Adaptive Trade-Offs in Off-Policy Learning

Mark Rowland* DeepMind

Will Dabney* DeepMind

Rémi Munos DeepMind

Abstract

A great variety of off-policy learning algorithms exist in the literature, and new breakthroughs in this area continue to be made, improving theoretical understanding and yielding state-of-the-art reinforcement learning algorithms. In this paper, we take a unifying view of this space of algorithms, and consider their trade-offs of three fundamental quantities: *update variance*, *fixed-point bias*, and *contraction rate*. This leads to new perspectives on existing methods, and also naturally yields novel algorithms for off-policy evaluation and control. We develop one such algorithm, C-trace, demonstrating that it is able to more efficiently make these trade-offs than existing methods in use, and that it can be scaled to yield state-of-the-art performance in large-scale environments.

1 Introduction

Off-policy learning is crucial to modern reinforcement learning, allowing agents to learn from memorised data, demonstrations, and exploratory behaviour [Szepesvári, 2010, Sutton and Barto, 2018]. As such, it is a long-studied problem, with a variety of well-understood associated algorithms; see [Precup et al., 2000, Kakade and Langford, 2002, Dudík et al., 2014, Thomas and Brunskill, 2016, Munos et al., 2016, Mahmood et al., 2017, Farajtabar et al., 2018] for a representative selection of publications.

However, this paper is motivated by the observation that in spite of this theoretical progress, several state-of-the-art value-based reinforcement learning agents (notably Rainbow [Hessel et al., 2018] and R2D2 [Kapturowski et al., 2019]) eschew these off-policy algorithms, attaining better performance by using *uncorrected* returns, which do not account for the fact that data is generated off-policy. Further research has corroborated this observation [Hernandez-Garcia and Sutton, 2019]. This raises two central research questions: (i) How can we understand the strong performance of uncorrected returns? (ii) Can we distil these advantages, and combine them with existing off-policy algorithms to improve their performance?

One of the principal contributions of this paper is to show that the performance of all off-policy evaluation algorithms can be decomposed into three fundamental quantities: *contraction rate*, *fixed-point bias*, and *variance*; see Figure 1 for a preliminary illustration. Intuitively, *fixed-point bias* describes the error of an algorithm in the limit of infinite data, *contraction rate* describes the speed at which an algorithm approaches its infinite-data limit, and *variance* describes to what extent randomly observed data can perturb the algorithm.

This decomposition yields an interpretation of the empirical success of uncorrected returns, and an answer to question (i) above; namely, that they are efficiently making a trade-off between fixed-point bias and the other fundamental quantities. Further, this suggests an answer to question (ii) — that we may be able to improve existing off-policy algorithms by incorporating a means of making such a trade-off. This leads us to the development of *C-trace*, a new off-policy algorithm that achieves strong empirical performance in several large-scale environments.

We develop the trade-off framework mentioned above in Section 2, proving the existence of the three fundamental quantities described above, and showing that all off-policy algorithms necessarily make an implicit trade-off between these quantities. We then use this framework to develop new off-policy learning algorithms, α-Retrace and C-trace, in Section 3, and study its contraction and convergence properties. Finally, we demonstrate their empirical effectiveness in tabular domains and when applied to two deep reinforcement learning agents, DQN [Mnih et al., 2015] and R2D2 [Kapturowski et al., 2019], in Section 4.

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108.

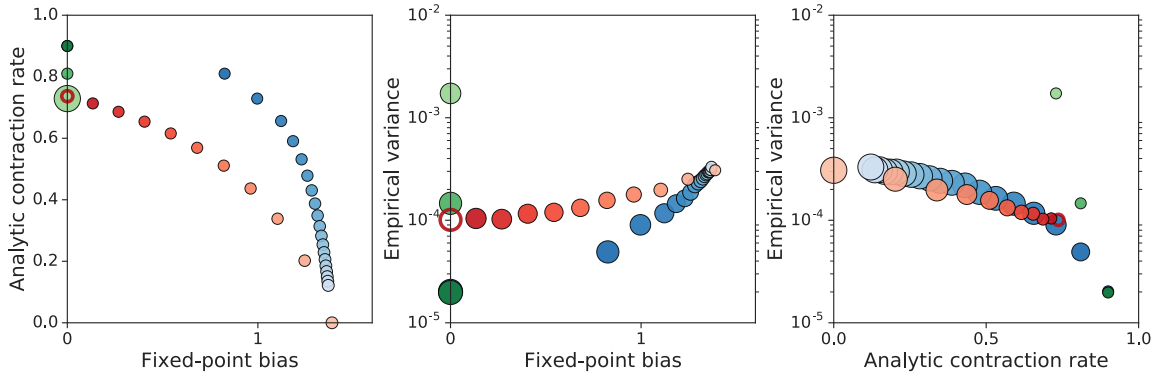

Figure 1: Trade-offs made by n-step uncorrected returns (dark blue [$n = 1$] through to light blue [$n = 20$]), n-step importance corrected returns (dark green [$n = 1$] through to light green [$n = 3$]), Retrace (red open circle). Also pictured is the new method α-Retrace (dark red [$\alpha = 1$] through to light red [$\alpha = 0$]), introduced in Section 3. All quantities are calculated for a fixed evaluation problem in a small, randomly generated MDP; see Appendix Section C.1 for further environment details. In each plot, the magnitude of the points illustrates the relative scale of the third trade-off variable.

1.1 Notation and preliminary definitions

Throughout, we consider a Markov decision process (MDP) with finite state space $\mathcal{X}$, finite action space $\mathcal{A}$, discount factor $\gamma \in [0, 1)$, transition kernel $P : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathcal{X})$, reward distributions $\mathcal{R} : \mathcal{X} \times \mathcal{A} \to \mathscr{P}(\mathbb{R})$, and some initial state distribution $\nu_0 \in \mathscr{P}(\mathcal{X})$. Given a Markov policy $\pi : \mathcal{X} \to \mathscr{P}(\mathcal{A})$, we write $(X_t, A_t, R_t)_{t\geq 0}$ for the process describing the sequence of states visited, actions taken, and rewards received by an agent acting in the MDP according to π, so that $R_t|X_t, A_t \sim \mathcal{R}(X_t, A_t)$ for all $t \geq 0$. Additionally, we write $r(x, a)$ for the expected immediate reward received after taking action a in state x. Given a policy π, the task of *evaluation* is to learn the function $Q^\pi(x, a) = \mathbb{E}_\pi \left[\sum_{t=0}^\infty \gamma^t R_t | X_0 = x, A_0 = a\right]$, where $\mathbb{E}_\pi$ denotes expectation with respect to the distribution over trajectories induced by π. The task of *control* is to identify the Markov policy π^* maximising the quantity $\mathbb{E}\left[Q^\pi(X_0, A_0)\right]$, where $A_0 \sim \pi(\cdot|X_0)$, and $X_0 \sim \nu_0$. We also define the one-step evaluation operator $T^\mu : \mathbb{R}^{\mathcal{X}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{X}\times\mathcal{A}}$ associated with a Markov policy $\mu : \mathcal{X} \to \mathscr{P}(\mathcal{A})$ by

$$(T^\mu Q)(x, a) = \qquad (1)$$
$$r(x, a) + \gamma \sum_{x'\in\mathcal{X}, a'\in\mathcal{A}} P(x'|x, a)\mu(a'|x')Q(x', a'),$$

for all $Q \in \mathbb{R}^{\mathcal{X}\times\mathcal{A}}$, and $(x, a) \in \mathcal{X} \times \mathcal{A}$.

We now briefly give formal definitions of the key concepts we seek to analyse in this paper.

Definition 1.1. An **evaluation update rule** for evaluating a policy π under a behaviour policy μ is a function $\hat{T} : \mathbb{R}^{\mathcal{X}\times\mathcal{A}} \times (\mathcal{X} \times \mathcal{A} \times \mathbb{R})^* \to \mathbb{R}$ that takes as input a value function estimate Q and a trajectory $(x_t, a_t, r_t)_{t\geq 0}$ given by following μ, and outputs an update for $Q(x_0, a_0)$. There is an associated **evaluation operator** $T : \mathbb{R}^{\mathcal{X}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{X}\times\mathcal{A}}$, given by

$$(TQ)(x, a) = \mathbb{E}_\mu \left[\hat{T}(Q, (X_t, A_t, R_t)_{t=0}^\infty) \middle| X_0 = x, A_0 = a\right],$$

for all $Q \in \mathbb{R}^{\mathcal{X}\times a}$ and $(x, a) \in \mathcal{X} \times \mathcal{A}$.

Definition 1.2. The **contraction rate** of an operator $T : \mathbb{R}^{\mathcal{X}\times\mathcal{A}} \to \mathbb{R}^{\mathcal{X}\times\mathcal{A}}$ is given by

$$\Gamma = \sup_{\substack{Q,Q'\in\mathbb{R}^{\mathcal{X}\times\mathcal{A}} \\ Q\neq Q'}} \|TQ - TQ'\|_\infty / \|Q - Q'\|_\infty ,$$

An operator is said to be *contractive* if $\Gamma < 1$. We can also consider state-action contraction rates via the quantities $\sup_{Q\neq Q'} |(TQ)(x, a) - (TQ')(x, a)|/\|Q - Q'\|_\infty$.

Definition 1.3. For a contractive operator T targeting a policy π, the **fixed-point bias** of T is given by $\|Q^\pi - \hat{Q}^\pi\|_2$, where $\hat{Q}^\pi$ is the unique fixed point of T (guaranteed to exist by the contractivity of T).

Definition 1.4. The **variance** of an update rule $\hat{T}$ stochastically approximating an operator T at approximate value function Q and an initial state-action distribution $\nu \in \mathscr{P}(\mathcal{X} \times \mathcal{A})$ is $\mathbb{E}_{(X_0,A_0)\sim\nu}\left[\mathbb{E}_\mu \left[\|\hat{T}(Q, (X_t, A_t, R_t)_{t=0}^\infty) - TQ\|_2^2 \middle| X_0, A_0\right]\right]$.

2 Contraction, bias, and variance

We begin with two motivating examples from recent research in off-policy evaluation methods, illustrating examples of the types of trade-offs we seek to describe in this paper.

n-step uncorrected returns. Recently proposed agents such as Rainbow [Hessel et al., 2018] and R2D2 [Kapturowski et al., 2019] have made use of the *uncorrected n-step return* in constructing off-policy learning algorithms. Consistent with these results, Hernandez-Garcia and Sutton [2019] observed that these uncorrected updates frequently outperformed off-policy corrections. Given an estimate $\hat{Q}$ of the action-value function Q^π, the n-step uncorrected target for $\hat{Q}(x_0, a_0)$, given a trajectory $(x_0, a_0, r_0, x_1, a_1, r_1, \ldots, x_n)$ of experience generated according to behaviour policy μ, is given by

$$\sum_{s=0}^{n-1} \gamma^s r_s + \gamma^n \mathbb{E}_{A \sim \pi(\cdot|x_n)} \left[\hat{Q}(x_n, A) \right] . \tag{2}$$

The adjective *uncorrected* contrasts this update target against the *n-step importance-weighted return* target, which takes the following form:

$$\sum_{s=0}^{n-1} \rho_{1:s} \gamma^s r_s + \rho_{1:n-1} \gamma^n \mathbb{E}_{A \sim \pi(\cdot|x_n)} \left[\hat{Q}(x_n, A) \right] , \tag{3}$$

where we write $\rho_t = \pi(a_t|x_t)/\mu(a_t|x_t)$, and $\rho_{s:t} = \prod_{u=s}^t \rho_u$ for each $1 \leq s \leq t$. Empirically, the former has been observed to work very well in these recent works, whilst the latter is often too unstable to be used; this fact is often attributed to the *high variance* of the importance-weighted update, with the uncorrected update having relatively low variance by comparison. We also observe that the uncorrected update is a stochastic approximation to the operator $(T^\mu)^{n-1} T^\pi$, whilst the importance-weighted update is a stochastic approximation to $(T^\pi)^n$. From this, it follows that under usual stochastic approximation conditions, a sequence of importance-weighted updates will converge to the true action-function Q^π associated with π, whilst the uncorrected updates will converge to the value function of the time-inhomogeneous policy that follows π for a single step, followed by $n-1$ steps of μ, and then repeats; see Proposition B.1 in Appendix Section B for further explanation.

The above discussion shows that we may view the use of uncorrected returns as trading off update *variance* for *accuracy of the operator fixed point*; an example of the classical bias-variance trade-off in statistics and machine learning, albeit in the context of fixed-point iteration.

Retrace. Munos et al. [2016] proposed an off-policy evaluation update target, Retrace, given in its forward-view version by

$$\hat{Q}(x_0, a_0) + \sum_{s \geq 0} \bar{\rho}_{1:s} \gamma^s \Delta_s , \tag{4}$$

where we write $\bar{\rho}_t = \min(1, \rho_t)$, and $\bar{\rho}_{s:t} = \prod_{u=s}^t \bar{\rho}_u$ for each $1 \leq s \leq t$, and define the temporal difference (TD) error Δ_s by

$$\Delta_s \stackrel{\text{def}}{=} r_s + \gamma \mathbb{E}_{A \sim \pi(\cdot|x_{s+1})} \left[\hat{Q}(x_{s+1}, A) \right] - \hat{Q}(x_s, a_s) .$$

By clipping the importance weights associated with each TD error, the variance associated with the update rule is reduced relative to importance-weighted returns, whilst no bias is introduced; the fixed point of the associated Retrace operator remains the true action-value function Q^π. However, the clipping of the importance weights effectively *cuts the traces* in the update, resulting in the update placing less weight on later TD errors, and thus worsening the contraction rate of the corresponding operator. Thus, Retrace can be interpreted as trading off a *reduction in update variance* for a *larger contraction rate*, relative to importance-weighted n-step returns.

We discuss more examples of off-policy learning algorithms in Section 5. We also note that λ-variants of the algorithms described above also exist; for clarity and conciseness, we limit our exposition to the case $\lambda = 1$ in the main paper, noting that the results straightforwardly extend to $\lambda \in (0, 1)$.

We now briefly return to Figure 1, which quantitatively illustrates the trade-offs discussed above. We highlight several interesting observations. Whilst all importance-weighted updates have no fixed-point bias, their variance grows exceptionally quickly with n. Retrace manages to achieve a similar contraction rate to the 3-step importance-weighted update, but without incurring high variance. Our new algorithm, α-Retrace, appears to be Pareto efficient relative to the n-step uncorrected methods in the left-most plot; for any contraction rate that an n-step uncorrected method achieves, there is a value of α such that α-Retrace achieves this contraction rate whilst incurring less fixed-point bias; this is corroborated by further empirical results in Appendix Section B.

2.1 Downstream tasks and bounds

Whilst the trade-offs at the level of individual updates described above are straightforward to describe, in reinforcement learning we are ultimately interested in one of two problems, either *evaluation* or *control*, defined formally below.

The evaluation problem. Given a target policy π, a budget of experience generated from a behaviour policy μ, and a computational budget, compute an accurate approximation $\hat{Q}$ to Q^π, in the sense of incurring low error $\|\hat{Q} - Q^\pi\|$, for some norm $\|\cdot\|$.

The control problem. Given a budget of experience and computation, find a policy π such that expected return under π is maximised.

It is intuitively clear that for each of these problems, an evaluation scheme with low contraction rate, low update variance, and low fixed-point bias is advantageous, but no update is known to possess all three of these attributes simultaneously. What is less clear is how these three properties should be traded off against one another in designing an efficient off-policy learning algorithm. For example, how much fixed-point accuracy should one be willing trade off in exchange for a halved update variance? Such questions, in general, have complicated dependence on the precise update rule, policies in question, and environment, and so it appears unlikely that too much progress can be made in great generality. However, we can provide some some insights from understanding this fundamental trade-off.

Proposition 2.1. Consider the task of evaluation of a policy π under behaviour μ, and consider an update rule $\hat{T}$ which stochastically approximates the application of an operator $\widetilde{T}$, with contraction rate Γ and fixed point $\widetilde{Q}$, to an initial estimate Q. Then we have the following decomposition:

$$\mathbb{E}\left[\|\hat{T}Q - Q^\pi\|_\infty\right] \leq \underbrace{\mathbb{E}\left[\|\hat{T}Q - \widetilde{T}Q\|_2^2\right]^{1/2}}_{\text{(Root) variance}} + \underbrace{\Gamma\|Q - \widetilde{Q}\|_\infty}_{\text{Contraction}} + \underbrace{\|\widetilde{Q} - Q^\pi\|_2}_{\text{Fixed-point bias}} .$$

Note that $\hat{T}$ is arbitrary, and may, for example, represent the n-fold application of a simpler operator. This result gives some sense of how these trade-offs feed into evaluation quality; related decompositions are also possible (see Appendix Section B). We next show that there really is a trade-off to be made, in the sense that it is not possible for an update based on limited data to simultaneously have low variance, contraction rate, and fixed-point bias across a range of MDPs.

Theorem 2.2. Consider an update rule $\hat{T}$ with corresponding operator T, and consider the collection $\mathcal{M} = \mathcal{M}(\mathcal{X}, \mathcal{A}, P, \gamma, R_{\max})$ of MDPs with common state space, action space, transition kernel, and discount factor (but varying rewards, with maximum immediate reward bounded in absolute value by $R_{\max}$). Fix a target policy π, and a random variable Z, the set of transitions used by the operator $\hat{T}$; these could be transitions encountered in a trajectory following the behaviour μ, or i.i.d. samples from the discounted state-action visitation distribution under μ. We denote the mismatch between π and Z at level $\delta \in (0,1)$ by

$$D(Z,\pi,\delta) \stackrel{\text{def}}{=} \max\{d_{(x,a),\pi}(\Omega) \mid \Omega \subseteq \mathcal{X}\times\mathcal{A} \text{ s.t. } \mathbb{P}(Z \cap \Omega \neq \emptyset) \leq \delta\,, (x,a) \in \mathcal{X}\times\mathcal{A}\}\,,$$

where $d_{(x,a),\pi}$ is the discounted state-action visitation distribution for trajectories initialised with (x,a), following π thereafter. Denoting the variance, contraction rate, and fixed-point bias of $\hat{T}$ for a particular MDP $M \in \mathcal{M}$ by $\mathbb{V}(M)$, $\Gamma(M)$ and $B(M)$ respectively, we have

$$\sup_{M\in\mathcal{M}}\left[\sqrt{\mathbb{V}(M)} + \frac{2R_{\max}}{1-\gamma}\Gamma(M) + B(M)\right] \geq \sup_{\delta\in(0,1)} (1-\delta)D(Z,\pi,\delta)R_{\max}/(1-\gamma)\,.$$

In addition to the above results, which we believe to be novel, there is extensive literature exploring particular aspects of these trade-offs, which we discuss further in Section 5. Having made this space of trade-offs between contraction, bias, and variance explicit, a natural questions is how other update rules might be modified to exploit different parts of the space. In particular, incurring some amount of fixed-point bias for reduced variance made by n-step uncorrected returns in Rainbow and R2D2 is particularly effective in practice — is there a way to introduce a similar trade-off in an algorithm with adaptive trace lengths, such as Retrace? We explore this question in the next section.

3 New off-policy updates: α-Retrace and C-trace

The Retrace update in Equation (4) has been observed, in certain scenarios, to cut traces prematurely [Mahmood et al., 2017]; that is, using n-step uncorrected returns for suitable n leads to a superior contraction rate relative to Retrace, outweighing the corresponding incurred bias. A natural question is how Retrace can be modified to overcome this phenomenon. In the language of Section 2, is there a way that Retrace can be adapted so as to trade off contraction rate for fixed-point bias? The reduced contraction rate comes from cases where the truncated importance weights $\min(1, \pi(a_t|x_t)/\mu(a_t|x_t))$ appearing in (4) are small, so a natural way to improve the contraction rate is to move the target policy closer towards the behaviour.

Algorithm 1 α-Retrace for policy iteration

Initialise target policy $\widetilde{\pi}$ and behaviour μ.
for each policy improvement round: **do**
  Select $\alpha \in [0,1]$, and set new target policy $\pi = \alpha\widetilde{\pi} + (1-\alpha)\mu$.
  Learn $\hat{Q}^\pi : \mathcal{X}\times\mathcal{A} \to \mathbb{R}$ via Retrace under behaviour policy μ.
  Set $\widetilde{\pi} = \text{Greedy}(\hat{Q}^\pi)$.
  Set new behaviour policy μ.
end for

To this end, we propose α-Retrace, a family of algorithms that applies Retrace to a target policy given by a mixture of the original target and the behaviour, thus achieving the aforementioned trade-off. In Algorithm 1, we describe how α-Retrace can be used within a (modified) policy iteration scheme for control. Note that 1-Retrace is simply the standard Retrace algorithm. We refer back to Figure 1, the left-most plot of which shows that this mixture coefficient precisely yields a trade-off between fixed-point bias and contraction rate that we sought at the end of Section 2.

The means by which α should be set is left open at this stage; adjusting it allows a trade-off of contraction rate and fixed-point bias. In Section 3.2, we describe a stochastic approximation procedure for updating α online to obtain a desired contraction rate.

Specificity to Retrace. Whilst the mixture target of α-Retrace is a natural choice, we highlight that this choice is in fact specific to the structure of Retrace. In Appendix Section D.1, we visualise trade-offs made by analogous adjustment to the TreeBackup update [Precup et al., 2000], showing that mixing the behaviour policy into the target simply leads to an accumulation of fixed-point bias, with limited benefits in terms of contraction rate or variance.

3.1 Analysis

We now provide several results describing the contraction rate of α-Retrace in detail, and how the fixed-point bias introduced by $\alpha < 1$ may be useful in the case of control tasks. We begin with a preliminary definition.

Definition 3.1. For a state-action pair $(x, a) \in \mathcal{X} \times \mathcal{A}$, Two policies π_1, π_2 are said to be (x, a)-**distinguishable** under a third policy μ if there exists $x' \in \mathcal{X}$ in the support of the discounted state visitation distribution under μ starting from state-action pair (x, a), such that $\pi_1(\cdot|x') \neq \pi_2(\cdot|x')$, and are said to be (x, a)-**indistinguishable** under μ otherwise.

Proposition 3.2. The operator associated with the α-Retrace update for evaluating π given behaviour μ has a state-action-dependent contraction rate of

$$C(\alpha|x,a) \stackrel{\text{def}}{=} 1 - (1-\gamma)\times \mathbb{E}_\mu\left[\sum_{t=0}^{\infty}\gamma^t\prod_{s=1}^{t}((1-\alpha)+\alpha\bar{\rho}_s)\middle|(X_0,A_0)=(x,a)\right], \tag{5}$$

for each $(x, a) \in \mathcal{X} \times \mathcal{A}$. Viewed as a function of $\alpha \in [0, 1]$, this contraction rate is continuous, monotonically increasing, with minimal value 0, and maximal value no greater than γ. Further, the contraction rate is *strictly* monotonic iff π and μ are (x, a)-distinguishable under μ.

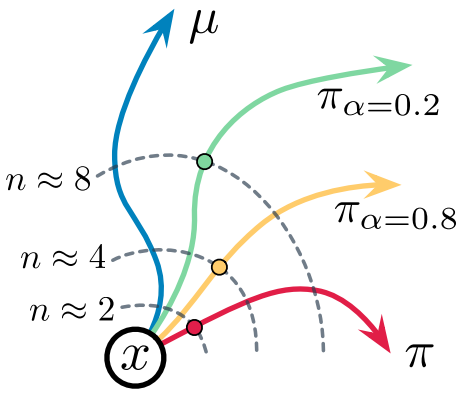

Figure 2: Interpolating between target policy π and behaviour policy μ with $\alpha \in \{0.0, 0.2, 0.8, 1.0\}$ produces different expected trajectories shown by each coloured line. As the mixture policy more closely resembles the behaviour policy, α-Retrace allows more off-policy data to be used (dashed line, numbers indicate expected trace-length), cuts traces (coloured points) later, yielding lower contraction rates equivalent to n-step methods with larger n. C-trace adapts α online to achieve a stable trace length throughout training.

The exact contraction rate of α-Retrace is thus $\sup_{(x,a)\in\mathcal{X}\times\mathcal{A}} C(\alpha|x, a)$, which inherits the continuity and monotonicity properties of the state-action-dependent rates. Our next result motivates the use of α-Retrace within control algorithms.

Proposition 3.3. Consider a target policy π, let μ be the behavioural policy, and assume that there is a unique greedy action $a^*(x) \in \mathcal{A}$ with respect to Q^π at each state x for each $x \in \mathcal{X}$. Then there exists a value of $\alpha \in (0, 1)$ such that the greedy policy with respect to the fixed point of α-Retrace coincides with the greedy policy with respect to Q^π, and the contraction rate for this α-Retrace is no greater than that for 1-Retrace. Further, if π and μ are (x, a)-distinguishable under μ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$, then the contraction rate of α-Retrace is strictly lower than that of 1-Retrace.

3.2 C-trace: adapting α online

An empirical shortcoming of Retrace noted earlier is its tendency to pessimistically cut traces. Adapting the mixture parameter α within α-Retrace yields a natural way to ensure that a desired trace length (or contraction rate) is attained. In this section, we propose C-trace, which uses α-Retrace updates whilst dynamically adjusting α to attain a target contraction rate Γ; a schematic illustration is given in Figure 2.

The contraction rate $\sup_{(x,a)\in\mathcal{X}\times\mathcal{A}} C(\alpha|x, a)$ is difficult to estimate online, so we work instead with the averaged contraction rate $C_\nu(\alpha) = \mathbb{E}_{(X,A)\sim\nu}[C(\alpha|X, A)]$, where

ν is the training distribution over state-action pairs; where clear, we will drop ν from notation. It follows straighforwardly from Proposition 3.2 that $C_\nu(\alpha)$ is monotonic in α. This suggests that a standard Robbins-Monro stochastic approximation update rule for α may be applied to guide $C_\nu(\alpha)$ towards Γ — we describe such a scheme below. To avoid optimisation issues with the constraint $\alpha \in [0, 1]$, we parameterise α as $\sigma(\phi)$, where σ is the standard sigmoid function, and $\phi \in \mathbb{R}$ is an unconstrained real variable. For brevity, we will simply write $\alpha(\phi)$. Since σ is monotonic and continuous, the contraction rate is still monotonic and continuous in ϕ.

Recall from (5) that the contraction rate $C(\alpha|x, a)$ of the α-Retrace operator with target π and behaviour μ can be expressed as an expectation over trajectories following μ, and thus can be unbiasedly approximated using such trajectories; given an i.i.d. sequence of trajectories $(x_t^{(k)}, a_t^{(k)}, r_t^{(k)})_{t\geq 0}$, we write $\hat{C}^{(k)}(\alpha(\phi))$ for the corresponding estimates of $C(\alpha(\phi))$. If the target contraction rate is Γ, we can adjust an initial parameter $\phi_0 \in \mathbb{R}$ using these estimates according to the Robbins-Monro rule

$$\phi_{k+1} = \phi_k - \varepsilon_k \left(\hat{C}^{(k)}(\alpha(\phi_k)) - \Gamma \right) \qquad \forall k \geq 0\,, \quad (6)$$

for some sequence of stepsizes $(\varepsilon_k)_{k=0}^\infty$. The following result gives a theoretical guarantee for the correctness of this procedure.

Proposition 3.4. Let $(x_t^{(k)}, a_t^{(k)}, r_t^{(k)})_{t=0}^\infty$ be an i.i.d. sequence of trajectories following μ, with initial state-action distribution given by ν. Let Γ be a target contraction rate such that $C_\nu(1) \geq \Gamma$. Let the stepsizes $(\varepsilon_k)_{k=0}^\infty$ satisfy the usual Robbins-Monro conditions $\sum_{k=0}^\infty \varepsilon_k = \infty$, $\sum_{k=0}^\infty \varepsilon_k^2 < \infty$. Then for any initial value ϕ_0 following the updates in (6), we have $\phi_k \to \phi^*$ in probability, where $\phi^* \in \mathbb{R}$ is the unique value such that $C_\nu(\alpha(\phi^*)) = \Gamma$.

C-trace thus consists of interleaving α-Retrace evaluation updates with α parameter updates as in (6).

Convergence analysis. It is possible to further develop the theory in Proposition 3.4 to prove convergence of C-trace as a whole, using techniques going back to those of Bertsekas and Tsitsiklis [1996] for convergence of TD(λ), and more recently used by Munos et al. [2016] to prove convergence of a control version of Retrace, as the following result shows.

Theorem 3.5. Assume the same conditions as Proposition 3.4, and additionally that: (i) trajectory lengths have finite second moment; (ii) immediate rewards are bounded. Let $(\phi_k)_{k=0}^\infty$ be defined as in Equation (6) and $(Q_k)_{k=0}^\infty$ be a sequence of Q-functions, with Q_{k+1} obtained from applying Retrace updates targeting $\alpha(\phi_k)\pi + (1 - \alpha(\phi_k))\mu$ to Q_k with trajectory $k+1$, using stepsize ε_k. Then we have $\alpha(\phi_k) \to \alpha(\phi^*) =: \alpha^*$ and $Q_k \to Q^{\alpha^*\pi + (1-\alpha^*)\mu}$ almost surely.

Truncated trajectory corrections. The method described above for adaptation of α is impractical in scenarios where episodes are particularly long, when the MDP is non-episodic, and when only partial segments of trajectories can be processed at once. Since such cases often arise in practice, this motivates modifications to the update of (6). Here, we describe one such modification which will be crucial to the deployment of C-trace in large-scale agents in Section 4. Given a *truncated trajectory* $(x_t, a_t, r_t)_{t=0}^N$, Retrace necessarily must cut traces after at most N time steps, and so can achieve a contraction rate of γ^N at the very lowest. We thus adjust the target contraction rate accordingly, and arrive at the following update:

$$\phi_{k+1} = \phi_k - \varepsilon_k \left(\hat{C}^{(k)}(\alpha(\phi_k)) - \max(\Gamma, \gamma^N) \right)\,. \quad (7)$$

4 Experiments

Having explored the types of trade-offs α-Retrace makes relative to existing off-policy algorithms, we now investigate the performance of these methods in the downstream tasks of evaluation and control described in Section 2.1.

Evaluation. In the left sub-plot of Figure 3, we compare the performance of α-Retrace, n-step uncorrected updates, and n-step importance-weighted updates, for various values of the parameters concerned, at an off-policy evaluation task. In this particular task, the environment is given by a chain MDP (see Appendix Section C.1), the target policy is optimal, and the behaviour is the uniform policy. We plot Q-function L^2 error against number of environment steps; see full details in Appendix Section C.2. Standard error is indicated by the shaded regions.

The best performing methods vary as a function of the number of environment steps experienced. For low numbers of environment steps, the best performing methods are n-step uncorrected updates for large n, and α-Retrace for α close to 0. Intuitively, in this regime, a good contraction rate outweighs fixed-point bias. As the number of environment steps increases, the fixed-point bias kicks in, and the optimally-performing α gradually increases from close to 0 to close to 1. Note that typically the high variance of the importance-weighted updates preclude them from attaining any reasonable level of evaluation error.

Control. In the right sub-plot of Figure 3, we compare the performance of a variety of modified policy iteration methods, each using a different off-policy evaluation

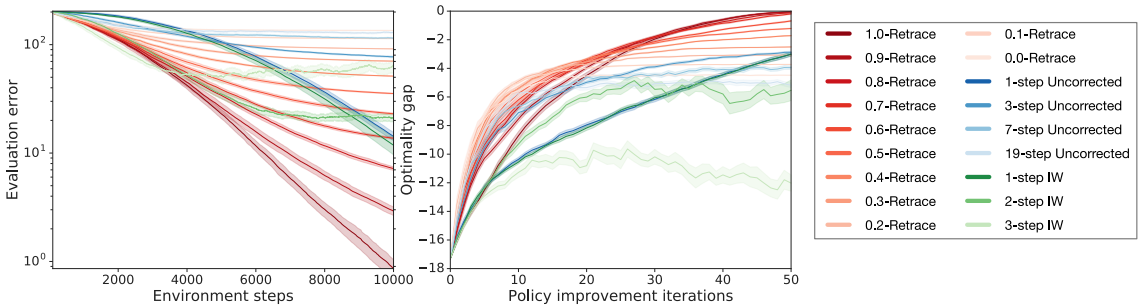

Figure 3: Left: Performance of a variety of off-policy evaluation methods on a small MDP; for further details, see text of Section 4. Right: Performance of a variety of modified policy iteration methods on a small MDP; for further details, see text of Section 4.

method. We use the same MDP as in the evaluation example above, and plot the sub-optimality of the learned policy (measured as difference between expected return under a uniformly random initial state for optimal and learned policies) against the number of policy improvement steps performed. In this experiment, the behaviour policy is fixed as uniform throughout. As with evaluation, we see that initial improvements in policy are strongest with highly-contractive methods incorporating some fixed-point bias, with less-biased approaches catching up (and ultimately surpassing) for greater amounts of environment interaction.

4.1 C-trace-R2D2

To test the performance of our methods at scale, we adapted R2D2 [Kapturowski et al., 2019] to replace the original n-step uncorrected update with Retrace and C-trace. For C-trace we targeted the contraction rate given by an n-step uncorrected update, using a discount rate of $\gamma = 0.997$ and $n = 10$. Based on the Pareto efficiency of α-Retrace relative to n-step uncorrected returns exhibited empirically in small-scale MDPs, we conjectured that this should lead to improved performance. The agent was trained on the Atari-57 suite of environments [Bellemare et al., 2013] with the same experimental setup as in [Kapturowski et al., 2019], a description of which we include in Appendix Section E.1. High-level results are displayed in Figure 4, plotting mean human-normalised performance, median human-normalised performance, and mean human-gap (across the 57 games) against wall-clock training time; detailed results are given in Appendix Section F.1. We also provide empirical verification that C-trace-R2D2 successfully attains its target contraction rate in practice in Appendix Section F.3.

C-trace-R2D2 attains comparable or superior performance relative to R2D2 and Retrace-R2D2 in all three performance measures. Thus, not only does C-trace-R2D2 match state-of-the-art performance for distributed value-based agents on Atari, it also achieves the earlier stated goal of bridging the gap between the performance of uncorrected returns and more principled off-policy algorithms in deep reinforcement learning.

4.2 C-trace-DQN

To illustrate the flexibility of C-trace as an off-policy learning algorithm, we also demonstrate its performance within a DQN architecture [Mnih et al., 2015]. We use Double DQN [Van Hasselt et al., 2016] as a baseline, and modify the one-step Q-learning rule to use n-step uncorrected returns, Retrace, and C-trace. As for the R2D2 experiment, we set the C-trace contraction target using $n = 10$, demonstrating the robustness of this C-trace hyperparameter across different architectures. Further, we found the behaviour of C-trace to be generally robust to the choice of n; see Appendix Section F.2. Full experimental specifications are given in Appendix Section E.2, with detailed results in Appendix Section F.2; a high-level summary is displayed in Figure 4. All sequence-based methods significantly outperform Double DQN, as we would expect. We notice that the performance gap between n-step and Retrace is not as large here as for R2D2. A possible explanation for this is that the data distribution used by DQN is typically "more off-policy" than R2D2, as the latter uses a distributed set of actors to increase data throughput. As with the R2D2 experiments we see that C-trace-DQN achieves similar learning speed as the targeted n-step update, but with improved final performance. One interpretation of these results is that the improved contraction rate of C-trace allows it to learn significantly faster than Retrace, while the better fixed-point error allows it to find a better long-term solution than n-step uncorrected.

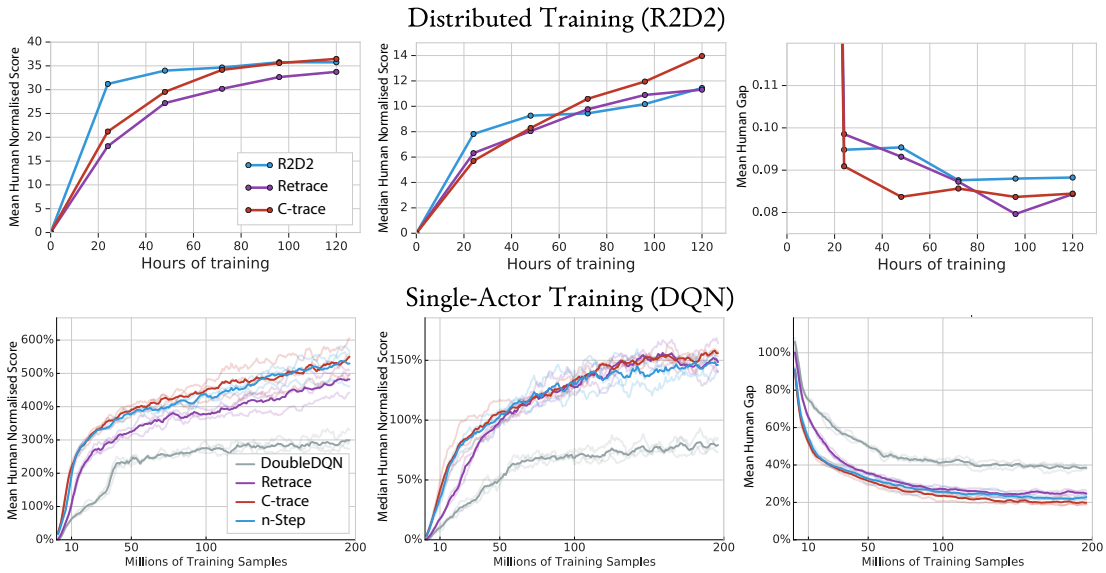

Figure 4: High-level performance of variants of R2D2 (top row) and DQN (bottom row) on the Atari suite of environments. R2D2-based methods are averages of two seeds. DQN-based methods are averages of three seeds. (**Left**) Mean human-normalised score, (**Centre**) median human-normalised score, and (**Right**) human gap.

5 Related work

A central observation of this work is that the fixed-point bias can be explicitly traded-off to improve contraction rates. To our understanding, this is the first work to directly study this possibility, and further to draw attention to three fundamental quantities to be traded-off in off-policy learning. However, investigating trade-offs in off-policy RL, and in particular parametrising methods to allow a spectrum of algorithms is a long-standing research topic [Sutton and Barto, 2018]. The most closely related methods come from a line of work that consider the bias-variance trade-off due to bootstrapping. In our framework, we understand this as a trade-off between variance and contraction rate, *but without modifying the fixed-point*. The recently introduced $Q(\sigma)$ algorithm uses the σ hyperparameter to mix between importance-weighted n-step SARSA and TreeBackup [De Asis et al., 2018]. In another recent related approach, Shi et al. [2019] uses σ to mix between TreeBackup(λ) and $Q(\lambda)$, although neither of these approaches adaptively set σ based on observed data. We have developed an adaptive method for adjusting α to achieve a desired trace length, and believe an interesting direction for future work would be to develop the adaptive methods described in this paper for use in other families of off-policy learning algorithms.

Conservatively updating policies within control algorithms is a well-established practice; Kakade and Langford [2002] consider a trust-region method for policy improvement, motivated by inexact policy evaluation due to function approximation. In contrast, in this work we consider regularised policy improvement as a means of improving evaluation of future policies, even in the absence of function approximation. More recently, this approach also led to several advances in policy gradient methods [Schulman et al., 2015, 2017] based on trust regions. Although not the focus of this work, there has been also been much progress on correcting state-visitation distributions [Sutton et al., 2016, Thomas and Brunskill, 2016, Hallak and Mannor, 2017, Liu et al., 2018], another form of off-policy correction important in function approximation, as illustrated in the classic counterexample of Baird [1995].

6 Conclusion

We have highlighted the fundamental role of variance, fixed-point bias, and contraction rate in off-policy learning, and described how existing methods trade off these quantities. With this perspective, we developed novel off-policy learning methods, α-Retrace and C-trace, and incorporated the latter into several deep RL agents, leading to strong empirical performance.

Interesting questions for future work include applying the adaptive ideas underlying C-trace to other families of off-policy algorithms, investigating whether there exist new off-policy learning algorithms in unexplored areas of the space of trade-offs, and developing a deeper understanding of the relationship between these fundamental properties of off-policy learning algorithms and downstream performance on large-scale control tasks.

Acknowledgements

Thanks in particular to Hado van Hasselt for detailed comments and suggestions on an earlier version of this paper, and thanks to Bernardo Avila Pires, Diana Borsa, Steven Kapturowski, Bilal Piot, Tom Schaul, and Yunhao Tang for interesting conversations during the course of this work. Thanks also to the anonymous reviewers for helpful comments during the review process.

References

TW Archibald, KIM McKinnon, and LC Thomas. On the generation of Markov decision processes. *Journal of the Operational Research Society*, 46(3):354–361, 1995.

Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine Learning Proceedings*. Elsevier, 1995.

M. G. Bellemare, Y. Naddaf, J. Veness, and M. Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, jun 2013.

Dimitri P Bertsekas and John N Tsitsiklis. *Neuro-dynamic programming*, volume 5. Athena Scientific Belmont, MA, 1996.

Shalabh Bhatnagar, Richard S Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, 45(11):2471–2482, 2009.

Kristopher De Asis, J Fernando Hernandez-Garcia, G Zacharias Holland, and Richard S Sutton. Multi-step reinforcement learning: A unifying algorithm. In *AAAI Conference on Artificial Intelligence*, 2018.

Miroslav Dudík, Dumitru Erhan, John Langford, and Lihong Li. Doubly robust policy evaluation and optimization. *Statistical Science*, 29(4):485–511, 2014.

Mehrdad Farajtabar, Yinlam Chow, and Mohammad Ghavamzadeh. More robust doubly robust off-policy evaluation. In *International Conference on Machine Learning*, 2018.

Matthieu Geist and Bruno Scherrer. Off-policy learning with eligibility traces: A survey. *The Journal of Machine Learning Research*, 15(1):289–333, 2014.

Assaf Hallak and Shie Mannor. Consistent on-line off-policy evaluation. In *International Conference on Machine Learning*, 2017.

J Fernando Hernandez-Garcia and Richard S Sutton. Understanding multi-step deep reinforcement learning: A systematic study of the DQN target. *arXiv*, 2019.

Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In *AAAI Conference on Artificial Intelligence*, 2018.

Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, 2002.

Steven Kapturowski, Georg Ostrovski, Will Dabney, John Quan, and Rémi Munos. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations*, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Qiang Liu, Lihong Li, Ziyang Tang, and Dengyong Zhou. Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Neural Information Processing Systems*, 2018.

Ashique Rupam Mahmood, Huizhen Yu, and Richard S Sutton. Multi-step off-policy learning without importance sampling ratios. *arXiv*, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Neural Information Processing Systems*, 2016.

Bilal Piot, Matthieu Geist, and Olivier Pietquin. Difference of convex functions programming for reinforcement learning. In *Neural Information Processing Systems*, 2014.

Doina Precup, Rich Sutton, and Satinder Singh. Eligibility traces for off-policy policy evaluation. In *International Conference on Machine Learning*, 2000.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *Ann. Math. Statist.*, 22(3): 400–407, 09 1951.

John Schulman, Sergey Levine, Pieter Abbeel, Michael I Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec

Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv*, 2017.

Longxiang Shi, Shijian Li, Longbing Cao, Long Yang, and Gang Pan. TBQ(σ): Improving efficiency of trace utilization for off-policy reinforcement learning. In *International Conference on Autonomous Agents and Multiagent Systems*, 2019.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Richard S Sutton, A Rupam Mahmood, and Martha White. An emphatic approach to the problem of off-policy temporal-difference learning. *Journal of Machine Learning Research*, 17(1):2603–2631, 2016.

Csaba Szepesvári. Algorithms for reinforcement learning. *Synthesis lectures on artificial intelligence and machine learning*, 4(1):1–103, 2010.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, 2016.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double Q-learning. In *AAAI Conference on Artificial Intelligence*, 2016.

Ziyu Wang, Tom Schaul, Matteo Hessel, Hado van Hasselt, Marc Lanctot, and Nando de Freitas. Dueling network architectures for deep reinforcement learning. In *International Conference on Machine Learning*, 2016.