Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence

Improving Approximate Value Iteration with Complex Returns by Bounding

Robert Wright,[1,2] **Xingye Qiao, Lei Yu***
Binghamton University[1]
Binghamton, NY
{rwright3, xqiao, lyu}@binghamton.edu

Steven Loscalzo
Information Directorate, Air Force Research Lab[2]
Rome, NY
steven.loscalzo@us.af.mil

Abstract

Approximate value iteration (AVI) is a widely used technique in reinforcement learning. Most AVI methods do not take full advantage of the sequential relationship between samples within a trajectory in deriving value estimates, due to the challenges in dealing with the inherent bias and variance in the n-step returns. We propose a bounding method which uses a negatively biased but relatively low variance estimator generated from a complex return to provide a lower bound on the observed value of a traditional one-step return estimator. In addition, we develop a new Bounded FQI algorithm, which efficiently incorporates the bounding method into an AVI framework. Experiments show that our method produces more accurate value estimates than existing approaches, resulting in improved policies.

Introduction

Approximate Value Iteration (AVI) is a popular framework for solving Reinforcement Learning (RL) problems that combines classical value iteration with function approximation (Gordon 1995; Munos 2005; Riedmiller 2005; Antos, Szepesvári, and Munos 2007). As an off-policy approach it can make effective use of all available samples. At the heart of AVI is its 1-step backup update function which makes use of the Bellman optimality equation (Sutton and Barto 1998). AVI uses this update to iteratively refine the estimated values for each of the individual samples it is provided. The accuracy of these sample value estimates determines the final quality of the value function approximation and the overall success of the approach.

Samples in RL domains are most commonly collected in episodic sequences known as trajectories. Given trajectories, the idea behind the 1-step return has been extended by Temporal Difference (TD) (Sutton and Barto 1998) to produce the n-step return estimates which may be subject to different variance and bias than the 1-step return depending on the learning contexts. In on-policy learning settings where the behavior policy (the policy that generated the trajectory) and target policy (the policy being learned) are the same, the idea of n-step returns has been exploited to great effect by the TD(λ) family of algorithms which utilize complex returns (weighted average of all n-step returns) in order to reduce variance and produce a more accurate value estimate than the 1-step return (Sutton and Barto 1998). In some off-policy learning settings where the behavior policy is different from the target policy, importance sampling has been employed to correct the off-policy bias in the n-step returns (Precup, Sutton, and Dasgupta 2001), and shown some successes in policy iteration methods (Geist and Scherrer 2014; Hachiya et al. 2009). This method takes advantage of prior knowledge of both the target policy and behavior policy (both of which are assumed to have non-zero action selection probabilities) to weight each return of a complex backup and reduce the bias of the individual $R^{(n)}$ returns.

Notwithstanding these advances, there has been little progress in exploiting the n-step returns for value iteration. The main reason for this lies in the fact that in the value iteration framework the target policy is always the optimal policy and is unknown, and it is a challenging issue to deal with the off-policy bias of the n-step returns. The importance sampling method suitable for the policy evaluation phase in policy iteration does not apply here, since the required prior knowledge about the target and behavior policies is not available in the value iteration framework. This work takes the first attempt in addressing this challenge.

For this purpose we propose a novel *bounding* method which uses a negatively biased, but relatively low variance *complex return* estimator to provide a lower bound on the value of the sample label obtained from the traditional one-step return. The method is motivated by a statistical observation that a biased estimator with relatively small variance can sometimes provide an effective bound on the value of another estimator to produce a better estimator than either one alone. The novelty of our approach is that we exploit the off-policy bias down the trajectories, instead of trying to correct it as the importance sampling approach would do. In addition, we develop a new Bounded FQI algorithm, which efficiently incorporates the bounding method into an AVI framework. Finally, we provide an empirical analysis of our method on a set of RL benchmarks demonstrating that the bounding method produces more accurate value estimates than existing approaches, resulting in improved policies.

*Yu is the last author who directs Wright's PhD research.

Background

RL problems can be elegantly described within the context of Markov Decision Processes (MDP) (Puterman 2009). An MDP, M, is defined as a 5-tuple, $M = (S, A, P, \mathcal{R}, \gamma)$, where S is a fully observable finite set of states, A is a finite set of possible actions, P is the state transition model such that $P(s'|s,a) \in [0,1]$ describes the probability of transitioning to state s' after taking action a in state s, $\mathcal{R}^a_{s,s'}$ is the expected value of the immediate reward r after taking a in s, resulting in s', and $\gamma \in (0,1)$ is the discount factor on future rewards.

A proposed solution to a MDP comes in the form of a policy, $\pi(s)$. Policies, $\pi : S \mapsto A$, are functions that prescribe which action to take given a particular state, $\pi(s) = a$. Given a policy, the value of any individual state-action pair in M can be inferred. The value of a state-action pair while following π, is given by its Q-value, $Q_\pi(s,a)$, which is defined as:

$$Q_\pi(s,a) = \mathbb{E}_\pi \left[\sum_{i=0}^{\infty} \gamma^i r_{i+1} \middle| s_0 = s, a_0 = a \right]$$

where r_{i+1} is the immediate reward given at time $i+1$.

The overall objective is to derive an optimal policy, π^*, that maximizes the expected long-term discounted value for any state-action pair in M. In order to derive π^*, value based RL approaches attempt to learn an approximation of the optimal Q-function, Q^*, which is defined as the solution to the optimal Bellman equation,

$$Q^*(s,a) = \sum_{s' \in S} P(s'|s,a) \left[\mathcal{R}^a_{s,s'} + \gamma \max_{a' \in A} Q^*(s',a') \right] \quad (1)$$

From this equation π^* can be extracted as $\pi^*(s) = \arg\max_{a \in A} Q^*(s,a)$.

In the RL setting P and $\mathcal{R}$ are unknown and must be learned from samples. Samples are atomic observations of transitions taken from the domain. They are represented by tuples, $(s_t, a_t, s_{t+1}, r_{t+1})$, consisting of a state s_t, an action a_t, the state s_{t+1} transitioned to by taking a_t in s_t, and r_{t+1}, the immediate reward for that transition. Samples are often collected as episodic sequences known as trajectories. Each trajectory, T, is a sequentially ordered collection of observations where, $T = [(s_0, a_0, s_1, r_1), (s_1, a_1, s_2, r_2), \dots]$.

Approximate Value Iteration

AVI derives $\hat{Q}$, an approximation of Q^*, from a fixed finite set of samples. Provided an initial estimate of Q^* called $\hat{Q}_0$, in iteration m AVI arrives at $\hat{Q}_m$ by employing the 1-step backup operator (which we call $R^{(1)}$ return[1]) over the provided samples and the previous estimate $\hat{Q}_{m-1}$:

$$\hat{Q}_m(s_t, a_t) \leftarrow R^{(1)}_t = r_{t+1} + \gamma \max_{a \in A} \hat{Q}_{m-1}(s_{t+1}, a) \,. \quad (2)$$

$R^{(1)}_t$ combines the 1-step observed immediate reward with a greedy choice among all *bootstrapped* estimates of future returns as approximated by $\hat{Q}_{m-1}$. Note that Eq. (2) differs from the 1-step return definition used by TD methods (Sutton and Barto 1998) by application of the max operator to guide action selection.

Function approximation must be used to provide generalization over limited training samples in any non-trivial domain where state-action spaces cannot be explicitly represented. Approximation necessarily introduces errors into the AVI process since the state-action function is now attempting to represent the value of possibly infinitely many points with a model described by a constant number of parameters. By varying the model's parameters or the sample set used in the approximation, we can view each sample's $R^{(1)}$ return as a random variable with both bias and variance. The next section describes how the properties of the bias and variance can be exploited to yield novel AVI methods.

Approach

Statistical Motivation

A Toy Example Let us first consider a simple example by assuming that $\hat{\theta}$ is a degenerated estimator with variance 0 (that is, it is a constant) and is always less than the true value θ. In this simplified situation, it is always of benefit to use $\max(R^{(1)}, \hat{\theta})$ in place of $R^{(1)}$ to estimate the value. The reason is obvious: when $R^{(1)}$ is less than $\hat{\theta}$, then it must be farther away from θ than $\hat{\theta}$, in which case the greater observation $\hat{\theta}$ should be used. In the worst case scenario, that is, $R^{(1)} > \hat{\theta}$ with high probability, it is no harm to use $\max(R^{(1)}, \hat{\theta})$ since it would coincide with $R^{(1)}$.

This simple example shows that sometimes taking the maximum of two estimators can improve over both. We shall attempt to relax the stringent assumptions herein. We may now assume that $\hat{\theta}$ has a positive variance, and is negatively biased (its expectation is less than the value θ), so that with high probability, $\hat{\theta} < \theta$. This is not difficult to achieve, since it is true either when the variance is not too large or when its expectation is small enough (compared to θ.) Again, in this case $\max(R^{(1)}, \hat{\theta})$ will be superior to $R^{(1)}$.

Clearly, there is another side of the story. The magnitude of improvement $\max(R^{(1)}, \hat{\theta})$ makes over $R^{(1)}$ may not be substantial. If $R^{(1)} < \hat{\theta}$ rarely occurs then $\hat{\theta}$ is useless as a bound. This could happen when the bias of $\hat{\theta}$ is too negative. Moreover, $\max(R^{(1)}, \hat{\theta})$ may even be a substantially worse estimator than $R^{(1)}$ if the variance of $\hat{\theta}$ is too large. In this case, the condition that $\hat{\theta} < \theta$ is no longer likely.

The Domain of Improvements We now investigate when $\max(R^{(1)}, \hat{\theta})$ improves $R^{(1)}$. The discussion above suggests that improvements are possible when $\mathrm{Var}(\hat{\theta})$ is small and/or when $\mathbb{E}(\hat{\theta}) - \theta$ is small enough to ensure that $\hat{\theta} < \theta$, but not so small that $\hat{\theta} < R^{(1)}$ always. Precisely when the improvements occur depends on the underlying distribution of $R^{(1)}$ and $\hat{\theta}$. Here we assume both estimators follow normal distributions. Hence the distributions are fully characterized by their expectations and variances respectively. Although

[1] For conciseness of notation, the subscript t is omitted where there is no confusion.

in reality, the true estimators are only approximately normal at best, the analysis conducted here is sufficient to convey the main message.

To show a concrete example, we assume that $\text{bias}(R^{(1)}) = \mathbb{E}(R^{(1)}) - \theta = 0$ and $\text{Var}(R^{(1)}) = 1$. The bias for $\hat{\theta}$ and the standard deviation of $\hat{\theta}$ are chosen from the range $[-1.5, 1] \times (0, 2]$. For each pair of the bias and std, we estimate the mean squared error of the estimators $R^{(1)}$ and $\max(R^{(1)}, \hat{\theta})$ by Monte Carlo integration. We want to identify the set of bias($\hat{\theta}$) and std($\hat{\theta}$), where $\max(R^{(1)}, \hat{\theta})$ improves $R^{(1)}$. In Figure 1, the red solid curve is the boundary where the two estimators are equally good. The domain to the southwest of the red curve is precisely the domain of improvement.

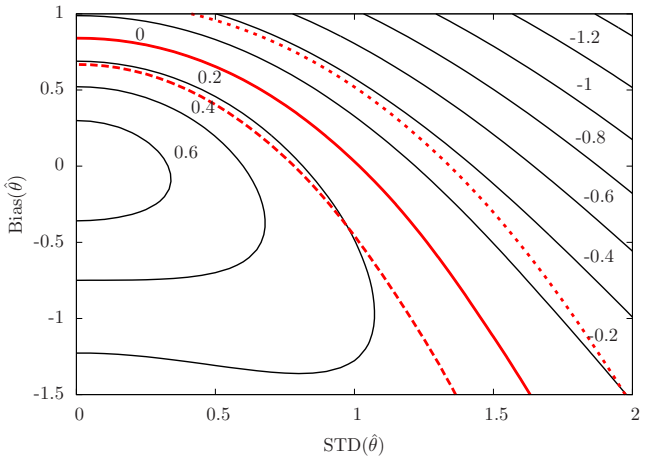

Figure 1: Contour plot showing the improvement of $\max(R^{(1)}, \hat{\theta})$ over $R^{(1)}$. Red curves: boundaries between improvement and no improvement for cases of $\text{bias}(R^{(1)}) =$ 0 (solid), 0.5 (dashed), and -0.5 (dotted).

Also shown in the figure are contours of the log ratio of MSE, $\log(MSE(R^{(1)})/MSE(\max(R^{(1)}, \hat{\theta})))$. The greater this measure is, the more improvement $\max(R^{(1)}, \hat{\theta})$ has. Clearly, the greatest improvement occurs at the unrealistic case where $\hat{\theta}$ is unbiased and has variance 0. Overall, a combination of small bias and small variance guarantees an improvement. More precisely, when the bias of $\hat{\theta}$ is negative, the variance of $\hat{\theta}$ can be greater than that of $R^{(1)}$ ($= 1$ in this case) for the maximum to provide an improvement. The more negative the bias is, the greater variance is allowed. When the bias is overly negative or the variance is too large, then the improvement becomes negligible. On the other hand, even if the bias of $\hat{\theta}$ is positive, there is still a chance for the maximum to be a better estimator. However, this comes with a more stringent assumption that the variance of $\hat{\theta}$ must be much smaller.

We also show the boundaries under the cases where $R^{(1)}$ is biased. The dashed curve and the dotted curve correspond to bias 0.5 and -0.5 respectively. Compared to the solid curve, we have the impression that it is more likely for a maximum estimator such as $\max(R^{(1)}, \hat{\theta})$ to improve $R^{(1)}$, when $R^{(1)}$ is itself negatively biased; and vice versa. This is consistent with the motivation that we want to bound the estimator from below so that it does not negatively deviate from the parameter (recall the simple toy example.) However, there is a potential risk of the bounding estimator $\hat{\theta}$ falls into the domain of loss (i.e., the northeast of the red curve). Such risk increases as $\text{bias}(R^{(1)})$ becomes more positive. Fortunately, $|\text{bias}(R^{(1)})|$ is generally under control in the AVI context. Ideally one would choose a model that fits well and thus does not have much bias without overfitting.

We are now at a position to identify some characteristics about $\hat{\theta}$ that make $\max(R^{(1)}, \hat{\theta})$ better than $R^{(1)}$ and hence make the bounding strategy work.

1. As a bottom line, the expectation of $\hat{\theta}$ needs to be smaller than a positive value τ that satisfies $MSE(\max(R^{(1)}, \tau)) = MSE(R^{(1)})$, or equivalently, $\tau^2\Phi(\tau) + \int_\tau^\infty t^2\phi(t)dt = 1$ in the current example, where $\Phi(t)$ and $\phi(t)$ are the distribution function and density function of standard normal distribution (direct calculation leads to $\tau \approx 0.8399$).
2. The variance of $\hat{\theta}$ is small in general, but can be greater than that of $R^{(1)}$ when the expectation of $\hat{\theta}$ is less than θ.

The first criterion makes sure that taking the maximum (bounding from below) is meaningful. Otherwise an alternative strategy, namely bounding from above, is needed. The second criterion prevents $\hat{\theta}$ from ruining the mean square error of $\max(R^{(1)}, \hat{\theta})$ through large variance.

Moreover, we have a third criterion to make sure that the improvement, when it exists, is substantial: $\hat{\theta}$ is not overly negatively biased. This allows a fair chance for $R^{(1)} < \hat{\theta}$.

It is worth noting that for off-policy n-step returns in AVI context, the expectation generally decreases as n increases. Hence, $\text{bias}(R^{(n)}) < \text{bias}(R^{(n-1)}) < \cdots < \text{bias}(R^{(1)})$ (see detailed discussions on the next subsection). This means if one was to use an n-step return, or a weighted average of many n-step returns, as the bounding estimator, it is more likely to fall into the improvement domain, because the smaller $\text{bias}(\hat{\theta})$ is, the greater variance is allowed, as can be seen in the figure. This motivates our bounding method in the following subsection.

Complex Returns as Bounds

Before we delve into our proposed choice of the bounding estimator, let us consider the n-step returns in the AVI context and discuss their variance and bias properties. Just as the 1-step return $R_t^{(1)}$ (See Eq. (2)) can be used as an estimator for $Q^*(s_t, a_t)$ in value iteration, we can define the n-step returns as follow:

$$R_t^{(n)} = \sum_{i=1}^{n-1} \gamma^{i-1} r_{t+i} + \gamma^n \max_{a \in A} \hat{Q}_{m-1}(s_{t+n}, a) \,. \quad (3)$$

All of the n-step returns are approximations of $Q^*(s_t, a_t)$. Again, the greedy choice by the max operation makes Equation (3) different from the classic n-step return definition used in the TD family of algorithms.

A salient feature of the n-step returns is that their variances increase with n due to the stochastic nature of the MDP. The function approximation variance, which can be a substantial component of the overall variance, is often considered to be roughly the same across different samples. The bias of n-step returns is a more complex issue. Among various types of biases (e.g., off-policy bias, function approximation bias, sampling bias, etc.), the behavior of the off-policy bias is unique. When the target policy is an optimal policy, like in the AVI context, the off-policy bias introduced by a sub-optimal trajectory is strictly negative, and its magnitude increases as more sub-optimal actions are followed down the trajectory. The same observation can be made when treating any state in a trajectory as the starting point for the rest of the trajectory. In contrast, other types of biases, if they exist, can be positive or negative, and often share roughly the same magnitude across different samples. Therefore, when combining the effects of various biases, the expectation of n-step returns generally decreases as n increases, as mentioned at the end of the previous subsection.

Given the above analysis, any of the n-step estimators can potentially fall into the domain of improvement. However, it is difficult to determine whether Conditions 1 and 2 identified earlier are met for each of the individual n-step estimators and which one is the best. Therefore, a logical choice is to consider a weighted average of a number of n-step returns, the so called *complex return*.

The TD(λ) (Sutton and Barto 1998) family of algorithms are the classic ways for deriving complex returns. The TD(λ) method yields the λ-return which is a weighted average of the n-step returns. Under certain assumptions, the λ-return can reduce the variance, enabling more accurate estimates of values in the on-policy settings. The general assumption behind existing complex return methods is that the variance of the n-step returns increases as n increases. Because of this assumption these methods always weight the n-returns with smaller n more heavily in calculating the composite return. Recently, a new approach called TD_γ was introduced (Konidaris, Niekum, and Thomas 2011). The authors revisited the assumptions on the variances made by TD(λ) methods and provided an alternative complex backup weighting based upon the discount factor γ:

$$R_t^{TD_\gamma} = \sum_{n=1}^{|T|} \frac{(\sum_{i=1}^{n} \gamma^{2(i-1)})^{-1}}{\sum_{m=1}^{|T|}(\sum_{i=1}^{m} \gamma^{2(i-1)})^{-1}} R_t^{(n)} \ . \qquad (4)$$

The authors reported favorable empirical performance of TD_γ when compared to TD(λ) on several RL benchmark domains in on-policy learning settings. Since the TD_γ return includes all n-step returns, its expected value is very likely to be negative given the previous analysis, making it a suitable choice as the bounding estimator.

However, the bias in the TD_γ return becomes more negative as n increases. Too much bias will make the improvement negligible. The remedy is to only consider enough of the returns in order to achieve reduced variance and negative bias, but not too small of a bias. TD_γ can accomplish this by limiting the length of the trajectory it considers through a tuning parameter as was shown in (Konidaris, Niekum, and Thomas 2011). Our focus in this paper is to introduce the novel bounding method and demonstrate its effectiveness based on an existing formula of complex returns. How to design an optimal weighting scheme is an interesting research direction.

Bounded Fitted Q-Iteration

In the preceding section we described how a complex return can effectively be utilized as a lower bound for the $R^{(1)}$ estimator. We now propose a new algorithm called Bounded Fitted Q-Iteration (BFQI) that makes use of this approach in an AVI framework to provide improved value estimates. The details for BFQI are provided by Algorithm 1. We leave exact selection of the bounding complex return as an open choice in this general algorithm framework. In this paper we consider the TD_γ return.

Algorithm 1 BFQI($\mathcal{T}, \gamma, M, R^B$)

Input: $\mathcal{T}$: set of trajectories, γ: discount factor,
M: number of iterations, R^B: Bounding Return

1: $\hat{Q}_0 \leftarrow 0$
2: **for** $m = 1$ to M **do**
3: Let X and Y be empty sets.
4: **for** $k = 1$ to $|\mathcal{T}|$ **do**
5: **for all** $(s_t, a_t, s_{t+1}, r_{t+1}) \in T_k$ **do**
6: $X \leftarrow Append(X, (s_t, a_t))$
7: $Y \leftarrow Append(Y, \max(R_t^{(1)}, R_t^B))$
8: **end for**
9: **end for**
10: $\hat{Q}_m \leftarrow Regression(X, Y)$
11: **end for**
12: **Return** $\hat{Q}_M$

BFQI is similar to the popular FQI algorithm (Ernst et al. 2005) except it uses our bounded return method in updating the sample value estimates (line 7) as opposed to just using $R_t^{(1)}$. Like FQI, BFQI is off-policy and sample efficient. Additionally, given an efficient implementation it has the same computational complexity as the FQI algorithm. Just as in (Wright et al. 2013), the samples in the provided trajectories can be processed in reverse order. This allows return calculations for samples later in a trajectory to be reused by samples earlier in the trajectory.

Alternative Methods

Trajectory Fitted Q-Iteration (TFQI) (Wright et al. 2013) is a recently introduced AVI based algorithm that also makes use of the n-step returns. Instead of using a complex return, TFQI uses the n-step return that has the highest observed value as the sample Q-value estimate.

$$R^{Max} = \max(R^{(1)}, R^{(2)}, \ldots, R^{(|T|)}) \qquad (5)$$

Although there was no explicit mentioning of the bounding idea, it essentially uses the R^{Max} return as the bound for the $R^{(1)}$ return. The authors of this method reported significantly improved performance compared to that of FQI on

two RL benchmark problems. Despite the good reported results, our analysis in the previous subsections suggests that if any of the $R^{(n)}$ returns exhibits positive bias and/or high variance, the R^{Max} return will be easily overestimating the sample labels. Besides the difference in the proposed methods, the TFQI work did not provide any statistical explanation of why and when the R^{Max} return succeeds or fails, and the experimental evaluation was conducted solely in deterministic environments.

Experimental Results

We provide an empirical comparison of our bounding method using the TD_γ return as a bound, R^{B_γ} (as used in BFQI), with the standard $R^{(1)}$ method (as used in FQI), and the newer R^{Max} method (as used in TFQI). All methods are embedded within the same FQI framework and tested with the same initial assignment $\hat{Q}_0$ (note: varying the values of $\hat{Q}_0$ does not change the behaviors of these methods).

In all our experiments we use linear regression models with Fourier Basis functions (Konidaris, Osentoski, and Thomas 2011) trained using ridge regression. We examined a comprehensive set of parameters varying the complexity of the model, regularization, number of iterations, and trajectory counts. The results we report are representative of the general trends we observed.

Value Function Approximation Accuracy

Our first set of experiments evaluates the accuracy of each method in deriving Q^*. We use a non-deterministic 51-state Markov chain similar to the one presented in (Lagoudakis and Parr 2003) as our testing environment. The goal in this domain is to traverse the chain, stating from some random state, to one of the terminal states in as few steps as possible. States 0, 25, and 50 are terminal. From any non-terminal state the agent can take an action to move to one of the two neighboring states with a cost of -1. We set the discount factor, γ, to 0.9 and there is a 20% probability that an action taken will result in no transition. The function approximation model uses a 10th order Fourier basis with no regularization in training the model.

In order to evaluate the methods under varying levels of off-policy bias we generated multiple repositories of 10,000 trajectories based on behavior policies that follow the optimal policy with probability 0.9 to 0.5 (equivalent to a random policy) at each step. Given a repository and a method, for each run, 100 trajectories are randomly selected from the repository to form a training set. The results are the average of 200 runs. We evaluate each method based on the average MSE of the $\hat{Q}$ functions after 50 iterations of learning.

For completeness, we also consider the LSTD-Q algorithm (Lagoudakis and Parr 2003) an alternative batch-mode algorithm. In our testing LSTD-Q performed nearly identically to $R^{(1)}$, so we do not include that result. This finding is expected given that both LSTD-Q and FQI use $R^{(1)}$ and optimize the same objective function. We also tested using the full R^γ return, and found it performs poorly due to the off-policy bias, as expected. So, we exclude it from our result too. However, using the full R^γ return as a bound in our method performs very well as shown in Figure 2.

Figure 2 shows that our bounding method R^{B_γ} is able to learn Q^* as well as or more accurately than $R^{(1)}$. The gain is increasingly significant as the off-policy bias increases. This demonstrates our method providing an effective bound that reduces overall error across a large spectrum of trajectory quality. The R^{Max} method, in comparison, provides the greatest overall improvement over $R^{(1)}$ when the off-policy bias is highest. However, it is unstable and performs significantly worse than our method and the baseline $R^{(1)}$ method when there is less off-policy bias, demonstrating how this method is prone to overestimating.

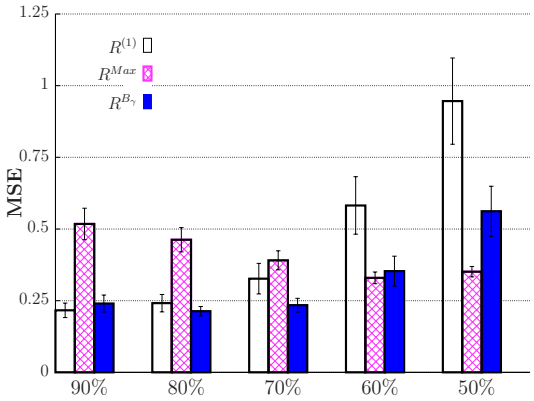

Figure 2: Average MSE of the learned $\hat{Q}$ functions. The behavior policy is varied from 90% to 50% of the optimal policy. Error bars show standard deviation.

While analyzing the results we performed further investigation in this domain, and observed that the distributions of the n-step returns, after convergence, closely resemble normal distributions and exhibit the general trend of increasingly negative bias we predicted. Due to space limitations, we omit graphs showing this trend.

Policy Performance

Here we compare methods based on policy performance using two additional RL benchmarks: the Acrobot (Acro) swing-up and the Cart Pole Balancing (CPB) (Sutton and Barto 1998). These two problems represent two different classes of domain: goal oriented and failure avoidance respectively. In the Acro domain the objective is to derive a policy that enables an under-actuated robot to swing-up in as few steps as possible, limited to 1,000. A cost of -1 is given for every non-terminal transition. Whereas, in the CPB domain the goal is to avoid the failure conditions, for up to 10,000 steps, of dropping the pole or exceeding the bounds of the track. Here a positive reward of $+1$ is given for every non-terminal transition. We set the discount factor for both domains $\gamma = 0.9999$. Like the Markov chain these domains were made non-deterministic by incorporating a 20% probability that an action results in no action taken. Fourier basis of orders 2, for Acro, and 3, for CPB, both trained with the same small regularization penalty, are used to represent $\hat{Q}$.

Policy performance is measured by the mean aggregate reward obtained by running a given policy over 50 trials. We used NEAT (Stanley and Miikkulainen 2002) to generate diverse trajectory sets, comprised of over 5,000 trajectories, for both domains as was done in the TFQI work (Wright et al. 2013). This form of data violates LSTD-Q's assumptions on sampling distribution, so we do not include it in these experiments. The reported results are an average of 200 runs for each setting.

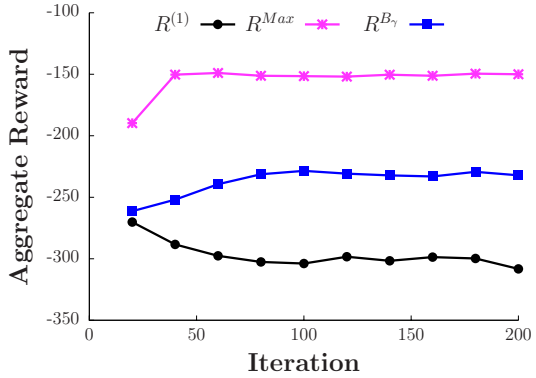

Figure 3: Mean policy performance at every 20 iterations (from 20-200) in the Acrobot domain using 100 trajectories.

As shown in Figure 3, in the Acro domain both R^{B_γ} and R^{Max} methods outperform $R^{(1)}$ with R^{Max} performing the best. Error bars are not shown, but a paired t-test ($p < 0.005$) was performed on the results at the last iteration to confirm the differences are statistically significant. In this domain the function approximation model appears to be a significant source of negative bias. This factor combined with off-policy bias prevents R^{Max} from overestimating, resulting in a much improved policy.

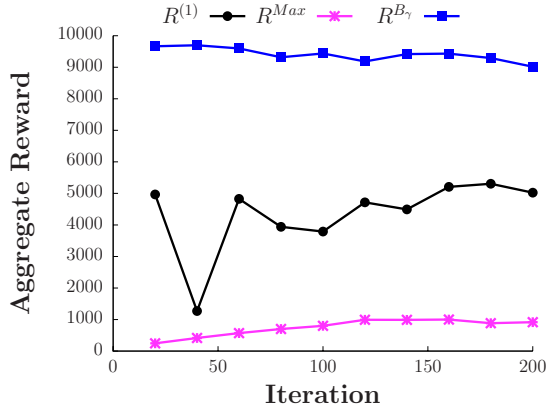

Figure 4: Mean policy performance at every 20 iterations (from 20-200) in the Cart Pole Balancing domain using 100 trajectories.

In contrast the results from the CPB domain, Figure 4 tells a much different story. The R^{Max} method performs the worst of all methods, demonstrating how R^{Max} can be too aggressive and unstable. R^{B_γ}, however, provides an effective bound that significantly improves policy performance over both competing methods. Together, these results clearly show how our bounding method can effectively and safely improve value estimation resulting in substantial policy performance gains.

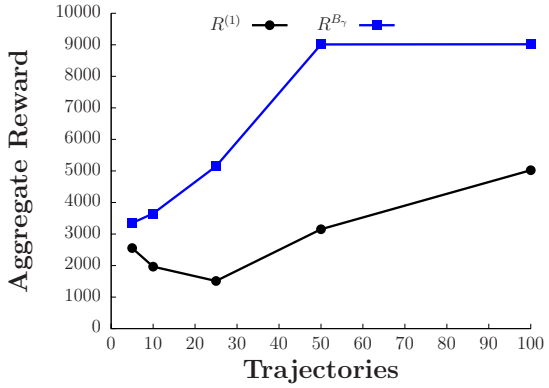

Figure 5: Final policy performance in the Cart Pole Balancing domain using trajectory sets of increasing size.

In addition, we also consider the performance of our method when provided with fewer trajectories. Sample efficiency is of critical importance, and demonstrating an improvement under low sample counts is valuable. Figure 5 shows the policy performance of our bounding method compared with $R^{(1)}$ in the CPB domain. R^{Max} is not included because it failed to demonstrate improved performance under any condition. The provided number of trajectories varied from 5 to 100. Across various trajectory set sizes we see very significant improvements in policy performance. Furthermore, with about 25 trajectories, our bounding method can achieve similar performance as the baseline method with 100 trajectories, which indicates superior sample efficiency of the bounding method.

Conclusion and Future Work

We have introduced a novel bounding approach that opens the door to exploiting complex returns for improving AVI. Our statistical analysis explained why and how this bounding approach works. We provided a new algorithm, BFQI, and empirically demonstrated how our method significantly improves value estimation for AVI, leading to improved policy performance and sample efficiency.

In future work we plan to perform a more formal analysis of the bounding method to discover better means of producing the bounding estimator. Since FQI is known to converge under certain assumptions, we plan to examine the theoretical convergence properties of BFQI. It is also interesting to study how other AVI methods can benefit from the idea of using complex returns by bounding.

Acknowledgements

This work is supported in part by grants from NSF (# 0855204) and the AFRL Information Directorate's Visiting Faculty Research Program. Qiao's contribution was partially supported by a Collaboration Grant from the Simons Foundation (# 246649).

References

Antos, A.; Szepesvári, C.; and Munos, R. 2007. Fitted q-iteration in continuous action-space mdps. In Platt, J.; Koller, D.; Singer, Y.; and Roweis, S., eds., *Advances in Neural Information Processing Systems 20*. 9–16.

Ernst, D.; Geurts, P.; Wehenkel, L.; and Littman, L. 2005. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research* 6:503–556.

Geist, M., and Scherrer, B. 2014. Off-policy learning with eligibility traces: A survey. *J. Mach. Learn. Res.* 15(1):289–333.

Gordon, G. J. 1995. Stable function approximation in dynamic programming. Technical report, DTIC Document.

Hachiya, H.; Akiyama, T.; Sugiayma, M.; and Peters, J. 2009. Adaptive importance sampling for value function approximation in off-policy reinforcement learning. *Neural Networks* 22(10):1399–1410.

Konidaris, G.; Niekum, S.; and Thomas, P. S. 2011. Td_γ: Re-evaluating complex backups in temporal difference learning. In *Advances in Neural Information Processing Systems*, 2402–2410.

Konidaris, G.; Osentoski, S.; and Thomas, P. 2011. Value function approximation in reinforcement learning using the Fourier basis. In *Proceedings of the Twenty-Fifth Conference on Artificial Intelligence*, 380–385.

Lagoudakis, M. G., and Parr, R. 2003. Least-squares policy iteration. *Journal of Machine Learning Research* 4:2003.

Munos, R. 2005. Error bounds for approximate value iteration. In *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*, AAAI'05, 1006–1011. AAAI Press.

Precup, D.; Sutton, R. S.; and Dasgupta, S. 2001. Off-policy temporal-difference learning with function approximation. In *ICML*, 417–424.

Puterman, M. L. 2009. *Markov decision processes: discrete stochastic dynamic programming*, volume 414. Wiley-Interscience.

Riedmiller, M. 2005. Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In *Machine Learning: ECML 2005*. Springer. 317–328.

Stanley, K. O., and Miikkulainen, R. 2002. Evolving neural networks through augmenting topologies. *Evolutionary Computation* 10(2):99–127.

Sutton, R. S., and Barto, A. G. 1998. *Introduction to reinforcement learning*. MIT Press.

Wright, R.; Loscalzo, S.; Dexter, P.; and Yu, L. 2013. Exploiting multi-step sample trajectories for approximate value iteration. In *Machine Learning and Knowledge Discovery in Databases*, volume 8188. Springer Berlin Heidelberg. 113–128.