

# A Comparative Analysis of Expected and Distributional Reinforcement Learning

Clare Lyle,<sup>1</sup> Marc G. Bellemare,<sup>2</sup> Pablo Samuel Castro<sup>2</sup>

<sup>1</sup>University of Oxford (work done while at Google Brain)

<sup>2</sup>Google Brain

clare.lyle@cs.ox.ac.uk, bellemare@google.com, psc@google.com

## Abstract

Since their introduction a year ago, distributional approaches to reinforcement learning (distributional RL) have produced strong results relative to the standard approach which models expected values (expected RL). However, aside from convergence guarantees, there have been few theoretical results investigating the reasons behind the improvements distributional RL provides. In this paper we begin the investigation into this fundamental question by analyzing the differences in the tabular, linear approximation, and non-linear approximation settings. We prove that in many realizations of the tabular and linear approximation settings, distributional RL behaves exactly the same as expected RL. In cases where the two methods behave differently, distributional RL can in fact hurt performance when it does not induce identical behaviour. We then continue with an empirical analysis comparing distributional and expected RL methods in control settings with non-linear approximators to tease apart where the improvements from distributional RL methods are coming from.

## 1 Introduction

The distributional perspective, in which one models the distribution of returns from a state instead of only its expected value, was recently introduced by (Bellemare, Dabney, and Munos 2017). The first distributional reinforcement learning algorithm, C51, saw dramatic improvements in performance in many Atari 2600 games when compared to an algorithm that only modelled expected values (Bellemare, Dabney, and Munos 2017). Since then, additional distributional algorithms have been proposed, such as quantile regression (Dabney et al. 2017) and implicit quantile networks (Dabney et al. 2018), with many of these improving on the results of C51. The abundance of empirical results make it hard to dispute that taking the distributional perspective is helpful in deep reinforcement learning problems, but theoretical motivation for this perspective is comparatively scarce. Possible reasons for this include the following, proposed by (Bellemare, Dabney, and Munos 2017).

1. **Reduced chattering:** modeling a distribution may reduce prediction variance, which may help in policy iteration.
2. **Improved optimization behaviour:** distributions may present a more stable learning target, or in some cases

(e.g. the softmax distribution used in the C51 algorithm) have a regularizing effect in optimization for neural networks.

3. **Auxiliary tasks:** the distribution offers a richer set of predictions for learning, serving as a set of auxiliary tasks which is tightly coupled to the reward.

Initial efforts to provide a theoretical framework for the analysis of distributional algorithms demonstrated their convergence properties (Rowland et al. 2018), and did not directly compare their expected performance to expected algorithms. Indeed, even experimental results supporting the distributional perspective have largely been restricted to the deep reinforcement learning setting, and it is not clear whether the benefits of the distributional perspective also hold in simpler tasks. In this paper we continue lifting the veil on this mystery by investigating the behavioural differences between distributional and expected RL, and whether these behavioural differences necessarily result in an advantage for distributional methods.

## 2 Background

We model the reinforcement learning problem as an agent interacting with an environment so as to maximize cumulative discounted reward. We formalize the notion of an environment with a Markov Decision Process (MDP) defined as the tuple  $(\mathcal{X}, \mathcal{A}, R, P, \gamma)$ , where  $\mathcal{X}$  denotes the state space,  $\mathcal{A}$  the set of possible actions,  $R : \mathcal{X} \times \mathcal{A} \rightarrow \text{Dist}([-R_{MAX}, R_{MAX}])$  is a stochastic reward function mapping state-action pairs to a distribution over a set of bounded rewards,  $P$  the transition probability kernel, and  $\gamma \in [0, 1)$  the discount factor.

We denote by  $\pi : \mathcal{X} \rightarrow \text{Dist}(\mathcal{A})$  a stochastic policy mapping states to a distribution over actions (i.e.  $\pi(a|x)$  is the agent’s probability of choosing action  $a$  in state  $x$ ). We will use the notation  $Q$  to refer to state-action value function, which has the type  $Q : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}$ . The value of a specific policy  $\pi$  is given by the value function  $Q^\pi$ , defined as the discounted sum of expected future rewards after choosing action  $a$  from state  $s$  and then following  $\pi$

$$Q^\pi(x, a) := \mathbb{E}_{\pi, P} \left[ \sum_{t=0}^{\infty} \gamma^t R(x_t, a_t) \mid x_0 = x, a_0 = a \right].$$

One can also express this value as the fixed point of the Bellman operator  $T^\pi$  (Bellman 1957), defined as

$$T^\pi Q(x, a) := \mathbb{E}[R(x, a)] + \gamma \sum_{x', a'} P(x'|x, a) \pi(a'|x') Q(x', a').$$

The Bellman operator  $T^\pi$  depends on the policy  $\pi$ , and is used in *policy evaluation* (Sutton and Barto 1998). When we seek to improve the current policy, we enter the *control* setting. In this setting, we modify the previous Bellman operator to obtain the Bellman optimality operator  $T^*$ , given by

$$T^* Q(x, a) := \mathbb{E}[R(x, a)] + \gamma \sum_{x'} P(x'|x, a) \max_{a'} Q(x', a').$$

In many problems of interest, we do not have a full model of the MDP, and instead use a family of stochastic versions of the Bellman operator called temporal difference (TD) updates (Sutton and Barto 1998). We will focus on the SARSA update (Rummery and Niranjan 1994), defined as follows. We fix a policy  $\pi$  and let  $(x_t, a_t, r_t, x_{t+1}, a_{t+1})$  be a sampled transition from the MDP, where  $r_t \sim R(x_t, a_t)$  is a realized reward and  $a_{t+1} \sim \pi(\cdot|x_{t+1})$ . We let  $\alpha_t$  be a step size parameter for time  $t$ . Then given a value function estimate  $Q_t : \mathcal{X} \times \mathcal{A} \rightarrow \mathbb{R}$  at time  $t$ , the SARSA update gives the new estimate  $Q_{t+1}$ :

$$Q_{t+1}(x_t, a_t) = (1 - \alpha_t) Q_t(x_t, a_t) + \alpha_t (r_t + \gamma Q_t(x_{t+1}, a_{t+1})). \quad (1)$$

Under certain conditions on the MDP and  $\alpha_t$ , SARSA converges to  $Q^\pi$  (Bertsekas 1996).

Semi-gradient SARSA updates extend the SARSA update from the tabular to the function approximation setting. We consider a parameter vector  $\theta_t$ , and feature vectors  $\phi_{x,a}$  for each  $(x, a) \in \mathcal{X} \times \mathcal{A}$  such that

$$Q_t(x, a) = \theta_t^T \phi_{x,a}.$$

Given  $\theta_t$ ,  $\theta_{t+1}$  is given by the *semigradient* update (Sutton and Barto 1998)

$$\theta_{t+1} := \theta_t - \alpha_t (\theta_t^T \phi_{x_t, a_t} - r_t + \gamma \theta_t^T \phi_{x_{t+1}, a_{t+1}}) \phi_{x_t, a_t}. \quad (2)$$

Instead of considering only the expected return from a state-action pair, one can consider the distribution of returns. We will use the notation  $Z : \mathcal{X} \times \mathcal{A} \rightarrow \text{Dist}(\mathbb{R})$  to denote a *return distribution function*. We can then construct an analogous Bellman operator for these functions, as shown by (Bellemare, Dabney, and Munos 2017) and termed the distributional Bellman operator,  $T_D^\pi$ :

$$T_D^\pi Z(x, a) \stackrel{D}{=} R(x, a) + \gamma Z(X', A') \quad (3)$$

where  $X', A'$  are the random variables corresponding to the next state and action. This is equality in distribution, and not an equality of random variables.

Analogous to the expected Bellman operator, repeated application of the distributional Bellman operator can be shown to converge to the true distribution of returns (Bellemare, Dabney, and Munos 2017). Later work showed the convergence of stochastic updates in the distributional setting (Rowland et al. 2018). The proof of convergence of the distributional Bellman operator uses a contraction argument, which in the

distributional setting requires us to be careful about how we define the distance between two return distribution functions.

Probability divergences and metrics capture this notion of distance. The theoretical properties of some probability distribution metrics have been previously explored by (Gibbs and Su 2002), and the Wasserstein distance in particular studied further in the context of MDPs (Ferns et al. 2012) as well as in the generative model literature (Arjovsky, Chintala, and Bottou 2017). The Wasserstein distance also appears in the distributional reinforcement learning literature, but we omit its definition in favour of the related Cramér distance, whose properties make it more amenable to the tools we use in our analysis.

The C51 algorithm uses the cross-entropy loss function to achieve promising performance in Atari games; however, the role of the cross-entropy loss in distributional RL has not been the subject of much theoretical analysis. We will use primarily the Cramér distance (Székely 2003) in the results that follow, which has been studied in greater depth in the distributional RL literature. Motivations for the use of this distance have been previously outlined for generative models (Bellemare et al. 2017).

**Definition 1** (Cramér Distance). *Let  $P, Q$  be two probability distributions with Cumulative Distribution Functions (CDFs)  $F_P, F_Q$ . The Cramér metric  $\ell_2$  between  $P$  and  $Q$  is defined as follows:*

$$\ell_2(P, Q) = \sqrt{\int_{\mathbb{R}} (F_P(x) - F_Q(x))^2 dx}$$

*We will overload notation and write equivalently  $\ell_2(P, Q) \equiv \ell_2(F_P, F_Q)$  or, when  $X$  and  $Y$  are random variables with laws  $P$  and  $Q$ ,  $\ell_2(P, Q) \equiv \ell_2(X, Y)$ .*

Practically, distributional reinforcement learning algorithms require that we approximate distributions. There are many ways one can do this, for example by predicting the quantiles of the return distribution (Dabney et al. 2017). In our analysis we will focus on the class of categorical distributions with finite support. Given some fixed set  $\mathbf{z} = \{z_1, \dots, z_K\} \in \mathbb{R}^K$  with  $z_1 \leq z_2 \leq \dots \leq z_K$ , a categorical distribution  $P$  with support  $\mathbf{z}$  is a mixture of Dirac measures on each of the  $z_i$ 's, having the form

$$P \in \mathcal{Z}_{\mathbf{z}} := \left\{ \sum_{i=1}^K \alpha_i \delta_{z_i} : \alpha_i \geq 0, \sum_{i=1}^K \alpha_i = 1 \right\}. \quad (4)$$

Under this class of distributions, the Cramér distance becomes a finite sum

$$\ell_2(F_P, F_Q) = \sqrt{\sum_{i=1}^{K-1} (z_{i+1} - z_i) (F_P(z_i) - F_Q(z_i))^2}$$

which amounts to a weighted Euclidean norm between the CDFs of the two distributions. When the atoms of the support are equally spaced apart, we get a scalar multiple of the Euclidean distance between the vectors of the CDFs.

We can use the Cramér distance to define a projection onto a fixed categorical support  $\mathbf{z}$  (Rowland et al. 2018).

**Definition 2** (Cramér Projection). *Let  $\mathbf{z}$  be an ordered set of  $K$  real numbers. For a Dirac measure  $\delta_y$ , the Cramér projection  $\Pi_C(\delta_y)$  onto the support  $\mathbf{z}$  is given by:*

$$\Pi_C(\delta_y) = \begin{cases} \delta_{z_1} & \text{if } y \leq z_1 \\ \frac{z_{i+1}-y}{z_{i+1}-z_i} \delta_{z_i} + \frac{y-z_i}{z_{i+1}-z_i} \delta_{z_{i+1}} & \text{if } z_i < y \leq z_{i+1} \\ \delta_{z_K} & \text{if } y > z_K \end{cases}$$

The operator  $\Pi_C$  has two notable properties: first, as hinted by the name *Cramér projection*, it produces the distribution supported on  $\mathbf{z}$  which minimizes the Cramér distance to the original distribution. Second, if the support of the distribution is contained in the interval  $[z_1, z_K]$ , we can show that the Cramér projection preserves the distribution’s expected value 1. It is thus a natural approximation tool for categorical distributions.

**Proposition 1.** *Let  $\mathbf{z} \in \mathbb{R}^k$ , and  $P$  be a mixture of Dirac distributions (see Eq. 4) whose support is contained in the interval  $[z_1, z_K]$ . Then the Cramér projection  $\Pi_C(P)$  onto  $\mathbf{z}$  is such that*

$$\mathbb{E}[\Pi_C(P)] = \mathbb{E}[P]$$

The Cramér projection is implicit in the C51 algorithm, introduced by (Bellemare, Dabney, and Munos 2017). The C51 algorithm uses a deep neural network to compute a softmax distribution, then updates its weights according to the cross-entropy loss between its prediction and a sampled target distribution, which is then projected onto the support of the predicted distribution using the Cramér projection.

### 3 The Coupled Updates Method

We are interested in the behavioural differences (or lack thereof) between distributional and expected RL algorithms. We will study these differences through a methodology where we couple the experience used by the update rules of the two algorithms.

Under this methodology, we consider pairs of agents: one that learns a value function (the *expectation learner*) and one that learns a value distribution (the *distribution learner*). The output of the first learner is a sequence of value functions  $Q_1, Q_2, \dots$ . The output of the second is a sequence of value distributions  $Z_1, Z_2, \dots$ . More precisely, each sequence is constructed via some update rule:

$$Q_{t+1} := U_E(Q_t, \omega_t) \quad Z_{t+1} := U_D(Z_t, \omega_t),$$

from initial conditions  $Q_0$  and  $Z_0$ , respectively, and where  $\omega_t$  is drawn from some sample space  $\Omega$ . These update rules may be deterministic (for example, an application of the Bellman operator) or random (a sample-based update such as TD-learning). Importantly, however, the two updates may be coupled through the common sample  $\omega_t$ . Intuitively,  $\Omega$  can be thought of as the source of randomness in the MDP.

Key to our analysis is to study update rules that are analogues of one another. If  $U_E$  is the Bellman operator, for example, then  $U_D$  is the distributional Bellman operator. More generally speaking, we will distinguish between *model-based* update rules, which do not depend on  $\omega_t$ , and *sample-based* update rules, which do. In the latter case, we will

assume access to a scheme that generates sample transitions based on the sequence  $\omega_1, \omega_2, \dots$ , that is, a generator  $G : \omega_1, \dots, \omega_t \mapsto (x_t, a_t, r_t, x_{t+1}, a_{t+1})$ . Under this scheme, a pair  $U_E, U_D$  of sampled-based update rules receive exactly the same sample transitions (for each possible realization); hence the *coupled updates method*, inspired by the notion of coupling from the probability theory literature (Thorisson 2000).

The main question we will answer is: *which analogue update rules preserve expectations?* Specifically, we write

$$Z \stackrel{\mathbb{E}}{=} Q \iff \mathbb{E}[Z(x, a)] = Q(x, a) \quad \forall (x, a) \in \mathcal{X} \times \mathcal{A}.$$

We will say that analogue rules  $U_D$  and  $U_E$  are *expectation-equivalent* if, for all sequences  $(\omega_t)$ , and for all  $Z_0$  and  $Q_0$ ,

$$Z_0 \stackrel{\mathbb{E}}{=} Q_0 \iff Z_t \stackrel{\mathbb{E}}{=} Q_t \quad \forall t \in \mathbb{N}.$$

Our coupling-inspired methodology will allow us to rule out a number of common hypotheses regarding the good performance of distributional reinforcement learning:

**Distributional RL reduces variance.** By our coupling argument, any expectation-equivalent rules  $U_D$  and  $U_E$  produce exactly the same sequence of expected values, *along each sample trajectory*. The distributions of expectations relative to the random draws  $\omega_1, \omega_2, \dots$  are identical and therefore,  $\text{Var}[\mathbb{E}Z_t(x, a)] = \text{Var}[Q_t(x, a)]$  everywhere, and  $U_D$  does not produce lower variance estimates.

**Distributional RL helps with policy iteration.** One may imagine that distributional RL helps identify the best action.

But if  $Z_t \stackrel{\mathbb{E}}{=} Q_t$  everywhere, then also greedy policies based on  $\arg \max Q_t(x, \cdot)$  and  $\arg \max \mathbb{E}Z_t(x, \cdot)$  agree. Hence our results (presented in the context of policy evaluation) extend to the setting in which actions are selected on the basis of their expectation (e.g.  $\epsilon$ -greedy, softmax).

**Distributional RL is more stable with function approximation.** We will use the coupling methodology to provide evidence that, at least combined with linear function approximation, distributional update rules do not improve performance.

## 4 Analysis of Behavioural Differences

Our coupling-inspired methodology provides us with a stable framework to perform a theoretical investigation of the behavioural differences between distributional and expected RL. We use it through a progression of settings that will gradually increase in complexity to shed light on what causes distributional algorithms to behave differently from expected algorithms. We consider three axes of complexity: 1) how we approximate the state space, 2) how we represent the distribution, and 3) how we perform updates on the predicted distribution function.

### 4.1 Tabular models

We first consider tabular representations, which uniquely represent the predicted return distribution at each state-action pair. We start with the simplest class of updates, that of the expected and distributional Bellman operators. Here and below we will write  $\mathcal{Z}$  for the space of bounded return distribution

functions and  $\mathcal{Q}$  for the space of bounded value functions. We begin with results regarding two model-based update rules.

**Proposition 2.** *Let  $Z_0 \in \mathcal{Z}$  and  $Q_0 \in \mathcal{Q}$ , and suppose that  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$ . If*

$$Z_{t+1} := T_{\mathcal{D}}^{\pi} Z_t \quad Q_{t+1} := T^{\pi} Q_t,$$

*then also  $Z_t \stackrel{\mathbb{E}}{=} Q_t \forall t \in \mathbb{N}$ .*

See the supplemental material for the proof of this result and those that follow.

We next consider a categorical, tabular representation where the distribution at each state-action pair is stored explicitly but, as per Equation (4), restricted to the finite support  $\mathbf{z} = \{z_1, \dots, z_K\}$ , (Rowland et al. 2018). Unlike the tabular representation of Prop. 2, this algorithm has a practical implementation; however, after each Bellman update we must project back the result into the space of those finite-support distributions, giving rise to a projected operator.

**Proposition 3.** *Suppose that the finite support brackets the set of attainable value distributions, in the sense that  $z_1 \leq -\frac{R_{\max}}{1-\gamma}$  and  $z_K \geq \frac{R_{\max}}{1-\gamma}$ . Define the projected distributional operator*

$$T_C^{\pi} := \Pi_C T_{\mathcal{D}}^{\pi}.$$

*Suppose  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$ , for  $Z_0 \in \mathcal{Z}$ ,  $Q_0 \in \mathcal{Q}$ . If*

$$Z_{t+1} := T_C^{\pi} Z_t \quad Q_{t+1} := T^{\pi} Q_t,$$

*then also  $Z_t \stackrel{\mathbb{E}}{=} Q_t \forall t \in \mathbb{N}$ .*

Next, we consider sample-based update rules, still in the tabular setting. These roughly correspond to the Categorical SARSA algorithm whose convergence was established by (Rowland et al. 2018), with or without the projection step. Here we highlight the expectation-equivalence of this algorithm to the classic SARSA algorithm (Rummery and Niranjan 1994).

For these results we will need some additional notation. Consider a sample transition  $(x_t, a_t, r_t, x_{t+1}, a_{t+1})$ . Given a random variable  $Y$ , denote its probability distribution by  $P_Y$  and its cumulative distribution function by  $F_Y$ , respectively. With some abuse of notation we extend this to value distributions and write  $P_Z(x, a)$  and  $F_Z(x, a)$  for the probability distribution and cumulative distribution function, respectively, corresponding to  $Z(x, a)$ . Finally, let  $Z'_t(x_t, a_t)$  be a random variable distributed like the target  $r_t + \gamma Z_t(x_{t+1}, a_{t+1})$ , and write  $\Pi_C Z'_t(x, a)$  for its Cramér projection onto the support  $\mathbf{z}$ .

**Proposition 4.** *Suppose that  $Z_0 \in \mathcal{Z}$ ,  $Q_0 \in \mathcal{Q}$  and  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$ . Given a sample transition  $(x_t, a_t, r_t, x_{t+1}, a_{t+1})$  consider the mixture update*

$$P_{Z_{t+1}}(x, a) := \begin{cases} (1 - \alpha_t) P_{Z_t}(x, a) + \alpha_t P_{Z'_t}(x_t, a_t) \\ P_{Z_t}(x, a) & \text{if } x, a \neq x_t, a_t \end{cases}$$

*and the SARSA update*

$$Q_{t+1}(x_t, a_t) := \begin{cases} Q_t(x_t, a_t) + \alpha_t \delta_t \\ Q_t(x, a) & \text{if } x, a \neq x_t, a_t \end{cases}$$

*where  $\delta_t := (r_t + \gamma Q_t(x_{t+1}, a_{t+1}) - Q_t(x_t, a_t))$ , then also  $Z_t \stackrel{\mathbb{E}}{=} Q_t \forall t \in \mathbb{N}$ .*

**Proposition 5.** *Suppose that  $Z_0 \in \mathcal{Z}$ ,  $Q_0 \in \mathcal{Q}$ ,  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$ , that  $\mathbf{z}$  brackets the set of attainable value distributions, and  $P_{Z'_t}$  in Prop. 4 is replaced by the projected target  $P_{\Pi_C Z'_t}$ . Then also  $Z_t \stackrel{\mathbb{E}}{=} Q_t \forall t \in \mathbb{N}$ .*

Together, the propositions above show that there is no benefit, at least in terms of modelling expectations, to using distributional RL in a tabular setting when considering the distributional analogues of update rules in common usage in reinforcement learning.

Next we turn our attention to a slightly more complex case, where distributional updates correspond to a semi-gradient update. In the expected setting, the mixture update of Prop. 4 corresponds to taking the semi-gradient of the squared loss of the temporal difference error  $\delta_t$ . While there is no simple notion of semi-gradient when  $Z$  is allowed to represent arbitrary distributions in  $\mathcal{Z}$ , there is when we consider a categorical representation, which is a finite object when  $\mathcal{X}$  and  $\mathcal{A}$  are finite (specifically,  $Z$  can be represented by  $|\mathcal{X}| |\mathcal{A}| K$  real values).

To keep the exposition simple, in what follows we ignore the fact that semi-gradient updates may yield an object which is not a probability distribution proper. In particular, the arguments remain unchanged if we allow the learner to output a signed distribution, as argued by (Bellemare et al. 2019).

**Definition 3** (Gradient of Cramér Distance). *Let  $Z, Z' \in \mathcal{Z}$  be two categorical distributions supported on  $\mathbf{z}$ . We define the gradient of the squared Cramér distance with respect to the CDF of  $Z$ , denoted  $\nabla_F \ell_2^2(Z, Z') \in \mathbb{R}^K$  as follows:*

$$\nabla_F \ell_2^2(Z, Z')[i] := \frac{\partial}{\partial F(z_i)} \ell_2^2(Z, Z').$$

*Similarly,*

$$\nabla_P \ell_2^2(Z, Z')[i] := \frac{\partial}{\partial P(z_i)} \ell_2^2(Z, Z').$$

We say that the categorical support  $\mathbf{z}$  is  $c$ -spaced if  $z_{i+1} - z_i = c$  for all  $i$  (recall that  $z_{i+1} \geq z_i$ ).

**Proposition 6.** *Suppose that the categorical support  $\mathbf{z}$  is  $c$ -spaced. Let  $Z_0 \in \mathcal{Z}$ ,  $Q_0 \in \mathcal{Q}$  be such that  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$ . Suppose that  $Q_{t+1}$  is updated according to the SARSA update with step-size  $\alpha_t$ . Let  $Z'_t$  be given by  $\Pi_C(r_t + \gamma Z_t(x_{t+1}, a_{t+1}))$ . Consider the CDF gradient update rule*

$$F_{Z_{t+1}}(x, a) := \begin{cases} F_{Z_t}(x, a) + \alpha'_t \nabla_F \ell_2^2(Z_t(x_t, a_t), Z'_t(x_t, a_t)) \\ F_{Z_t}(x, a) & \text{if } x, a \neq x_t, a_t. \end{cases}$$

*If  $\alpha'_t = \frac{\alpha_t}{2c}$ , then also  $Z_t \stackrel{\mathbb{E}}{=} Q_t \forall t \in \mathbb{N}$ .*

Prop. 6 shows that there exists a semi-gradient update which is expectation-equivalent to the SARSA update, with only a change of step-size required. While this is not too surprising ((Rowland et al. 2018) remarked on the relationship between the mixture update of Prop. 4 and the CDF semi-gradient), the result does highlight that the equivalence continues to hold even in gradient-based settings. The resemblance stops here, however, and we now come to our first negative result.

**Proposition 7.** *Suppose the CDF gradient in update rule of Prop. 6 is replaced by the PDF gradient  $\nabla_{P\ell_2^2}(Z_t, Z'_t)$ . Then for each choice of step-size  $\alpha'$  there exists an MDP  $M$  and a time step  $t \in \mathbb{N}$  for which  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$  but  $Z_t \neq Q_t$ .*

The counterexample used in the proof of Prop. 7 illustrates what happens when the gradient is taken w.r.t. the probability mass: some atoms of the distribution may be assigned negative probabilities. Including a projection step does not rectify the issue, as the expectation of  $Z_t$  remains different from  $Q_t$ .

## 4.2 Linear Function Approximation

In the linear approximation setting, we represent each state-action pair  $x, a$  as a feature vector  $\phi_{x,a} \in \mathbb{R}^d$ . We wish to find a linear function given by a weight vector  $\theta$  such that

$$\theta^T \phi_{x,a} \approx Q^\pi(x, a).$$

In the categorical distributional setting,  $\theta$  becomes a matrix  $W \in \mathbb{R}^{K \times d}$ . Here we will consider approximating the cumulative distribution function:

$$F_{Z(x,a)}(z_i) = W \phi_{x,a}[z_i].$$

We can extend this parametric form by viewing  $F$  as describing the CDF of a mixture of Diracs (Equation 4). Thus,  $F(z) = 0$  for  $z < z_1$ , and similarly  $F(z) = F(z_k)$  for  $z \geq z_k$ ; see (Bellemare et al. 2019) for a justification. In what follows we further assume that the support  $\mathbf{z}$  is 1-spaced.

In this setting, there may be no  $W$  for which  $F_{Z(x,a)}$  describes a proper cumulative distribution function: e.g.  $F(y)$  may be less than or greater than 1 for  $y > z_k$ . Yet, as shown by (Bellemare et al. 2019), we can still analyze the behaviour of a distributional algorithm which is allowed to output improper distributions. In our analysis we will assume that all predicted distributions sum to 1, though they may assign negative mass to some outcomes.

We write  $\mathcal{Q}_\phi := \{\theta^\top \phi : \theta \in \mathbb{R}^d\}$  for the set of value functions that can be represented by a linear approximation over  $\phi$ . Similarly,  $\mathcal{Z}_\phi := \{W\phi : W \in \mathbb{R}^{K \times d}\}$  is the set of CDFs that are linearly representable. For  $Z_t \in \mathcal{Z}_\phi$ , let  $W_t$  be the corresponding weight matrix. As before, we define  $Z'_t(x_t, a_t)$  to be the random variable corresponding to the projected target  $\Pi_C[r_t + \gamma Z_t(x_{t+1}, a_{t+1})]$ .

**Proposition 8.** *Let  $Z_0 \in \mathcal{Z}_\phi$ ,  $Q_0 \in \mathcal{Q}_\phi$ , and  $\mathbf{z} \in \mathbb{R}^K$  such that  $\mathbf{z}$  is 1-spaced. Suppose that  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$ , and that  $Z_0(x, a)[z_K] = 1 \forall x, a$ . Let  $W_t, \theta_t$  respectively denote the weights corresponding to  $Z_t$  and  $Q_t$ . If  $Z_{t+1}$  is computed from the semi-gradient update rule*

$$W_{t+1} := W_t + \alpha_t (F_{Z'_t}(x_t, a_t) - W_t \phi_{x_t, a_t}) \phi_{x_t, a_t}^T$$

and  $Q_{t+1}$  is computed according to Equation 2 with the same step-size  $\alpha_t$ , then also  $Z_t \stackrel{\mathbb{E}}{=} Q_t \forall t \in \mathbb{N}$ .

Importantly, the gradient in the previous proposition is taken with respect to the CDF of the distribution. Taking the gradient with respect to the Probability Mass Function (PMF) does not preserve the expected value of the predictions, which we have already shown in the tabular setting. This negative result is consistent with the results on signed distributions by (Bellemare et al. 2019).

## 4.3 Non-linear Function Approximation

To conclude this theoretical analysis, we consider more general function approximation schemes, which we will refer to as the non-linear setting. In the non-linear setting, we consider a differentiable function  $\psi(W, \cdot)$ . For example, the function  $\psi(W, \phi_{x,a})$  could be a probability mass function given by a softmax distribution over the logits  $W\phi_{x,a}$ , as is the case for the final layer of the neural network in the C51 algorithm.

**Proposition 9.** *There exists a (nonlinear) representation of the cumulative distribution function parametrized by  $W \in \mathbb{R}^{K \times d}$  such that  $Z_0 \stackrel{\mathbb{E}}{=} Q_0$  but after applying the semi-gradient update rule*

$$W_{t+1} := W_t + \alpha_t \nabla_W \ell_2^2(\psi(W, \phi(x_t, a_t)), F_{Z_t}),$$

where  $F_{Z_t}$  is the cumulative distribution function of the projected Bellman target, we have  $Z_1 \neq Q_1$ .

The key difference with Prop. 8 is the change from the gradient of a linear function (the feature vector  $\phi_{x_t, a_t}$ ) to the gradient of a nonlinear function; hence the result is not as trivial as it might look. Still, while the result is not altogether surprising, combined with our results in the linear case it shows that the interplay with nonlinear function approximation is a viable candidate for explaining the benefits of the distributional approach. In the next section we will present empirical evidence to this effect.

## 5 Empirical Analysis

Our theoretical results demonstrating that distributional RL often performs identically to expected RL contrast with the empirical results of (Bellemare, Dabney, and Munos 2017; Dabney et al. 2017; 2018; Barth-Maron et al. 2018; Hessel et al. 2018), to name a few. In this section we confirm our theoretical findings by providing empirical evidence that distributional reinforcement learning does not improve performance when combined with tabular representations or linear function approximation. Then, we find evidence of improved performance when combined with deep networks, suggesting that the answer lies, as suggested by Prop. 9, in distributional reinforcement learning’s interaction with nonlinear function approximation.

### 5.1 Tabular models

Though theoretical results indicate that performing gradient updates with respect to the distribution’s CDF should produce different predicted distributions from gradient updates with respect to its PMF, it is not immediately clear how these differences affect performance. To explore this, we considered a 12x12 gridworld environment and ran two distributional versions of Q-learning, one which performed gradient updates with respect to the CDF and one which performed updates with respect to the PMF of the predicted distribution, alongside traditional Q-learning. We found that, as predicted by Proposition 4, when given the same random seed the CDF update method had identical performance to traditional Q-learning. The PMF update method, though

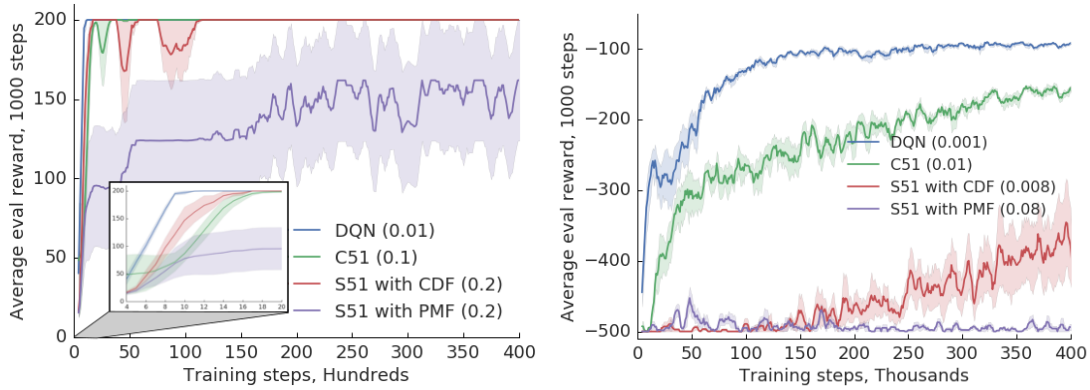


Figure 1: Cartpole (left) and Acrobot (right) with Fourier features of order 4. Step size in parentheses. Algorithms correspond to the lite versions described in text.

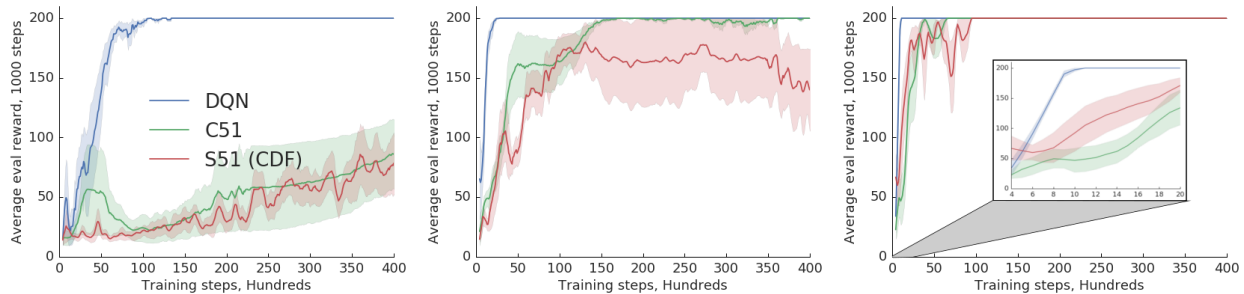


Figure 2: Varying the basis orders on CartPole. Orders 1 (left), 2 (center), 3 (right). Step sizes same as in Fig.1.

not significantly worse in performance, did exhibit *different* performance, performing better on some random seeds and worse on others than expected Q-learning. We observed a similar phenomenon with a simple 3-state chain MDP. Results for both experiments are omitted for brevity, but are included in the supplemental material.

## 5.2 Linear Function Approximation

We next investigate whether the improved performance of distributional RL is due to the outputs and loss functions used, and whether this can be observed even with linear function approximation.

We make use of three variants of established algorithms for our investigation, modified to use linear function approximators rather than deep networks. We include in this analysis an algorithm that computes a softmax over logits that are a linear function of a state feature vector. Although we did not include this type of approximator in the linear setting of our theoretical analysis, we include it here as it provides an analogue of C51 against which we can compare the other algorithms.

1. **DQN-lite**, based on (Mnih et al. 2015), predicts  $Q(x, a)$ , the loss is the squared difference between the target and the prediction.
2. **C51-lite**, based on (Bellemare, Dabney, and Munos 2017), outputs  $Z(x, a)$ , a softmax distribution whose logits are

linear in  $\phi_{x,a}$ . It minimizes the cross-entropy loss between the target and the prediction.

3. **S51-lite**, based on (Bellemare et al. 2019), outputs  $Z(x, a)$  as a categorical distribution whose probabilities are a linear function of  $\phi_{x,a}$  and minimizes the squared Cramér distance.

We further break S51-lite down into two sub-methods. One of these performs updates by taking the gradient of the Cramér distance with respect to the points of the PMF of the prediction, while the other takes the gradient with respect to the CDF. For a fair comparison, all algorithms used a stochastic gradient descent optimizer except where noted. We used the same hyperparameters for all algorithms, except for step sizes, where we chose the step size that gave the best performance for each algorithm. We otherwise use the usual agent infrastructure from DQN, including a replay memory of capacity 50,000 and a target network which is updated after every 10 training steps. We update the agent by sampling batches of 128 transitions from the replay memory.

We ran these algorithms on two classic control environments: CartPole and Acrobot. In CartPole, the objective is to keep a pole balanced on a cart by moving the cart left and right. In Acrobot, the objective is to swing a double-linked pendulum above a threshold by applying torque to one of its joints. We encode each original state  $x$ , ( $x \in \mathbb{R}^4$  for CartPole and  $x \in \mathbb{R}^6$  for Acrobot) as a feature vector  $\phi(x)$  given by the Fourier basis for some fixed order (Konidaris, Osentoski,

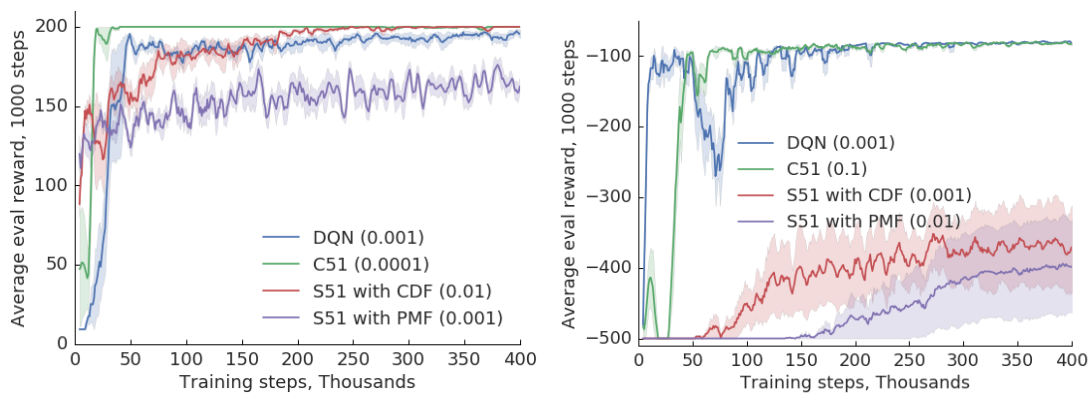


Figure 3: Cartpole with Adam optimizer (left) and Acrobot (right) with deep networks. Learning rate parameter in parentheses.

and Thomas 2011). For completeness, on Cartpole the basis of order 1 yields 15 features, order 2: 80 features, 3: 255, and finally 4: 624 features.

We first compare how DQN-, C51-, and S51-lite perform on the two tasks in Figure 1 with the order 4 basis, which is more than sufficient to well-approximate the value function. We observe that DQN learns more quickly than C51 and S51 with CDF, while S51 with PMF underperforms significantly. On Acrobot, the difference is even more pronounced. This result at first glance seems to contradict the theoretical result we showed indicating that S51-lite should perform similarly to DQN in the linear function approximation case, but we attribute this difference in performance to the fact that the initialization in the S51 algorithm doesn't enforce the assumption that the predicted distributions sum to 1.

We then investigate the effect of reducing the order in all algorithms in Figure 2. We observe that the distributional algorithms performs poorly when there are too few features; by contrast, DQN-lite can perform both tasks with an order 2 basis. This indicates that the distributional methods suffered more when there were fewer informative features available than expectation-based methods did in this setting.

### 5.3 Nonlinear function approximation

We repeat the above experiment, but now replace the Fourier basis features with a two-layer ReLU neural network that is trained along with the final layer (which remains linear). In the CartPole task we found that DQN often diverged with the gradient descent optimizer, so we used Adam for all the algorithms, and chose the learning rate parameter that gave the best performance for each. Results are displayed in Figure 3. We can observe that C51 generally outperforms DQN, although they both eventually converge to the same value. It is interesting to notice that S51 has a harder time achieving the same performance, and comes nowhere near in the harder Acrobot task. This suggests that despite being theoretically unnecessary, the softmax in C51 is working to its advantage. This finding is consistent with the empirical results observed in Atari games by (Bellemare et al. 2019).

The results of the previous two sets of experiments indicate that the benefits of distributional reinforcement learning are

primarily gained from improved learning in the earlier layers of deep neural networks, as well as in the nonlinear softmax used in C51. We believe further investigations in this direction should lead to a deeper understanding of the distributional approach.

## 6 Discussion and Future Work

In this paper we have provided theoretical and empirical results that give evidence on the benefits (or, some cases, lack thereof) of the distributional approach in reinforcement learning. Together, our results point to function approximation as the key driver in the difference in behaviour between distributional and expected algorithms.

To summarize, our findings are:

1. Distributional methods are generally expectation-equivalent when using tabular representations or linear function approximation, but
2. diverge from expected methods when we use non-linear function approximation.
3. Empirically, we provide fresh confirmation that modelling a distribution helps when using non-linear approximation.

There are a few notions from distributional reinforcement learning not covered by our study here, including the effect of using Wasserstein projections of the kind implied by quantile regression (Dabney et al. 2017), and the impact of the softmax transfer function used by C51 on learning. In particular the regression setting, (Imani and White 2018) show that even for a fixed set of features, optimizing a distributional loss results in better accuracy than minimizing the squared error of predictions.

Yet we believe the most important question raised by our work is: what happens in deep neural networks that benefits most from the distributional perspective? Returning to the proposed reasons for the distributional perspective's success in the introduction, we note that the potentially regularizing effect of modeling a distribution, and a potential role as an 'auxiliary task' played by the distribution are both avenues that remain largely unaddressed by this work.

## References

- Arjovsky, M.; Chintala, S.; and Bottou, L. 2017. Wasserstein generative adversarial networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 214–223. International Convention Centre, Sydney, Australia: PMLR.
- Barth-Maron, G.; Hoffman, M. W.; Budden, D.; Dabney, W.; Horgan, D.; TB, D.; Muldal, A.; Heess, N.; and Lillicrap, T. 2018. Distributional policy gradients. In *Proceedings of the International Conference on Learning Representations, to appear*.
- Bellemare, M. G.; Danihelka, I.; Dabney, W.; Mohamed, S.; Lakshminarayanan, B.; Hoyer, S.; and Munos, R. 2017. The cramer distance as a solution to biased wasserstein gradients. *CoRR* abs/1705.10743.
- Bellemare, M. G.; Roux, N. L.; Castro, P. S.; and Moitra, S. 2019. Distributional reinforcement learning with linear function approximation. *To appear in Proceedings of AISTATS*.
- Bellemare, M. G.; Dabney, W.; and Munos, R. 2017. A distributional perspective on reinforcement learning. In Precup, D., and Teh, Y. W., eds., *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, 449–458. International Convention Centre, Sydney, Australia: PMLR.
- Bellman. 1957. Dynamic programming.
- Bertsekas, I. 1996. Temporal differences-based policy iteration and applications in neuro-dynamic programming.
- Dabney, W.; Rowland, M.; Bellemare, M. G.; and Munos, R. 2017. Distributional reinforcement learning with quantile regression. *CoRR* abs/1710.10044.
- Dabney, W.; Ostrovski, G.; Silver, D.; and Munos, R. 2018. Implicit quantile networks for distributional reinforcement learning. 80:1096–1105.
- Ferns, N.; Castro, P. S.; Precup, D.; and Panangaden, P. 2012. Methods for computing state similarity in markov decision processes. *CoRR* abs/1206.6836.
- Gibbs, A., and Su, F. 2002. On Choosing and Bounding Probability Metrics. *International Statistical Review* 30:419–435.
- Hessel, M.; Modayil, J.; van Hasselt, H.; Schaul, T.; Ostrovski, G.; Dabney, W.; Horgan, D.; Piot, B.; Azar, M.; and Silver, D. 2018. Rainbow: Combining improvements in deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*.
- Imani, E., and White, M. 2018. Improving regression performance with distributional losses. 80.
- Konidaris, G.; Osentoski, S.; and Thomas, P. S. 2011. Value function approximation in reinforcement learning using the fourier basis. In *Proceedings of the AAAI Conference*.
- Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M.; Fidjeland, A. K.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.
- Rowland, M.; Bellemare, M.; Dabney, W.; Munos, R.; and Teh, Y. W. 2018. An analysis of categorical distributional reinforcement learning. In Storkey, A., and Perez-Cruz, F., eds., *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, volume 84 of *Proceedings of Machine Learning Research*, 29–37. Playa Blanca, Lanzarote, Canary Islands: PMLR.
- Rummery, G. A., and Niranjan, M. 1994. On-line Q-learning using connectionist systems. Technical report, Cambridge University Engineering Department.
- Sutton, R. S., and Barto, A. G. 1998. *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1st edition.
- Székely, G. 2003. E-statistics: The energy of statistical samples. *Bowling Green State University, Department of Mathematics and Statistics Technical Report* 3(05):1–18.
- Thorisson, H. 2000. *Coupling, Stationarity, and Regeneration*. Probability and Its Applications. Springer New York.