + \mathbf{P}\left(\hat{Y}_n > \bar{Y} + \bar{\Delta}/2\right) \tag{3b}$$

$$\leq 2\exp\left(-\frac{n}{2}\bar{\Delta}^2\right). \tag{3c}$$

Now, consider $\hat{a}^*_{s,\pi}$ such that $\hat{Q}^\pi(s,\hat{a}^*_{s,\pi}) \geq \hat{Q}^\pi(s,a)$ for all a. Setting $\hat{X}_n = Z^{-1}\hat{Q}^\pi(s,\hat{a}^*_{s,\pi})$ and $\hat{Y}_n = Z^{-1}\hat{Q}^\pi(s,a)$, where Z is a normalising constant such that $Q \in [b, b+1]$, we can apply (3). Note that the bound is largest for the action a' with value closest to $a^*_{s,\pi}$, for which it holds that $Q^\pi(s,a^*_{s,\pi}) - Q^\pi(s,a') = \Delta^\pi(s)$. Using this fact and an application of the union bound, we conclude that for any state s, from which we have taken $c(s)$ samples, it holds that:

$$\mathbf{P}[\exists \hat{a}^*_{s,\pi} \neq a^*_{s,\pi} : \hat{Q}^\pi(s,\hat{a}^*_{s,\pi}) \geq \hat{Q}^\pi(s,a)] \leq 2|\mathcal{A}|\exp\left(-\frac{c(s)}{2Z^2}\Delta^\pi(s)^2\right). \tag{4}$$

5.4 Uniform sampling: the FIXED algorithm

As we have seen in the previous section, if we employ a grid of n states, covering $\mathcal{S}$ with spheres $B(s,\rho)$, where $\rho = \frac{1}{2n^{1/d}}$, and taking action $a^*_{s,\pi}$ in each sphere centred in s, then the resulting policy π' is only guaranteed to be improved within ϵ of the optimal improvement from π, where $\epsilon = L\rho^\alpha$. Now, we examine the case where, instead of obtaining the true $a^*_{s,\pi}$, we have an estimate $\hat{a}^*_{s,\pi}$ arising from c samples from each action in each state, for a total of $cn|\mathcal{A}|$ samples. Algorithm 3 accepts (i.e. it sets $\hat{a}^*_{s,\pi}$ to be the empirically highest value action in that state) for all states satisfying:

$$\hat{\Delta}^\pi(s) \geq Z\sqrt{\frac{2\log(2n|\mathcal{A}|/\delta)}{c}}. \tag{5}$$

The condition ensures that the probability that $Q^\pi(s, \hat{a}^*_{s,\pi}) < Q^\pi(s, a^*_{s,\pi})$, meaning the optimally improving action is not $\hat{a}^*_{s,\pi}$, at any state is at most δ. This can easily be seen by substituting the right hand side of (5) for ϵ in (4). As $\varDelta^\pi(s) > 0$, this results in an error probability of a single state smaller than δ/n and we can use a union bound to obtain an error probability of δ for each policy improvement step.

For each state $s \in S$ that the algorithm considers, the following two cases are of interest: (a) $\varDelta^\pi(s) < \epsilon$, meaning that even when we have correctly identified $a^*_{s,\pi}$, we are still not improving over all of $B(s, \rho)$ and (b) $\varDelta^\pi(s) \geq \epsilon$.

While the probability of accepting the wrong action is always bounded by δ, we must also calculate the probability that we fail to accept an action at all, when $\varDelta^\pi(s) \geq \epsilon$ to estimate the expected regret. Restating our acceptance condition as $\hat{\varDelta}^\pi(s) \geq \theta$, this is given by:

$$\begin{aligned}\mathbf{P}[\hat{\varDelta}^\pi(s) < \theta] &= \mathbf{P}[\hat{\varDelta}^\pi(s) - \varDelta^\pi(s) < \theta - \varDelta^\pi(s)] \\ &= \mathbf{P}[\varDelta^\pi(s) - \hat{\varDelta}^\pi(s) > \varDelta^\pi(s) - \theta], \quad \varDelta^\pi(s) > \theta. \end{aligned} \tag{6}$$

Is $\varDelta^\pi(s) > \theta$? Note that for $\varDelta^\pi(s) > \epsilon$, if $\epsilon > \theta$ then so is $\varDelta^\pi$. So, in order to achieve total probability δ for all state-action pairs in this case, after some calculations, we arrive at this expression for the regret

$$\epsilon = \max\left\{ L\left(\frac{1}{2n^{1/d}}\right)^\alpha, Z\sqrt{\frac{8\log(2n|\mathcal{A}|/\delta)}{c}} \right\}. \tag{7}$$

By equating the two sides, we get an expression for the minimum number of samples necessary per state:

$$c = 8\frac{Z^2}{L^2}4^\alpha n^{2\alpha/d}\log(2n|\mathcal{A}|/\delta).$$

This directly allows us to state the following proposition.

Proposition 2. *The sample complexity of Algorithm 3 to achieve regret at most ϵ with probability at least $1-\delta$ is $\mathcal{O}\left(\epsilon^{-2}L^{d/\alpha}\left[2\epsilon^{1/\alpha}\right]^{-d}\log\frac{2|\mathcal{A}|}{\delta}L^{d/\alpha}\left[2\epsilon^{1/\alpha}\right]^{-d}\right)$.*

5.5 The Count algorithm

The Count algorithm starts with a policy π and a set of states S_0, with $n = |S_0|$. At each iteration k, each sample in S_k is sampled once. Once a state $s \in S_k$ contains a dominating action, it is removed from the search. So,

$$S_k = \left\{ s \in S_{k-1} : \hat{\varDelta}^\pi(s) < Z\sqrt{\frac{\log(2n|\mathcal{A}|/\delta)}{c(s)}} \right\}$$

Thus, the number of samples from each state is $c(s) \geq k$ if $s \in S_k$.

We can apply similar arguments to analyse Count, by noting that the algorithm spends less time in states with higher $\varDelta^\pi$ values. The measure assumption

Algorithm 4 Count

```
Input: $n$, $\pi$, $C$, $T$, $\delta$
Set $S_0$ to a uniform grid of $n$ states in $\mathcal{S}$, $c_1, \ldots, c_n = 0$.
for $k = 1, 2, \ldots$ do
  for $s \in S_k$ do
    Estimate $\hat{Q}_c^{\pi,T}(s, a)$ for all $a$, increment $c(s)$
    $S_k = \left\{s \in S_{k-1} : \hat{\Delta}^\pi(s) < Z\sqrt{\frac{2\log(2n|\mathcal{A}|/\delta)}{c(s)}}\right\}$
  end for
  if $\sum_s c(s) >= C$ then
    Break.
  end if
end for
```

then allows us to calculate the number of states with large Δ^π and thus, the number of samples that are needed.

We have already established that there is an upper bound on the regret depending on the grid resolution $\epsilon < L\rho^\alpha$. We proceed by forming subsets of states $W_m = \{s \in S : \Delta^\pi(s) \in [2^{-m}, 2^{1-m}\}$. Note that we only need to consider $m < 1 + \frac{1}{\log 1/2}(\log L + \alpha \log \rho)$.

Similarly to the previous algorithm, and due to our acceptance condition, for each state $s \in W_m$, we need $c(s) \geq 2^{2m+1} Z^2 \log \frac{2n|\mathcal{A}|}{\delta}$ in order to bound the total error probability by δ. The total number of samples necessary is

$$Z^2 \log \frac{2n|\mathcal{A}|}{\delta} \sum_{m=0}^{\lceil \frac{1}{\log 1/2}(\log L + \alpha \log \rho)\rceil} |W_m| 2^{2m+1}.$$

A bound on $|W_m|$ is required to bound this expression. Note that

$$\mu\left\{B(s,\rho) : \Delta^\pi(s') < \epsilon \forall s' \in B(s,\rho)\right\} \leq \mu\left\{s : \Delta^\pi(s) < \epsilon\right\} < M\epsilon^\beta. \tag{8}$$

It follows that $|W_m| < M 2^{\beta(1-m)} \rho^{-d}$ and consequently

$$\begin{aligned}\sum_{s \in S} c(s) &= Z^2 \log \frac{2n|\mathcal{A}|}{\delta} \sum_{m=0}^{\lceil \frac{1}{\log 1/2}(\log L + \alpha \log \rho)\rceil} M 2^{\beta(1-m)} \rho^{-d} 2^{2m+1} \\ &\leq M 2^{\beta+1} 2^{\frac{1+\frac{1}{\log 1/2}(\log L + \alpha \log \rho)}{2-\beta}} 2^d Z^2 n \log \frac{2n|\mathcal{A}|}{\delta}.\end{aligned} \tag{9}$$

The above results directly in the following proposition:

Proposition 3. *The sample complexity of Algorithm 4 to achieve regret at most ϵ with probability at least $1 - \delta$, is $\mathcal{O}\left(L^{d/\alpha}\left[2\epsilon^{1/\alpha}\right]^{-d} \log \frac{2|\mathcal{A}|}{\delta} L^{d/\alpha}\left[2\epsilon^{1/\alpha}\right]^{-d}\right)$.*

We note that we are of course not able to remove the dependency on d, which is only due to the use of a grid. Nevertheless, we obtain a reduction in sample complexity of order ϵ^{-2} for this very simple algorithm.

6 Discussion

We have derived performance pounds for approximate policy improvement without a value function in continuous MDPs. We compared the usual approach of sampling equally from a set of candidate states to the slightly more sophisticated method of sampling from all candidate states in parallel, and removing a candidate state from the set as soon as it was clear which action is best. For the second algorithm, we find an improvement of approximately ϵ^{-2}. Our results complement those of Fern et al [7] for relational Markov decision processes. However significant amount of future work remains.

Firstly, we have assumed everywhere that $T > T_\epsilon$. While this may be a relatively mild assumption for $\gamma < 1$, it is problematic for the undiscounted case, as some states would require far deeper rollouts than others to achieve regret ϵ. Thus, in future work we would like to examine sample complexity in terms of the depth of rollouts as well.

Secondly, we would like to extend the algorithms to increase the number of states that we look at: whenever $\hat{V}^\pi(s) \approx \hat{V}^{\pi'}(s)$ for all s, then we could increase the resolution. For example if,

$$\sum_{s \in S} \mathbf{P}\left(\hat{V}^\pi(s) + \epsilon < \hat{V}^{\pi'}(s) \mid V^\pi(s) > V^{\pi'}(s)\right) < \delta$$

then we could increase the resolution around those states with the smallest Δ^π. This would get around the problem of having to select n.

A related point that has not been addressed herein, is the choice of policy representation. The grid-based representation probably makes poor use of the available number of states. For the increased-resolution scheme outlined above, a classifier such as k-nearest-neighbour could be employed. Furthermore, regularised classifiers might affect a smoothing property on the resulting policy, and allow the learning of improved policies from a set of states containing erroneous best action choices.

As far as the state allocation algorithms are concerned, in a companion paper [4], we have compared the performance of COUNT and FIXED with additional allocation schemes inspired from the UCB and successive elimination algorithms. We have found that all methods outperform FIXED in practice, sometimes by an order of magnitude, with the UCB variants being the best overall.

For this reason, in future work we plan to perform an analysis of such algorithms. A further extension to deeper searches, by for example managing the sampling of actions within a state, could also be performed using techniques similar to [8].

6.1 Acknowledgements

Thanks to the reviewers and to Adam Atkinson, Brendan Barnwell, Frans Oliehoek, Ronald Ortner and D. Jacob Wildstrom for comments and useful discussions.

Bibliography

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multi-armed bandit problem. *Machine Learning Journal*, 47(2-3):235–256, 2002.

[2] P. Auer, R. Ortner, and C. Szepesvari. Improved Rates for the Stochastic Continuum-Armed Bandit Problem. *Proceedings of the Conference on Computational Learning Theorey (COLT)*, 2007.

[3] Dimitri Bertsekas. Dynamic programming and suboptimal control: From ADP to MPC. *Fundamental Issues in Control, European Journal of Control*, 11(4-5), 2005. From 2005 CDC, Seville, Spain.

[4] Christos Dimitrakakis and Michail Lagoudakis. Rollout sampling approximate policy iteration. *Machine Learning*, 72(3), September 2008.

[5] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7:1079–1105, 2006.

[6] A. Fern, S. Yoon, and R. Givan. Approximate policy iteration with a policy language bias. *Advances in Neural Information Processing Systems*, 16(3), 2004.

[7] A. Fern, S. Yoon, and R. Givan. Approximate policy iteration with a policy language bias: Solving relational Markov decision processes. *Journal of Artificial Intelligence Research*, 25:75–118, 2006.

[8] Levente Kocsis and Csaba Szepesvári. Bandit based Monte-Carlo planning. In *Proceedings of ECML-2006*, 2006.

[9] Michail G. Lagoudakis and Ronald Parr. Reinforcement learning as classification: Leveraging modern classifiers. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 424–431, Washington, DC, USA, August 2003.

[10] John Langford and Bianca Zadrozny. Relating reinforcement learning performance to classification performance. In *Proceedings of the 22nd International Conference on Machine learning (ICML)*, pages 473–480, Bonn, Germany, 2005.