CFQI: Fitted Q-Iteration with Complex Returns

Robert Wright[*], Xingye Qiao, Lei Yu[†]
Binghamton University
Binghamton, NY 13902, USA
{rwright3,xqiao,lyu}@binghamton.edu

Steven Loscalzo
AFRL Information Directorate
26 Electronic Pkwy
Rome, NY 13441, USA
Steven.Loscalzo@us.af.mil

ABSTRACT

Fitted Q-Iteration (FQI) is a popular approximate value iteration (AVI) approach that makes effective use of off-policy data. FQI uses a 1-step return value update which does not exploit the sequential nature of trajectory data. Complex returns (weighted averages of the n-step returns) use trajectory data more effectively, but have not been used in an AVI context because of off-policy bias. In this paper we propose a new generalization of FQI called Complex Fitted Q-Iteration (CFQI) which allows for complex returns. Theoretical properties are proved that show CFQI does not break existing convergence properties. Two methods for integrating complex returns are presented. The first method uses a simple truncating procedure for reducing off-policy bias. Our second method applies a novel bounding operation that utilizes the off-policy bias. We provide an empirical evaluation of the proposed methods on several reinforcement learning benchmarks. The results demonstrate that our methods significantly improve over FQI in terms of value estimation accuracy, policy performance, and convergence speed.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms

Keywords

Reinforcement Learning; Approximate Value Iteration; Off-Policy; Fitted Q-Iteration; Complex Returns

1. INTRODUCTION

Fitted Q-Iteration (FQI) is a widely used approximate value iteration (AVI) framework for solving reinforcement learning (RL) problems [4]. Since its introduction FQI has been used and extended by numerous works [1, 8, 13, 17]. FQI's most compelling feature is its ability to learn effectively from varied sources of off-policy sample data. Off-policy data are generated from behavior policies that differ from the target policy, the policy being learned. Whereas, with on-policy data the behavior policy and target policy are the same. In a multi-agent context, this ability is analogous to learning from the aggregate experiences of heterogeneous agents solving a given problem. Most other RL methods make restrictive assumptions on sample data rendering such collections of data useless. In many realistic learning situations, where simulation is impractical and obtaining samples is difficult and costly, it is critically important to be able to use all available data. FQI is a means to use all available data to learn an approximation of the optimal policy.

Although FQI can use off-policy sample data effectively, it does not exploit this data to the fullest extent. The key operation of FQI, and its derivatives, is its Q-value update function which makes use of the greedy 1-step return [19]. This 1-step update treats each sample as an independent event and relies completely on bootstrapped value estimates. These bootstrapped estimates can have significant error due to the use of function approximation and irregularly distributed sample sets. Samples, however, are not typically gathered as single-step experiences and they are not independent of each other. Instead they are gathered as multi-step experiences known as trajectories and share sequential relationships that can be used to reduce this error.

Trajectory data has been used to great effect in on-policy and policy iteration RL contexts through the use of *complex returns*[5, 7, 9, 19]. Complex returns are weighted averages of the n-step returns, value estimates made by looking further down a trajectory. Through careful design, the aggregated value estimates produced by complex returns have low variance and are generally more accurate than 1-step return estimates. Despite this advantage and the general availability of trajectory data, complex returns have not been considered in an AVI context to the best of the author's knowledge. There are two primary reasons. First, off-policy trajectories introduce bias into estimates. Second, the target policy is unknown, making it impossible to apply importance sampling to mitigate off-policy biases [15].

To meet these challenges we propose Complex Fitted Q-Iteration (CFQI), a generalization of the FQI framework which allows for any general return based estimate, enabling the seamless integration of complex returns and AVI. We introduce and analyze two distinct methods for utilizing complex returns within the CFQI framework. The first method

[*] Dr. Wright is currently affiliated with AFRL. Email Robert.Wright.25@us.af.mil

[†] Dr. Yu is the last author who directed Wright's Ph.D. research including this work.

Appears in: *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2015), Bordini, Elkind, Weiss, Yolum (eds.), May 4–8, 2015, Istanbul, Turkey.*

is similar to the idea of $Q(\lambda)$ [19] and uses truncated portions of trajectories, that are consistent with the approximation of Q^*, to calculate complex return estimates without introducing off-policy bias. The second method is a more nuanced approach that makes use of the inherent negative bias of complex returns, due to the value iteration context, as a lower bound for value estimates. We provide statistical evidence and analysis that shows how an estimator with predictable, but unknown, bias can provide a bound on value estimates to produce a more accurate estimator. Additionally, we include convergence proofs showing that CFQI is guaranteed to converge under the same assumptions as FQI. Finally, we provide an empirical evaluation of our methods on several RL benchmarks that show how CFQI improves the accuracy of the learned Q^* approximation, the quality of the learned policy, and convergence behavior.

2. BACKGROUND

Our focus in this work is finding solutions to Markov Decision Processes (MDP) [16]. 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.

Solutions to MDPs are policies, which are functions on the state space that determine which action to take, $\pi : S \mapsto A$. The quality of a policy is determined by the expected value that can be obtained by following it from any given state. In RL we are concerned with Q-values which, given π, are defined as: $Q_\pi(s, a) = \mathbb{E}_\pi[\sum_{i=0}^{\infty} \gamma^i r_{i+1}|s_0 = s, a_0 = a]$, where r_{i+1} is the immediate reward given at time $i + 1$.

Our goal is to derive an optimal policy, π^*, that maximizes this value for all $s \in S$. For this purpose we can estimate the optimal Q-function, Q^*, which is defined as the solution to the optimal Bellman equation: $Q^*(s, a) = \sum_{s' \in S} P(s'|s, a)[\mathcal{R}^a_{s,s'} + \gamma \max_{a' \in A} Q^*(s', a')]$. From this equation π^* can be extracted as: $\pi^*(s) = \arg\max_{a \in A} Q^*(s, a)$.

If P and $\mathcal{R}$ are known, Q^* can be solved for efficiently using dynamic programming. However, in RL scenarios P and $\mathcal{R}$ are unknown and Q^* must be learned from samples. Samples are single-step observations of transitions 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. There are many different RL approaches for solving for Q^*. Here we focus on Fitted Q-Iteration.

2.1 Fitted Q-Iteration

Fitted Q-Iteration (FQI) [4] is a batch-mode, off-line, off-policy approach for solving RL problems. It is an approximate value iteration [12] framework that solves directly for Q^* through a sequence of standard supervised learning regression problems. As a batch-mode algorithm it makes efficient use of samples. It has also been proven that, under restrictive assumptions of the regression model, FQI is guaranteed to converge towards Q^* [14].

FQI starts with an arbitrarily initialized approximation of Q^*, $\hat{Q}_0$. This approximation of Q^* is then refined through an iterative process. In this process the estimated Q-values of each sample are calculated using the current $\hat{Q}$ approximation. These values are then used as target values for a regression algorithm that "fits" them with their corresponding sample state and action features producing the next approximation, $\hat{Q}_m$. The process is repeated for M iterations or until some other stopping criteria.

A crucial component of this process is how the sample value estimates are calculated. The accuracy of these estimates dictates the final accuracy of $\hat{Q}_M$ and in turn the quality of the derived policy. For this purpose FQI uses the greedy 1-step return estimate, $R^{(1)}_t$:

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

which combines the single-step observed immediate reward with a greedy choice among all *bootstrapped* estimates of future returns provided by $\hat{Q}_{m-1}$. $R^{(1)}_t$ is a reasonable choice for a value estimator as it is unbiased with regard to the sampling policies. However, it is not the only choice and it is very sensitive to error caused by biases and variances in an imperfect function approximation model and irregular sample distributions. In the following we show how complex returns are more robust and can be used to provide more accurate estimates in an AVI context.

2.2 Complex Returns

Sample data are most commonly collected in sequences known as trajectories. A trajectory, T, is a sequentially ordered collection of observations where, $T = [(s_0, a_0, s_1, r_1), (s_1, a_1, s_2, r_2), \dots]$. Trajectories provide an alternative to using just the $R^{(1)}_t$ return. Given trajectory data, the 1-step return estimate in Eq(1) can be generalized to produce the n-step returns:

$$R^{(n)}_t = \sum_{i=1}^{n-1} \gamma^{i-1} r_{t+i} + \gamma^n \max_{a \in A} \hat{Q}_{m-1}(s_{t+n}, a) \ . \qquad (2)$$

It should be noted that this definition of the n-step returns differs from the standard on-policy definition of the n-step returns because of its use of the *max* operation. In principle each of the n-step returns can be used as approximations of $Q^*(s_t, a_t)$. Individually each estimator has its own distinct bias and variance. However, when combined, through averaging they can produce an estimator with lower variance than any one individual return [3]. It is this idea that motivated the development of *complex* returns [19, Chapter 7].

A complex return is a weighted average, with the weights summing to 1, of the n-step returns. The n-returns are weighted differently because of their assumed relative variance behaviors. The general assumption behind existing complex return methods is that the variance of the n-step returns increases as n increases. From the on-policy literature there are two competing complex return approaches. The classic complex return is the λ-return which serves as the basis for the TD(λ) family of algorithms [19]. More recently, the γ-return was introduced based upon different variance assumptions of the n-step returns [9]. The γ-return is defined as:

$$R^\gamma_t = \sum_{n=1}^{|T|} \frac{(\sum_{i=1}^{n} \gamma^{2(i-1)})^{-1}}{\sum_{m=1}^{|T|} (\sum_{i=1}^{m} \gamma^{2(i-1)})^{-1}} R^{(n)}_t \ . \qquad (3)$$

The difficulty in applying complex returns to FQI is that in an AVI context the trajectories can be sampled off-policy and the target policy is also unknown. Off-policy trajectories introduce undesirable bias into the n-step return estimates that cannot be reduced through averaging. If the target policy were known, as in policy iteration, importance sampling can be used to reduce off-policy bias [15]. However, the target policy is unknown in our AVI context. In the following section we will introduce two methods that utilize complex returns effectively in an AVI context.

3. COMPLEX FITTED Q-ITERATION

Here we introduce Complex Fitted Q-Iteration (CFQI) our generalization of the popular FQI framework that enables the use of complex return based value estimates. Algorithm 1 provides the details of the approach.

Algorithm 1 CFQI($\mathcal{T}, M, R^C$)

```
Input: $\mathcal{T}$: set of trajectories,
    $M$: number of iterations, $R^c$: complex return function
 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 $t = 1$ to $|T_k|$ do
 6:       $X \leftarrow Append(X, (s_t^{T_k}, a_t^{T_k}))$
 7:       $Y \leftarrow Append(Y, R^C(t, T_k, \hat{Q}_{m-1}))$
 8:     end for
 9:   end for
10:   $\hat{Q}_m \leftarrow Regression(X, Y)$
11: end for
12: Return $\hat{Q}_M$
```

The primary distinction between FQI and CFQI lies in the two update rules (line 7). FQI is limited to the $R^{(1)}$ return estimate, while CFQI makes use of any chosen complex return R^C to provide value estimates. A second difference between FQI and CFQI is that CFQI processes over trajectories, not unordered samples as in FQI. CFQI has the same computational complexity as FQI, since the derivations of the n-returns and complex returns can be performed efficiently by processing trajectories in reverse order. One of our contributions is to show that using complex returns as value estimates does not break the theoretical convergence guarantees of the original approach.

THEOREM 1. *CFQI converges w.p.1 if a normalized complex return, computed from fixed length trajectories, is used to derive value targets to be used in a kernel regression model, as defined by Eq(7) and Eq(8).*

The proof for Theorem 1 and equations Eq(7) and Eq(8) are provided in the Appendix. Still, although CFQI can be guaranteed to converge, the bias of off-policy n-step returns will introduce bias into the final result possibly eliminating any potential benefit from variance reduction. In the following subsections we propose two methods which aim to either mitigate or utilize this bias.

3.1 Method 1: Truncated Complex Returns

One way to handle the off-policy bias of the complex returns is to attempt to avoid it by truncating the trajectories where they appear to go off-policy. This idea is borrowed from the $Q(\lambda)$ [21] approach. However, this is the first work to the author's knowledge that has considered it for AVI. In this approach the current $\hat{Q}$ provides an approximation of the optimal policy that can be used to infer when a trajectory takes an off-policy sub-optimal action. During the process of calculating the complex return estimates, samples in a trajectory after the first off-policy action are not considered. Assuming $\hat{Q}$ is a close approximation of Q^* this approach should not introduce off-policy bias and can take advantage of portions of trajectories that follow the optimal policy to reduce variance and overall error. We refer to this method as CFQI-C.

However, the assumption that $\hat{Q}$ is an accurate approximation of Q^* is a poor one, especially early in the iterative process. Additionally, because the learned policy will likely change during the iterative process, the lengths of the trajectories used to calculate the complex returns will change dynamically from one iteration to the next. Changing lengths of the trajectories violates one of the assumptions made by Theorem 1 and convergence may no longer be guaranteed. This issue is examined further in the empirical study.

3.2 Method 2: Complex Returns as Bounds

Our second approach uses the off-policy n-step return bias rather than attempting to eliminate it. It exploits the predictability of this bias enabling the effective use of complex returns as a bound on the value of the $R^{(1)}$ return. In an AVI context, such as CFQI, this is possible due to the fact that the target policy is an optimal policy. Because the target policy is optimal, it is a safe assumption that any off-policy bias present in the n-step returns is negative in value. A complex return derived from the biased n-step returns will also be biased negatively, but should have relatively low variance. This insight directly leads to the derivation of a bounded complex return, R^B:

$$R^B = \max(R^{(1)}, R^C). \tag{4}$$

where R^C is some chosen complex return function. When integrated with CFQI, we refer to this method as CFQI-B.

To help motivate this approach we first consider a simple scenario. Let us assume $R^{(1)}$ is an unbiased estimator with non-zero variance and R^C is a constant that is known to be less than the target value being estimated. In this case it is always better to use $\max(R^{(1)}, R^C)$ in place of $R^{(1)}$ to estimate the value. When $R^{(1)}$ is less than R^C it must be farther away from the true value than R^C, in which case the greater observation R^C should be used. Realistically, however, R^C is not a constant and we must further examine the conditions under which R^C can provide an effective bound.

We now investigate when R^B improves $R^{(1)}$ using some distributional assumptions. The occurrence and degree of improvement depends on the underlying distribution of $R^{(1)}$ and R^C. Here we assume both estimators follow normal distributions. Further, let us also assume that bias($R^{(1)}$) = 0 and $STD(R^{(1)}) = 1$. The bias and STD of R^C 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 R^B by Monte Carlo integration. Figure 1 provides a rendering of this model. The red solid curve is the boundary where R^B and $R^{(1)}$ are equally good. The area below the red curve is where R^B improves upon $R^{(1)}$.

The black contours in Figure 1 show the log ratio of MSE,

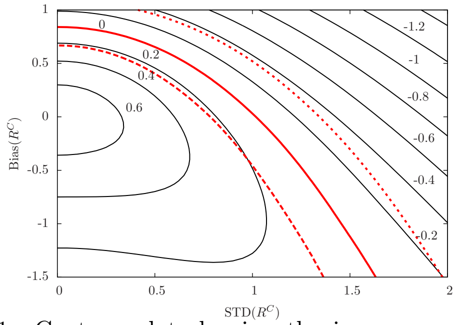

Figure 1: Contour plot showing the improvement of R^B over $R^{(1)}$. Red curves: boundaries between improvement and no improvement for cases of bias$(R^{(1)}) = 0$ (solid), 0.5 (dashed), and -0.5 (dotted).

$\log(MSE(R^{(1)})/MSE(\max(R^{(1)}, R^C)))$. The greater this value is, the more improvement $\max(R^{(1)}, R^C)$ has. Clearly the greatest improvement occurs at the unrealistic case where R^C is unbiased and has 0 variance. In general a combination of small bias and small variance is sufficient to guarantee an improvement. The more negative the bias is, the greater variance is allowed. However, if the bias is overly negative, then the improvement becomes negligible. Additionally, even if the bias of R^C is positive there is still a chance for the R^B to be a better estimator if $\text{Var}(R^C) < \text{Var}(R^{(1)})$.

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 $\max(R^{(1)}, R^C)$ to improve $R^{(1)}$, when $R^{(1)}$ is itself negatively biased; and vice versa. However, there is a potential risk of the bounding estimator R^C falls into the domain of loss (i.e., the northeast of the red curve). Such risk increases as bias$(R^{(1)})$ becomes more positive. Fortunately, $|\text{bias}(R^{(1)})|$ is generally under control in an AVI context. Ideally one would choose a model that fits well and thus does not have much bias without over-fitting.

We can now identify some characteristics about R^C that ensure an effective R^B:

1. The bias of R^C must be less than a positive value τ that satisfies $MSE(\max(R^{(1)}, \tau)) = MSE(R^{(1)})$. $\tau \approx 0.8399$ in the example given in Figure 1.

2. The variance of R^C should be small. It can be greater than that of $R^{(1)}$ as long as R^C has negative bias.

3. The bias of R^C should not be overly negative.

The first two criteria ensure a safe bound from below such that R^B is no worse than $R^{(1)}$. The third criterion is necessary to ensure there is a fair chance for $R^{(1)} < R^C$, enabling an effective bound.

The preceding analysis provides the motivation for our bounding approach. Given our AVI context the bias of off-policy n-step returns generally decreases as n increases. Hence, $\text{bias}(R^{(n)}) < \text{bias}(R^{(n-1)}) < \cdots < \text{bias}(R^{(1)})$. This bias behavior means if one was to use an n-step return or a complex return as the bounding estimator, it is very likely to satisfy the first criterion. Our choice in R^C will ultimately determine if the second and third criterion are satisfied.

In this paper we choose the γ-return as R^C for a proof of concept. Optimal design and selection of the bounding estimator is potentially an interesting future research direction. Given its proven variance reduction property [9], the γ-return is likely to have small variance, satisfying criterion 2. Since the γ-return includes non-negligible weights for all n-step returns its expected value is very likely to be negative, satisfying criterion 1. It is possible for this bias to become too negative, failing criterion 3. This can be remedied by restricting the length of the trajectory γ-return considers as was shown in [9].

Theorem 2 (proof in Appendix) below assures that CFQI with the bounding method also converges under the same conditions as FQI. This theorem can be further generalized to state that if CFQI converges with any two complex return estimates independently, then using one to bound the value of the other is also guaranteed to converge.

Theorem 2. *CFQI converges w.p.1 if the $R^{(1)}$ return is bounded by a normalized complex return on fixed length trajectories to produce value estimates used in a kernel regression model, as defined by Eq(7) and Eq(8).*

4. RELATED WORK

Trajectory Fitted Q-Iteration (TFQI) [22] is a recently introduced FQI 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. However, 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 likely overestimate values. In addition, the TFQI work did not provide any statistical explanation of why and when the R^{Max} return works or fails, and the experimental evaluation was conducted in deterministic environments.

There exists many approaches that enable the use of complex returns to improve value estimation in certain off-policy contexts through the use of importance sampling[5, 20]. These methods are similarly motivated but are limited to the policy iteration RL contexts where the target policy is known.

5. EMPIRICAL STUDY

In this section we provide an empirical evaluation of our approaches on several non-deterministic RL benchmarks. The methods are compared based upon accuracy of the learned value function, quality of the derived policy, and convergence behavior. We report comparative results for four methods, with the return used by each method listed in the following.

Method	**Value Estimator**
FQI	$R^{(1)}$
TFQI	R^{Max}
CFQI-C$_\gamma$	Truncated R^γ
CFQI-B$_\gamma(l)$	$R^B = max(R^{(1)}, R^{(\gamma)})$

For the CFQI-B$_\gamma(l)$ method, l denotes the limit on how many steps down the trajectory to use when computing the R^γ return. If l is not listed, it uses the full trajectory.

In all our experiments we use ridge linear regression with Fourier Basis functions [10]. AVI is known to exhibit divergence behavior when paired with this type of function approximation model [2]. We are able to circumvent this issue by bounding the values returned by the models with V_{max}. V_{max} is the maximum possible value for any state-action pair and can be calculated a priori as:

$$V_{max} = \frac{R_{max}}{(1-\gamma)} \tag{6}$$

where R_{max} is the maximum single step reward in the domain. This change was sufficient to ensure convergence for most methods we tested. In our experiments this form of function approximation provided results superior to the kernel averaging based methods that AVI is guaranteed to converge with. If the approximated values were not bounded by V_{max} we did observe divergence behavior from all methods. In our experimentation 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 consistent with the general trends we observed. Statistical significance is determined by performing a paired t-test. We report a result as significant in the following analysis if ($p < 0.005$).

5.1 Value Function Approximation Accuracy

In this series of experiments we examine how accurately the methods can derive Q^* using identical trajectory data sets. For this purpose we use a non-deterministic 51-state Markov chain similar to the one presented in [11]. This environment is chosen because Q^* can be calculated exactly using dynamic programming. The goal is to traverse the chain, starting from some random state, to one of the terminal states in as few steps as possible. There are three terminal states: 0, 25, and 50. 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 order to evaluate the methods under varying levels of off-policy bias we generated multiple repositories of 10000 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. For each run 1000 trajectories are randomly selected from a chosen repository to form a training data set. The reported results are the average of 200 runs. We evaluate each method based on the average MSE of the $\hat{Q}$ functions, comparing to the true Q^* function, after 50 iterations of learning (sufficient to ensure convergence).

For completeness, we also consider LSTD-Q [11], an alternative batch-mode algorithm. LSTD-Q's performance was nearly identical to FQI's, 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.

The results reported in Figure 2 show the CFQI based methods are stable and outperform FQI at most levels of off-policy bias. The only exception is with data from a 90% optimal policy, where CFQI performs comparably to FQI. TFQI on the other hand shows unstable results. It performs poorly when there is less off-policy bias, demonstrating that the R^{Max} return can be prone to overestimate. However, it is significantly better than all other methods on near random trajectory data. Comparing CFQI-C and CFQI-B, we see that the bounding approach can perform significantly better than its truncated complex return counterpart.

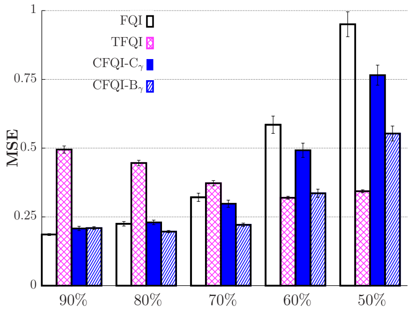

Figure 2: Average MSE of the $\hat{Q}$ functions for the various methods. The behavior policy is varied from 90% to 50% of the optimal policy. Error bars show standard deviation.

5.2 Policy Performance

In this set of experiments we examine the impact of improved value estimation on the policy quality on two RL benchmarks: the Acrobot (Acro) swing-up and the Cart Pole Balancing (CPB) [19]. These two particular problems were chosen because they represent 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 1000. 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 $\gamma = 0.9999$ for Acro and $\gamma = 0.99$ for CPB. 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 and trained with small regularization penalties are used to represent $\hat{Q}$ in both domains.

Policy performance is measured by the mean aggregate reward obtained by running a given policy over 50 trials, necessary due to the non-determinism. Experiments are run on data sets comprised of increasing numbers of trajectories to examine the relative sample efficiency of the methods. NEAT [18] is used to generate diverse trajectory sets, comprised of over 5000 trajectories, for both domains as was done in the TFQI work [22]. 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 after 300 iterations of learning. Error bars are not included in the results, but statistically significant results are reported.

Figure 3 shows the policy performance results in the Acro domain. TFQI performs the best, significantly outperforming all other methods except at 100 trajectories. This observation suggests that there is significant negative bias stemming from either the trajectories, model, or both, making R^{Max} a safe estimator. The results for CFQI-C$_\gamma$ are purposely missing from Figure 3. CFQI-C$_\gamma$ has difficulty con-

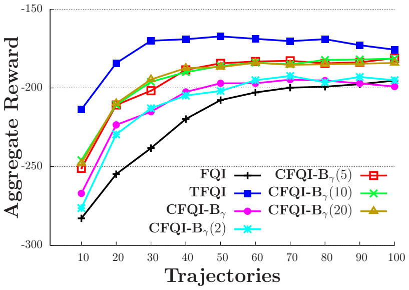

Figure 3: Policy performance after 300 iterations in the Acrobot domain using trajectory sets of increasing size.

verging on all data sets in this domain, confirming our suspicion from Section 3.1. The resulting policies all averaged an aggregate reward value of -700 or less.

CFQI-B$_\gamma$ performs well, but only significantly outperforms FQI in the 20 through 50 trajectory range. This demonstrates how the full γ-return can fail our third criterion by incorporating too much off-policy bias. In Section 3.2 we proposed a solution that limits the length of complex return. Figure 3 also shows results for CFQI-B$_\gamma(l)$ for various l settings. Setting $l = 2$ performs comparably to the default full length setting, CFQI-B$_\gamma$, which are representatives of the two extremes of the parameter's range. It is worth noting at $l = 1$ CFQI-B$_\gamma(l)$ reduces to FQI. From $l = 5$ to 20, CFQI-B$_\gamma(l)$ demonstrates significantly better performance than FQI at all trajectory counts. In terms of sample efficiency the improvement is dramatic. With just 30 trajectories CFQI-B$_\gamma(10)$ achieves the same level of performance as FQI with 100 trajectories. These results show CFQI-B$_\gamma(l)$ provides effective bounds that enable significantly better policies to be learned on less data.

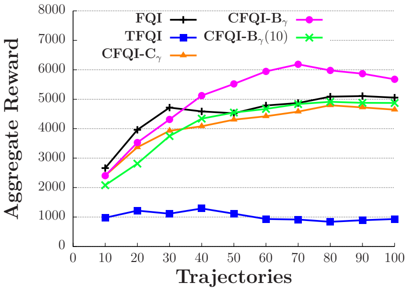

Figure 4: Policy performance after 300 iterations in the Cart Pole domain using trajectory sets of increasing size.

In Figure 4 we present the results from experiments on the CPB domain. In sharp contrast to the Acro results, TFQI performs the worst in this domain. It fails to find a competent policy at all trajectory counts, confirming that it can be an overly aggressive bound and an unstable approach. All other methods perform comparably with FQI with the exception of CFQI-B$_\gamma$. At higher trajectory counts CFQI-B$_\gamma$ learns a significantly better policy than all other methods. This observation can be explained by the γ-return's long-tail weighting and the specifics of the CPB domain. In the CPB domain all rewards are positive with the exception of transitions to failure states. As a result, it is hard to accumulate negative bias needed for the bounding estimator along a short trajectory segment.

5.3 Convergence Behavior

Convergence behavior is an important consideration because it determines how long before a consistent policy can be extracted or if the approach will succeed at all. Here we examine that behavior based on policy convergence and convergence of the approximated Q functions, $\hat{Q}$.

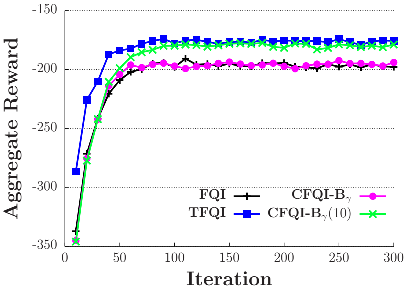

Figure 5: Mean policy performance at every 10 iterations in the Acrobot domain using 100 trajectories.

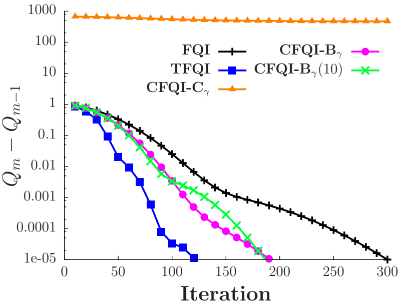

Figure 6: Mean per-iteration $\hat{Q}$ difference at every 10 iterations in Acrobot using 100 trajectories.

The results for the Acrobot domain are shown in Figures 5 and 6. Figure 5 shows the policy performance evaluated at every 10th iteration and Figure 6 shows the per-iteration difference in $\hat{Q}$ models. From Figure 5 it appears that the policy for the methods shown all converge around the 100th iteration. The explanation for this is that CFQI-C(γ) fails to converge as shown in Figure 6. The lack of convergence is caused by the non-fixed length of truncated trajectories. This finding suggests that the CFQI-C approach is not reliable. TFQI converges the fastest of all approaches followed by the CFQI-B methods, which all converge significantly faster than FQI.

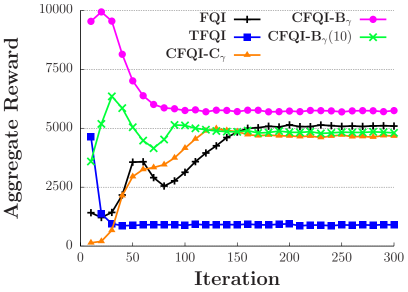

Figure 7: Mean policy performance at every 10 iterations in the Cart Pole Balancing domain using 100 trajectories.

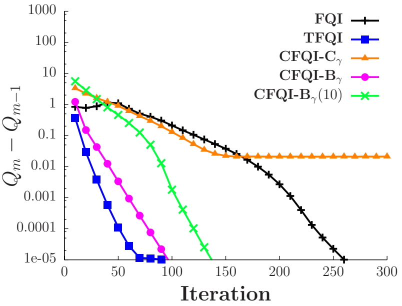

Figure 8: Mean per-iteration $\hat{Q}$ difference at every 10 iterations in the Cart Pole domain using 100 trajectories.

Figures 7 and 8 show the similar results for the CPB domain. FQI, CFQI-C$_\gamma$, and CFQI-B$_\gamma$(10) all converge to policies with similar performance after 100 iterations. It is somewhat odd that the CFQI-B$_\gamma$ runs produce near optimal policies early in the iterative process before converging to a lesser performing policy. This result demonstrates how there can be a disconnect between deriving an accurate value function and actual policy performance. TFQI also converges quickly, but to a poor policy. Figure 8, again, shows that CFQI-C$_\gamma$ fails to converge, even though it does manage to derive a stable policy. Truncating the trajectories dynamically causes oscillations in the policy and value estimates that prevent convergence. CFQI-B$_\gamma$ meanwhile, converges towards the best performing policy significantly quicker than FQI.

6. CONCLUSIONS AND FUTURE WORK

In this paper we have explored, for the first time, the idea of utilizing complex returns for AVI. A new AVI framework, CFQI, and two new approaches based on this framework, CFQI-C and CFQI-B, have been proposed. We have shown through statistical evidence how the sub-optimal off-policy bias, unique to the AVI context, can be exploited by using the complex returns as bounds on value estimates. We have provided proof that CFQI converges with fixed length complex returns and when bounding is used. Finally, we have provided an empirical evaluation that clearly demonstrates our bounding approach improves the accuracy of value estimates for AVI, resulting in significantly better policies, faster convergence, and improved sample efficiency.

Specialized bounding complex returns are a potential future direction for this research. In this paper we examined the γ-return as bound. It was not designed with this purpose and there are likely more effective bounding complex returns to be discovered. In addition, our empirical results show different methods perform best depending on the learning scenario. An adaptive method that mixes various approaches based upon the data and domain could lead to even better overall performance.

Acknowledgements

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

REFERENCES

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

[2] J. A. Boyan and A. W. Moore. Generalization in reinforcement learning: Safely approximating the value function. In *Advances in Neural Information Processing Systems 7*, pages 369–376, 1995.

[3] T. G. Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[4] D. Ernst, P. Geurts, L. Wehenkel, and L. Littman. Tree-based batch mode reinforcement learning. *J. Mach. Learning Research*, 6:503–556, 2005.

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

[6] G. J. Gordon. Approximate solutions to markov decision processes. *Robotics Institute*, page 228, 1999.

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

[8] S. Kalyanakrishnan and P. Stone. Batch reinforcement learning in a complex domain. In *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, page 94. ACM, 2007.

[9] G. Konidaris, S. Niekum, and P. S. Thomas. Td$_\gamma$: Re-evaluating complex backups in temporal difference learning. In *Advances in Neural Information Processing Systems*, pages 2402–2410, 2011.

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

[11] M. G. Lagoudakis and R. Parr. Least-squares policy iteration. *J. Mach. Learn. Res.*, 4:2003, 2003.

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

[13] A. Nouri and M. L. Littman. Multi-resolution exploration in continuous spaces. In *Advances in Neural Information Processing Systems*, pages 1209–1216, 2008.

[14] D. Ormoneit and Ś. Sen. Kernel-based reinforcement learning. *Machine learning*, 49(2-3):161–178, 2002.

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

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

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

[18] K. O. Stanley and R. Miikkulainen. Evolving neural

networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002.

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

[20] C. Thiery, B. Scherrer, et al. Least-squares λ policy iteration: Bias-variance trade-off in control problems. In *Int. Conference on Machine Learning*, 2010.

[21] C. J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, UK, 1989.

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

APPENDIX

A. PROOFS FOR THEOREMS 1 AND 2

In previous works AVI, using $R^{(1)}$ as the value estimator, has been shown to converge as long as the regression model is an averager such as a normalized kernel method [6, 14, 4]. Specifically, the supervised learning method learns a model, $\hat{Q}(s,a)$, defined by:

$$\hat{Q}(s,a) = \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k\left((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)\right) R_{\ell,t}^{c} \tag{7}$$

where $R_{\ell,t}^{c}$ is the estimated value for sample t from T_ℓ. $R_{\ell,t}^{c}$ can be $R^{(1)}$, as in the standard AVI, or, as we will show, any normalized complex return. Additionally the kernel, $k : (S \times A)^2 \mapsto \mathbb{R}$, must satisfy the following condition:

$$\sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} \left| k\left((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)\right) \right| = 1, \forall (s,a) \tag{8}$$

A.1 Proof for Theorem 1

PROOF. Following the proof from [4], the sequence of M Q-functions can be rewritten as, $\hat{Q}_m = \hat{H}\hat{Q}_{m-1}$ where $\hat{H}$ is an operator mapping any function in a Banach space $\mathcal{H}$ of functions over $S \times A$ to $\mathcal{H}$ itself. $\hat{H}$ is defined as:

$$\begin{aligned}(\hat{H}K)(s,a) = \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k\left((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)\right) * \\ \sum_{n=1}^{|T_\ell - t|} w(n) \left[\sum_{i=1}^{n-1} \gamma^{i-1} r_{t+i} + \gamma^n \max_{a' \in A} K(s_{t+n}, a')\right]\end{aligned}$$

where $w(n)$ is the set of weights on the individual n-returns defined by the complex return. Next we show that $\hat{H}$ is a contraction in $\mathcal{H}$. Specifically we show $\|\hat{H}K - \hat{H}\bar{K}\|_\infty < \|K - \bar{K}\|_\infty$ for any K and $\bar{K} \in \mathcal{H}$

$$\begin{aligned}
&\|\hat{H}K - \hat{H}\bar{K}\|_\infty \\
= &\max_{(s,a)\in S\times A} \left| \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)) \sum_{n=1}^{|T_\ell - t|} w(n)\gamma^n \right. \\
&\left. \left[\max_{a' \in A} K(s_{t+n}, a') - \max_{a' \in A} \bar{K}(s_{t+n}, a')\right] \right| \\
\leq &\max_{(s,a)\in S\times A} \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)) * \\
&\sum_{n=1}^{|T_\ell - t|} w(n)\gamma^n \left|\max_{a' \in A} K(s_{t+n}, a') - \max_{a' \in A} \bar{K}(s_{t+n}, a')\right| \\
< &\gamma \max_{(s,a)\in S\times A} \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)) * \\
&\sum_{n=1}^{|T_\ell - t|} w(n) \max_{a' \in A} \left|K(s_{t+n}, a') - \bar{K}(s_{t+n}, a')\right| \\
\leq &\gamma \max_{(s,a)\in S\times A} \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)) * \\
&\max_{t' \geq t, (s_{t'}, a') \in S_{T_l} \times A} |K(s_{t'}, a') - \bar{K}(s_{t'}, a')| \\
\leq &\gamma \max_{(s,a)\in S\times A} |K(s,a) - \bar{K}(s,a)| \\
= &\gamma \|K - \bar{K}\|_\infty \\
< &\|K - \bar{K}\|_\infty
\end{aligned}$$

By fixed-point theorem, we have completed the proof. □

A.2 Proof for Theorem 2

PROOF. Following proof of Theorem 1, $\hat{H}$ is defined as:

$$\begin{aligned}
(\hat{H}K)(s,a) = \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k\left((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)\right) \\
\max\left\{r_{t+1} + \max_{a' \in A} K(s_{t+1}, a'),\right. \\
\left.\sum_{n=1}^{|T_\ell - t|} w(n)\left[\sum_{i=1}^{n-1} \gamma^{i-1} r_{t+i} + \gamma^n \max_{a' \in A} K(s_{t+n}, a')\right]\right\}
\end{aligned}$$

We now show $\hat{H}$ is a contraction in $\mathcal{H}$.

$$\begin{aligned}
&\|\hat{H}K - \hat{H}\bar{K}\|_\infty \\
= &\max_{(s,a)\in S\times A} \left| \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k\left((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)\right) \right. \\
&\Bigg(\max\left\{r_{t+1} + \gamma \max_{a' \in A} K(s_{t+1}, a'),\right. \\
&\left.\sum_{n=1}^{|T_\ell - t|} w(n)\left[\sum_{i=1}^{n-1} \gamma^{i-1} r_{t+i} + \gamma^n \max_{a' \in A} K(s_{t+n}, a')\right]\right\} - \\
&\max\left\{r_{t+1} + \gamma \max_{a' \in A} \bar{K}(s_{t+1}, a'),\right. \\
&\left.\left.\sum_{n=1}^{|T_\ell - t|} w(n)\left[\sum_{i=1}^{n-1} \gamma^{i-1} r_{t+i} + \gamma^n \max_{a' \in A} \bar{K}(s_{t+n}, a')\right]\right\}\Bigg) \right| \\
\leq &\max_{(s,a)\in S\times A} \left| \sum_{T_\ell \in \mathcal{T}} \sum_{t=1}^{|T_\ell|} k\left((s_t^{T_\ell}, a_t^{T_\ell}), (s,a)\right) \right. \\
&\max\left\{\gamma \max_{a' \in A} K(s_{t+1}, a') - \gamma \max_{a' \in A} \bar{K}(s_{t+1}, a'), \sum_{n=1}^{|T_\ell - t|} w(n) * \right. \\
&\left.\left.\left[\gamma^n \max_{a' \in A} K(s_{t+n}, a') - \gamma^n \max_{a' \in A} \bar{K}(s_{t+n}, a')\right]\right\} \right|
\end{aligned}$$

At this point all that remains is to show that both choices in the second $max\{\}$ function are less than $\|K - \bar{K}\|_\infty$ independently. The first choice was proven in [4]. Finally, the second choice is proven by Theorem 1. □