

Title

# Acknowledgements

# Abstract

Temporal difference learning algorithms classically learn value functions (sum of discounted rewards) by solving for the fixed point that relates the value of one state with the expected value of the following states. Other TD learning algorithms have been used to learn *multiplicative value functions*, which also have a fixed point solution. An example of such a function is the ratio between stationary distributions of two policies. It has been shown that learning of these multiplicative value functions suffers from higher variance and looser convergence guarantees. One potential approach to solving this issues would be to learn in log space, turning the multiplicative value function into an additive one. Unfortunately, it is easy to show with Jensen’s inequality that the exponential of the log fixed point does not correspond to the multiplicative fixed point.

Distributional RL has been used to learn the return distribution (return is the sum of future discounted rewards). Although it has been shown that it mainly plays an auxiliary task role for representation learning, we propose that distributional fixed points could play a much more fundamental role to learning non additive value functions. In the case of multiplicative value functions, learning the additive fixed point in log space and exponentiating leads to the correct solution.

In particular, we will study this approach in the case of the learning process of the Covariate Shift ratio, which defines a multiplicative value function.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Distributional Reinforcement Learning</b>	<b>2</b>
2.1	Reinforcement Learning Setting . . . . .	4
2.2	Towards a Distributional Reinforcement Learning . . . . .	5
2.3	Approximation Framework in the Distributional Setting . . . . .	8
2.4	Categorical Distributional Reinforcement Learning . . . . .	8
2.5	Distributional RL with Linear Function Approximation . . . . .	10
<b>3</b>	<b>Covariate Shift Ratio</b>	<b>11</b>
3.1	COP-TD . . . . .	13
<b>4</b>	<b>Distributional COP-TD</b>	<b>16</b>
<b>5</b>	<b>Logarithmic Approach to Distributional COP-TD</b>	<b>18</b>
<b>6</b>	<b>Implementation</b>	<b>19</b>
6.1	Categorical COP-TD . . . . .	19
6.2	Exponential Bins . . . . .	20
6.3	Log-Categorical COP-TD . . . . .	20
<b>7</b>	<b>Evaluation of the proposal</b>	<b>21</b>
<b>8</b>	<b>Conclusions</b>	<b>22</b>
	<b>Appendix</b>	<b>23</b>
A.1	Useful Distributional Notation . . . . .	23
A.2	Metrics over Distributions . . . . .	24
A.3	Contraction Mappings . . . . .	24
A.4	Implementation Details . . . . .	24
	<b>Referencias</b>	<b>25</b>

# 1 Introduction

- Reinforcement Learning
  - Distributional Reinforcement Learning
  - Off-Policy Learning
  - Covariate Shift Ratio

## Motivation and Goals

- Relevance of logarithmic Distributional RL
  - Design of Distributional Covariate Shift Ratio
  - Evaluation of the logarithmic approach within the Distributional CSR setting

## Structure of the Report

## 2 Distributional Reinforcement Learning

TODO: INTRO

Initial paper[1]: In this paper we argue for the fundamental importance of the value distribution: the distribution of the random return received by a reinforcement learning agent. This is in contrast to the common approach to reinforcement learning which models the expectation of this return, or value. Although there is an established body of literature studying the value distribution, thus far it has always been used for a specific purpose such as implementing risk-aware behaviour. We begin with theoretical results in both the policy evaluation and control settings, exposing a significant distributional instability in the latter. We then use the distributional perspective to design a new algorithm which applies Bellman’s equation to the learning of approximate value distributions. We evaluate our algorithm using the suite of games from the Arcade Learning Environment. We obtain both state-of-the-art results and anecdotal evidence demonstrating the importance of the value distribution in approximate reinforcement learning. Finally, we combine theoretical and empirical evidence to highlight the ways in which the value distribution impacts learning in the approximate setting.

Although the distributional perspective is almost as old as Bellman’s equation itself (Jaquette, 1973; Sobel, 1982; White, 1988), in reinforcement learning it has thus far been subordinated to specific purposes: to model parametric uncertainty (Dearden et al., 1998), to design risk-sensitive algorithms (Morimura et al., 2010b;a), or for theoretical analysis (Azar et al., 2012; Lattimore Hutter, 2012). By contrast, we believe the value distribution has a central role to play in reinforcement learning.

Contraction of the policy evaluation Bellman operator. Basing ourselves on results by Ross (1992) we show that, for a fixed policy, the Bellman operator over value distributions is a contraction in a maximal form of the Wasserstein (also called Kantorovich or Mallows) metric. Our particular choice of metric matters: the same operator is not a contraction in total variation, Kullback-Leibler divergence, or Kolmogorov distance.

Instability in the control setting. We will demonstrate an instability in the distributional version of Bellman’s optimality equation, in contrast to the policy evaluation case. Specifically, although the optimality operator is a contraction in expected value (matching the usual optimality result), it is not a contraction in any metric over distributions. These results provide evidence in favour of learning algorithms that model the effects of nonstationary policies.

Better approximations. From an algorithmic standpoint, there are many benefits to learning an approximate distribution rather than its approximate expectation. The distributional Bellman operator preserves multimodality in value distributions, which we believe leads to more stable learning. Approximating the full distribution also mitigates the effects of learning from a nonstationary policy. As a whole, we argue that this approach makes approximate reinforcement learning significantly better behaved.

From a supervised learning perspective, learning the full value distribution might seem obvious: why restrict ourselves to the mean? The main distinction, of course, is that in our setting there are no given targets. Instead, we use Bellman’s equation to make the learning process tractable; we must, as Sutton and Barto (1998) put it, “learn a guess from a guess”. It is our belief that this guesswork ultimately carries more benefits than costs.

WASSERSTEIN METRIC! THEORY DEVELOPED + The Cramer Distance as a Solution to Biased Wasserstein Gradients

## INTERESTING APPENDIX ON RELATED WORK!!

It is surprising that, when we use a policy which aims to maximize expected return, we should see any difference in performance. The distinction we wish to make is that learning distributions matters in the presence of approximation. We now outline some possible reasons.

Reduced chattering. Our results from Section 3.4 highlighted a significant instability in the Bellman optimality operator. When combined with function approximation, this instability may prevent the policy from converging, what Gordon (1995) called chattering. We believe the gradient-based categorical algorithm is able to mitigate these effects by effectively averaging the different distributions, similar to conservative policy iteration (Kakade AND Langford, 2002). While the chattering persists, it is integrated to the approximate solution.

State aliasing. Even in a deterministic environment, state aliasing may result in effective stochasticity. McCallum (1995), for example, showed the importance of coupling representation learning with policy learning in partially observable domains. We saw an example of state aliasing in PONG, where the agent could not exactly predict the reward timing. Again, by explicitly modelling the resulting distribution we provide a more stable learning target.

A richer set of predictions. A recurring theme in artificial intelligence is the idea of an agent learning from a multitude of predictions (Caruana 1997; Utgoff and Stracuzzi 2002; Sutton et al. 2011; Jaderberg et al. 2017). The distributional approach naturally provides us with a rich set of auxiliary predictions, namely: the probability that the return will take on a particular value. Unlike previously proposed approaches, however, the accuracy of these predictions is tightly coupled with the agent's performance.

Framework for inductive bias. The distributional perspective on reinforcement learning allows a more natural framework within which we can impose assumptions about the domain or the learning problem itself. In this work we used distributions with support bounded in  $[V_{\min}, V_{\max}]$ . Treating this support as a hyperparameter allows us to change the optimization problem by treating all extremal returns (e.g. greater than  $V_{\max}$ ) as equivalent. Surprisingly, a similar value clipping in DQN significantly degrades performance in most games. To take another example: interpreting the discount factor as a proper probability, as some authors have argued, leads to a different algorithm.

Well-behaved optimization. It is well-accepted that the KL divergence between categorical distributions is a reasonably easy loss to minimize. This may explain some of our empirical performance. Yet early experiments with alternative losses, such as KL divergence between continuous densities, were not fruitful, in part because the KL divergence is insensitive to the values of its outcomes. A closer minimization of the Wasserstein metric should yield even better results than what we presented here.

An Analysis of Categorical Distributional Reinforcement Learning

Distributional Reinforcement Learning with Quantile Regression

Implicit Quantile Networks for Distributional Reinforcement Learning

DISTRIBUTED DISTRIBUTIONAL DETERMINISTIC POLICY GRADIENTS

A Comparative Analysis of Expected and Distributional Reinforcement Learning

Statistics and Samples in Distributional Reinforcement Learning:

## 2.1 Reinforcement Learning Setting

We consider an agent interacting with an environment in the standard setting[8]: at each step  $t$ , the agent selects an action  $a_t$  based on its current state  $s_t$ , to which the environment responds with a reward  $r_t$  and then moves to the next state  $s_{t+1}$ . We model this interaction as a time-homogeneous Markov Decision Process  $(\mathcal{S}, \mathcal{A}, r, P, \gamma)$ , where

- $\mathcal{S}$  and  $\mathcal{A}$  are the state and action spaces, respectively, we assume that both are finite, with  $n := |\mathcal{S}|$ ;
- $P$  is the transition kernel,  $s_{t+1} \sim P(\cdot | s_t, a_t)$ ; the Markov assumption states that  $P(s_{t+1} | s_t, a_t, s_{t-1}, a_{t-1}, \dots) = P(s_{t+1} | s_t, a_t)$ ;
- $r(s, a)$  represents the immediate reward given by the environment after taking action  $a$  being in state  $s$ . These rewards are considered to be sampled from the reward function  $R(s, a)$ , i.e.  $r_t \sim R(s_t, a_t)$ ;
- $\gamma$  is the discount factor

A policy  $\pi$  maps each state to a probability distribution over the action space,  $a_t \sim \pi(\cdot | s_t)$ . In addition, we combine the policy  $\pi$  and transition function  $P$  into a state-to-state transition function  $P_\pi \in \mathbb{R}^{n \times n}$ , whose entries are

$$P_\pi(s' | s) := \text{Prob}_\pi(s_{t+1} = s' | s_t = s) = \sum_{a \in \mathcal{A}} \pi(a | s) P(s' | s, a) \quad (2.1)$$

In particular, powers of  $P_\pi$  represent the transition function across different time-steps.

Given  $\pi$ , the action value function is defined as the expected sum of discounted rewards from a state-action pair by following the policy:

$$Q^\pi(s, a) = \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 = s, a_0 = a \right] \quad (2.2)$$

The Bellman's equation can be obtained from this expression:

$$\begin{aligned} Q^\pi(s, a) &= \mathbb{E} [r(s, a)] + \gamma \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_{t+1}, a_{t+1}) \middle| s_0 = s, a_0 = a \right] \\ &= \mathbb{E} [r(s, a)] + \gamma \sum_{s'} P(s' | s, a) \left( \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_{t+1}, a_{t+1}) \middle| s_1 = s' \right] \right) \\ &= \mathbb{E} [r(s, a)] + \\ &\quad \gamma \sum_{s'} P(s' | s, a) \sum_{a'} \pi(a' | s') \left( \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_{t+1}, a_{t+1}) \middle| s_1 = s', a_1 = a' \right] \right) \\ &= \mathbb{E} [r(s, a)] + \gamma \mathbb{E}_{s' \sim P(\cdot | s, a), a' \sim \pi(\cdot | s')} [Q(s', a')] \end{aligned} \quad (2.3)$$



Analogously for the state value function (considering  $r_\pi(s) := \mathbb{E}_{a \sim \pi(\cdot|s)} [r(s, a)]$ ):

$$\begin{aligned} V^\pi(s) &= \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 = s \right] \\ &= r_\pi(s) + \gamma \sum_a \pi(a|s) \sum_{s'} P(s'|s, a) \left( \mathbb{E}_\pi \left[ \sum_{t=0}^{\infty} \gamma^t r(s_{t+1}, a_{t+1}) \middle| s_1 = s' \right] \right) \\ &= r_\pi(s) + \gamma \mathbb{E}_{s' \sim P_\pi(\cdot|s)} \left[ V^\pi(s') \right] \end{aligned} \quad (2.4)$$

In the context of *policy evaluation*, and considering state-action value functions  $Q^\pi$  as vectors in  $\mathbb{R}^{\mathcal{S} \times \mathcal{A}}$ , the *Bellman operator*  $\mathcal{T}^\pi : \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \rightarrow \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$  is defined as

$$\mathcal{T}^\pi Q(s, a) := \mathbb{E} [r(s, a)] + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a), a' \sim \pi(\cdot|s')} \left[ Q(s', a') \right] \quad (2.5)$$

Such operator is useful to describe the expected behaviour of popular learning algorithms (e.g. Q-learning) and satisfies some interesting and desirable properties[8]:

- It is a contraction mapping (DEFINE IT, OR APPENDIX)
- The process  $Q_t = \mathcal{T}^\pi Q_{t-1}$ , for some initial value  $Q_0$ , converges exponentially to  $Q^\pi$  as  $t \rightarrow \infty$ .

Regarding the *control setting*, where the goal is to improve the current policy  $\pi$ , a *Bellman optimality operator*  $\mathcal{T}^* : \mathbb{R}^{\mathcal{S} \times \mathcal{A}} \rightarrow \mathbb{R}^{\mathcal{S} \times \mathcal{A}}$  is defined as follows:

$$\mathcal{T}^* Q(s, a) := \mathbb{E} [r(s, a)] + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a)} \left[ \max_{a' \in \mathcal{A}} Q(s', a') \right] \quad (2.6)$$

As it was shown in [8],  $\mathcal{T}^*$  is also a contraction mapping, and the expectation of  $\{Q_t\}$ ,  $Q_t = \mathcal{T}^* Q_{t-1}$ , converges exponentially to a fixed point (in this case, the optimal value function  $Q^*$ ).

## 2.2 Towards a Distributional Reinforcement Learning

TODO: INTRO

In [1] a distributional Bellman equation is defined:

$$Z^\pi(s, a) \stackrel{D}{=} R(s, a) + \gamma Z^\pi(S', A') \quad (2.7)$$

where

- $Z^\pi : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{Z}$  is the *value distribution*, a mapping from state-action pairs to distributions over returns by following policy  $\pi$ . Its expectation is the value  $Q^\pi$

$$Q^\pi(s, a) := \mathbb{E}[Z^\pi(s, a)] \quad (2.8)$$

We will also call it the *random return*.

- $(S', A') \in \mathcal{S} \times \mathcal{A}$  is the next state-action random variable:  $S' \sim P(\cdot|s, a)$ ,  $A' \sim \pi(\cdot|S')$

- $R(s, a) \in \mathcal{Z}$  is the random reward, or equivalently the reward function. Note that now we are dealing with it as an explicit random variable.
- $Z^\pi(S', A') \in \mathcal{Z}$  is the random return over the random next state-action following  $\pi$ . This notation implies that all possible next state-action pairs need to be considered as to generate this return distribution. Thus,  $Z^\pi(S', A')$  may be seen as a mixture distribution of the distributions  $Z^\pi(s', a')$  where  $s'$  and  $a'$  are sampled from  $(S', A')$ :

$$f_{Z^\pi(S', A')}(z) = \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') f_{Z^\pi(s', a')}(z) \quad (2.9)$$

The expected value, using 2.8, can be then expressed as:

$$\begin{aligned} \mathbb{E}[Z^\pi(S', A')] &= \int_{-\infty}^{\infty} z \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') f_{Z^\pi(s', a')}(z) dz \\ &= \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') \int_{-\infty}^{\infty} z f_{Z^\pi(s', a')}(z) dz \\ &= \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') \mathbb{E}[Z^\pi(s', a')] \\ &= \mathbb{E}_{s' \sim P(\cdot|s, a), a' \sim \pi(\cdot|s')} \left[ Q^\pi(s', a') \right] \end{aligned} \quad (2.10)$$

**Remark.** A distributional equation  $U \stackrel{D}{=} V$  indicates that the random variable  $U$  is distributed according to the same law as  $V$ .

Note that we can easily recover the classical Bellman's equation 2.3 for the action value  $Q$  by using 2.8 and 2.10 when taking the expected value over its distributional version 2.7:

$$\begin{aligned} Q^\pi(s, a) &= \mathbb{E}[Z^\pi(s, a)] \\ &= \mathbb{E}[R(s, a)] + \gamma \mathbb{E}[Z^\pi(S', A')] \\ &= \mathbb{E}[r(s, a)] + \gamma \mathbb{E}_{s' \sim P(\cdot|s, a), a' \sim \pi(\cdot|s')} \left[ Q^\pi(s', a') \right] \end{aligned} \quad (2.11)$$

Finally, let's try to find out what actually the random return  $Z$  represents by expanding its density function:

$$\begin{aligned} f_{Z^\pi(s, a)}(z) &= f_{R(s, a) + \gamma Z^\pi(S', A')}(z) \\ &= \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') f_{R(s, a) + \gamma Z^\pi(s', a')}(z) \\ &= \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') f_{R(s, a) + \gamma(R(s', a') + \gamma Z^\pi(S'', A''))}(z) \\ &= \sum_{s'} P(s'|s, a) \sum_{a'} \pi(a'|s') \sum_{s''} P(s''|s', a') \sum_{a''} \pi(a''|s'') \\ &\quad f_{R(s, a) + \gamma R(s', a') + \gamma^2 Z^\pi(s'', a'')}(z) \\ &= \sum_{s_1} P(s_1|s_0, a_0) \sum_{a_1} \pi(a_1|s_1) \cdots \sum_{s_t} P(s_t|s_{t-1}, a_{t-1}) \sum_{a_t} \pi(a_t|s_t) \\ &\quad f_{\sum_{i=0}^t \gamma^i R(s_i, a_i)}(z) \end{aligned} \quad (2.12)$$

where we have repeatedly used property 2 of mixture distributions. In addition, note that assuming independence between the random reward  $R$  of a certain state-action pair and the

return distribution  $Z^\pi$  of the possible next state-action (which is NOT TRUE in general), we can rewrite it in terms of convolutions as

$$f_{Z^\pi(s,a)}(z) = \sum_{s_1} P(s_1|s_0, a_0) \sum_{a_1} \pi(a_1|s_1) \cdots \sum_{s_t} P(s_t|s_{t-1}, a_{t-1}) \sum_{a_t} \pi(a_t|s_t) (f_{R(s_0,a_0)} * f_{\gamma R(s_1,a_1)} * \cdots * f_{\gamma^t R(s_t,a_t)})(z) \quad (2.13)$$

According to 2.12,  $Z^\pi(s, a)$  can be interpreted as a convex combination of the sum of discounted reward distributions of all possible agent trajectories starting at the state-action  $(a, s)$  and following policy  $\pi$  from then on, each weight corresponding to the probability of that precise trajectory. It is important to note that  $Z^\pi$  encodes the intrinsic randomness of the agent's interactions with its environment; we should avoid considering it as a measure of uncertainty about the environment itself.

Moving to the *policy evaluation* setting, now we are interested in studying the behaviour of a distributional version of the policy evaluation operator  $\mathcal{T}^\pi$ . First we define the transition operator  $P^\pi : \mathcal{Z} \rightarrow \mathcal{Z}$

$$P^\pi Z(s, a) \stackrel{D}{=} Z(S', A') \quad (2.14)$$

Given this, the *distributional Bellman operator*  $\mathcal{T} : \mathcal{Z} \rightarrow \mathcal{Z}$  is defined as

$$\mathcal{T}^\pi Z(s, a) \stackrel{D}{=} R(s, a) + \gamma P^\pi Z(s, a) \quad (2.15)$$

We emphasize that three sources of randomness are involved in the compound distribution  $\mathcal{T}^\pi Z$ , i.e.

1. Randomness in the reward  $R$ ,
2. Randomness in the transition  $P^\pi$ , and
3. Randomness in the next state-value distribution  $Z(S', A')$ ,

which together make this distributional Bellman operator fundamentally different to the expected value-cased one (Equation 2.5). Authors in [1] demonstrate, under the assumption that these three random quantities are independent, that  $\mathcal{T}^\pi$  is a contraction mapping whose unique fixed point is the value distribution  $\mathcal{Z}^\pi$ .

However, more difficulties arise in the *control setting* when dealing with this distributional perspective; as stated in [1], while every optimal policy attain the same value  $Q^*$  in the expected-valued case, there might be many optimal value distributions. Considering the set  $\Pi^*$  of optimal policies, an optimal value distribution is precisely characterized as follows

**Definition 1.** An optimal value distribution is the value distribution of an optimal policy. The set of optimal value distributions is  $\mathcal{Z}^* := \{Z^{\pi^*} : \pi^* \in \Pi^*\}$ .

Note that an optimal value distribution must match the full distribution of returns under some optimal policy, so that not all value distributions with expectation  $Q^*$  are optimal. A *distributional Bellman optimality operator*  $\mathcal{T}$  is defined as well, but it doesn't behave as well as the policy evaluation operators: it is not a contraction in any usual metric between distributions, and its convergence to the set of optimal value distributions is weak.

CHANGE: In the following two sections, we focus our analysis of Distributional Reinforcement Learning when working with Tabular Representations and Linear Function Approximators, respectively; this will allow us to study in more detail the cases where good behaviour and convergence are guaranteed.

## 2.3 Approximation Framework in the Distributional Setting

We should be aware that the full computation of the distributional Bellman operators on return distribution functions is generally either impossible (as we typically do not have access to the MDP dynamics, but to merely sample transitions) or infeasible (since the value distribution cannot be stored exactly in the general case).

This leads us to the design of several key approximations which are required to implement practical and scalable distributional RL algorithms[12]:

### Distribution Parametrisation

Due to the fact that the full space of probability distributions,  $\mathcal{P}(\mathbb{R})$ , cannot be algorithmically encoded with a finite number of parameters, we need to approximate the distribution throughout a parametric family  $\mathcal{P} \in \mathcal{P}(\mathbb{R})$ .

### Stochastic Bellman Operators

In order to evaluate the distributional Bellman operator  $\mathcal{T}^\pi$ , all possible next state-action-reward combinations should be taken into account. As in the expected valued case, the usual way of overcoming this practical limitation is by learning through transition samples  $(s, a, r, s', a')$  of the MDP. Hence, we can define a *stochastic distributional Bellman operator*  $\hat{\mathcal{T}}^\pi$  adapted to the randomness of these transitions, which defines a random measure whose behaviour is equal in expectation to the true Bellman operator  $\mathcal{T}^\pi$ .

### Projection of Bellman Target Distribution

Another problem usually arises after computing the stochastic operator  $\hat{\mathcal{T}}^\pi$  over a value distribution  $Z(s, a)$ : the new distribution may no longer lie in the selected parametric family  $\mathcal{P}$ . If this is the case, we further need to apply a *projection operator*  $\Pi : \mathcal{P}(\mathbb{R}) \rightarrow \mathcal{P}$  so as to map  $\hat{\mathcal{T}}^\pi Z(s, a)$  into the proper parametric family.

### Gradient Updates

Finally, having computed a stochastic approximation  $\hat{Z}_k(s_k, a_k) = \Pi \hat{\mathcal{T}}^\pi Z_k(s_k, a_k)$  to the full target distribution, we still have to define how to compute the next iterate  $Z_{k+1}$ . For that, the use of gradient updates seems to be appropriate[8], as it helps dissipate some noise introduced in the target by the stochastic approximation.

## 2.4 Categorical Distributional Reinforcement Learning

Although the distributional perspective is almost as old as Bellman's equations[13], it was not until the recent introduction of the Categorical Distributional Reinforcement[1] (CDRL) that it has become a central role within reinforcement learning. Their algorithm, called C51, was able to obtain state-of-the-art results in the Arcade Learning Environment[14] (ALE), outperforming the top expected-valued solutions by then.

The name 'categorical' comes from the distribution parametrisation that is used, which consists in the parametric family of categorical distributions over some fixed set of equally-spaced supports  $z_1 < \dots < z_K$ :

$$\mathcal{P} = \left\{ \sum_{i=1}^K p_i \delta_{z_i} \mid p_1, \dots, p_K \geq 0, \sum_{k=1}^K p_k = 1 \right\} \quad (2.16)$$

Apart from that, the algorithm can be fully characterized by the rest of approximations implemented within the distributional RL framework described in the previous subsection:

- So as to learn from sampling transitions, we move to the *stochastic distributional Bellman operator*  $\hat{\mathcal{T}}^\pi$  and the *stochastic distributional Bellman optimality operator*  $\hat{\mathcal{T}}^*$ . Given a sampled transition  $(s_t, a_t, r, s_{t+1}, a_{t+1})$ , these stochastic operators basically transform the supports of the distributions by an affine shift map  $f_{r,\gamma} : \mathbb{R} \rightarrow \mathbb{R}$ , defined by  $f_{r,\gamma}(z) = r + \gamma z$ ; in our notation,

$$\hat{\mathcal{T}}^\pi Z_t(s_t, a_t) = (f_{r,\gamma})_\# Z_t(s_{t+1}, a_{t+1})$$

with  $a_{t+1}$  selected by sampling the policy  $\pi(\cdot | s_{t+1})$  (categorical policy evaluation), and

$$\hat{\mathcal{T}}^* Z_t(s_t, a_t) = (f_{r,\gamma})_\# Z_t(s_{t+1}, a_{t+1})$$

with  $a_{t+1}$  being the action with the highest estimated expected returns (categorical Q-learning).

- CDRL applies the heuristic projection operator  $\Pi_C$  after computing the stochastic Bellman operators in order to recover a distribution within the selected parametric family  $\mathcal{P}$ . This projection is defined for single Dirac measures as

$$\Pi_C(\delta_y) = \begin{cases} \delta_{z_1} & y \leq z_1 \\ \frac{z_{i+1}-y}{z_{i+1}-z_i} \delta_{z_i} + \frac{y-z_i}{z_{i+1}-z_i} \delta_{z_{i+1}} & z_i < y < z_{i+1} \\ \delta_{z_K} & y > z_K \end{cases} \quad (2.17)$$

and can be easily extended to finite mixtures of Dirac measures:

$$\Pi_C \left( \sum_{i=1}^N p_i \delta_{y_i} \right) = \sum_{i=1}^N p_i \Pi_C(\delta_{y_i})$$

We will also denote by  $\Pi_C$  the operator  $\mathcal{P}(\mathbb{R})^{\mathcal{S} \times \mathcal{A}} \rightarrow \mathcal{P}^{\mathcal{S} \times \mathcal{A}}$  that applies this projection to each distribution in the return distribution function.

- Finally, C51 performs a single step of gradient descent on the Kullback-Leibler divergence (see Appendix A.2 for the definition of this metric) of the prediction  $Z_t(s_t, a_t)$  from the target  $\hat{Z}_t(s_t, a_t)$ :

$$\text{KL} \left( \hat{Z}_t(s_t, a_t) \parallel Z_t(s_t, a_t) \right)$$

with respect to the parameters of  $Z_t(s_t, a_t)$ ; this gradient is used to generate the new estimate  $Z_{t+1}(s_t, a_t) = \sum_{k=1}^K p_{t+1,k}(s_t, a_t) \delta_{z_k}$ .

Pseudo-Algorithm 1 (provided by [12]) synthesises the steps performed by CDRL.

---

**Algorithm 1:** Categorical Distributional Reinforcement Learning
 

---

**Require:**  $Z_t(s, a) = \sum_{k=1}^K p_{t,k}(s, a) \delta_{z_k}$  for each  $(s, a)$   
**Input:** A sample transition  $(s_t, a_t, r_t, s_{t+1})$   
 #Compute distributional Bellman target  
**if** Categorical Policy Evaluation **then**  
    $a^* \sim \pi(\cdot | s_{t+1})$   
**else if** Categorical Q-learning **then**  
    $a^* \leftarrow \arg \max_a Q(s_{t+1}, a)$   
**end if**  
 $\hat{Z}_*(s_t, a_t) \leftarrow (f_{r_t, \gamma})_{\#} Z_t(s_{t+1}, a^*)$   
 #Project target onto support  
 $\hat{Z}_t(s_t, a_t) \leftarrow \Pi_C \hat{Z}_*(s_t, a_t)$   
 #Compute KL loss  
 Find gradient  $\text{KL}(\hat{Z}_t(s_t, a_t) || Z_t(s_t, a_t))$   
 Generate new estimate  $Z_{t+1}(s_t, a_t) = \sum_{k=1}^K p_{t+1,k}(s_t, a_t) \delta_{z_k}$   
**Output:** Estimate  $Z_{t+1}(s, a)$  for each  $(s, a)$

---

TABULAR REPRESENTATIONS, CONVERGENCE! (Analysis of C51)

## 2.5 Distributional RL with Linear Function Approximation

### 3 Covariate Shift Ratio

TODO: INTRO off-policy learning and COPTD

Original paper [2]:

The problem of on-line off-policy evaluation (OPE) has been actively studied in the last decade due to its importance both as a stand-alone problem and as a module in a policy improvement scheme. However, most Temporal Difference (TD) based solutions ignore the discrepancy between the stationary distribution of the behavior and target policies and its effect on the convergence limit when function approximation is applied. In this paper we propose the Consistent Off-Policy Temporal Difference (COP-TD( $\lambda$ ,  $\beta$ )) algorithm that addresses this issue and reduces this bias at some computational expense. We show that COP-TD( $\lambda$ ,  $\beta$ ) can be designed to converge to the same value that would have been obtained by using on-policy TD( $\lambda$ ) with the target policy. Subsequently, the proposed scheme leads to a related and promising heuristic we call log COP-TD( $\lambda$ ,  $\beta$ ). Both algorithms have favorable empirical results to the current state of the art online OPE algorithms. Finally, our formulation sheds some new light on the recently proposed Emphatic TD learning.

Reinforcement Learning (RL) techniques were successfully applied in fields such as robotics, games, marketing and more (Kober et al., 2013; Al-Rawi et al., 2015; Barrett et al., 2013). We consider the problem of offpolicy evaluation (OPE) – assessing the performance of a complex strategy without applying it. An OPE formulation is often considered in domains with limited sampling capability. For example, marketing and recommender systems (Theocharous Hallak, 2013; Theocharous et al., 2015) directly relate policies to revenue. A more extreme example is drug administration, as there are only few patients in the testing population, and sub-optimal policies can have life threatening effects (Hochberg et al., 2016). OPE can also be useful as a module for policy optimization in a policy improvement scheme (Thomas et al., 2015a). In this paper, we consider the OPE problem in an on-line setup where each new sample is immediately used to update our current value estimate of some previously unseen policy. We propose and analyze a new algorithm called COP-TD.

Our algorithm resembles [9] Emphatic TD that was extended by [10] to the general parametric form ETD( $\lambda$ ,  $\beta$ ). We clarify the connection between the algorithms and compare them empirically. Unlike ETD( $\lambda$ ,  $\beta$ ), COP-TD( $\lambda$ ,  $\beta$ )’s effectiveness depends on the available resources. The number of features can be adjusted accordingly to provide the most affordable approximation. The added cost is fine-tuning another step-size, though  $\beta$ ’s effect is less prominent.

Conclusions: Research on off-policy evaluation has flourished in the last decade. While a plethora of algorithms were suggested so far, ETD( $\lambda$ ,  $\beta$ ) by Hallak et al. (2015) has perhaps the simplest formulation and theoretical properties. Unfortunately, ETD( $\lambda$ ,  $\beta$ ) does not converge to the same point achieved by on-line TD when linear function approximation is applied. We address this issue with COP-TD( $\lambda$ ,  $\beta$ ) and proved it can achieve consistency when used with a correct set of features, or at least allow trading-off some of the bias by adding or removing features. Despite requiring a new set of features and calibrating an additional update function, COP-TD( $\lambda$ ,  $\beta$ )’s performance does not depend as much on  $\beta$  as ETD( $\lambda$ ,  $\beta$ ), and shows promising empirical results.

Paper Carles Discounted COP-TD [3]:

Central to reinforcement learning is the idea that an agent should learn from experience. While many algorithms learn in a purely online fashion, sample-efficient methods typically

make use of past data, viewed either as a fixed dataset, or stored in a replay memory (Lin 1993, Mnih et al. 2015). Because this past data may not be generated according to the policy currently under evaluation, the agent is said to be learning off-policy (Sutton and Barto 2018).

By now it is well-documented that off-policy learning may carry a significant cost when combined to function approximation. Early results have shown that estimating the value function off-policy, using Bellman updates, may diverge (Baird 1995), (Tsitsiklis and Van Roy 1997). More recently, value divergence was perhaps the most significant issue dealt with in the design of the DQN agent (Mnih et al. 2015), and remains a source of concern in deep reinforcement learning (van Hasselt, Guez, and Silver 2016).

Further, under off-policy learning, the quality of the Bellman fixed point suffers as studied by Kolter (2011) and Munos (2003). The value function error can be unboundedly large even if the value function can be perfectly approximated. Hence, even in the case where convergence to the fixed point with off-policy data occurs, solutions can be of poor quality. Thus, the existing TD learning algorithms with convergence guarantees under off-policy data (Maei et al. 2009), (Sutton et al. 2009) can still suffer from off-policy issues.

This paper studies the covariate shift method for dealing with the off-policy problem. The covariate shift method, studied by Hallak and Mannor (2017) and Sutton, Mahmood, and White (2016), reweights online updates according to the ratio of the target and behavior stationary distributions. Under optimal conditions, the covariate shift method recovers convergent behavior with linear approximation, breaking what Sutton and Barto (2018) call the “deadly triad” of reinforcement learning. We argue the method is particularly appealing in the context of replay memories, where the reweighting can be replaced by a reprioritization scheme similar to that of Schaul et al. (2016).

We improve on Hallak and Mannor’s COP-TD algorithm, which has provable guarantees but is difficult to implement in a deep reinforcement learning setting. First, we introduce a discount factor into their update rule to obtain a more stable algorithm. Second, we develop an alternative normalization scheme that can be combined with deep networks, avoiding the projection step necessary in the original algorithm.

**Conclusion:** In this paper we revisited Hallak and Mannor’s COP-TD algorithm and extended it to be applicable to the deep reinforcement learning setting. While these results on the Atari 2600 suite of games remain preliminary, they demonstrate the practicality of learning the covariate shift in complex settings. We believe our results further open the door to increased sample efficiency in deep reinforcement learning.

We emphasize that the instabilities observed when learning the covariate shift under prioritized sampling point to the importance of the data distribution used to learn the ratios. Which distribution is optimal will be the focus of future work. The covariate shift method is a “backward” off-policy method, in the sense that it corrects a mismatch between distributions based on past transitions. It would be interesting to combine our method to “forward” off-policy methods such as Retrace (Munos et al. 2016), which have also yielded good results on the Atari 2600 (Gruslys et al. 2018). Then, it would be interesting to understand whether overfitting does occur due to a smaller effective replay size, and how this can be addressed. Finally, an exciting avenue would be extending the method to the more general case where multiple policies have generated off-policy data, which would allow COP-TD to be applied in the standard control setting.



### 3.1 COP-TD

As stated in [3], here we move to the *policy evaluation* problem within *off-policy learning*, where we want to learn the value function  $V^\pi$  of a *target policy*  $\pi$  from samples drawn from  $P$  and a *behaviour policy*  $\mu$ . Some useful notation:

- The Bellman equation for the state value function can be expressed in vector notation as  $V^\pi = r_\pi + \gamma P_\pi V^\pi$ , where  $V^\pi \in \mathbb{R}^n$ ,  $r_\pi \in \mathbb{R}^n$  and  $P_\pi \in \mathbb{R}^{n \times n}$ . The value function is in fact the fixed point of the *Bellman operator*  $\mathcal{T}_\pi : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n \times n}$ , defined as  $\mathcal{T}_\pi V := r_\pi + \gamma P_\pi V$ . It defines a single step of *bootstrapping*: the process  $V^{k+1} := \mathcal{T}_\pi V^k$  converges to  $V^\pi$ .
- Let  $d \in \mathbb{R}^n$ ; we write  $D_d \in \mathbb{R}^{n \times n}$  for the corresponding diagonal matrix, and consider the weighted squared seminorm notation of vectors  $x \in \mathbb{R}^n$   $\|x\|_d^2 := \|Ax\|^2 = x^T A^T A x$ ,  $\|x\|_d^2 := \|x\|_{D_d}^2 = \sum_{i=1}^n d(i)^2 x(i)^2$ .
- $e \in \mathbb{R}^n$  accounts for the vector of all ones, and  $\Delta(\mathcal{S})$  for the simplex over states:  $d \in \Delta(\mathcal{S}) \implies d^T e = 1, d \geq 0$ .
- $d \in \Delta(\mathcal{S})$  is the stationary distribution of a transition function  $P$  if and only if  $d = d \cdot P$ . This distribution is unique when  $P$  defines a Markov chain with a single recurrent class[4].

In this particular setting we distinguish between two different state-to-state transition functions,  $P_\pi$  and  $P_\mu$ , one for each policy; their respective stationary distributions will be represented by  $d_\pi$  and  $d_\mu$ .

It is common to estimate the value function through *linear function approximation*, which uses a mapping from states to features  $\phi : \mathcal{S} \rightarrow \mathbb{R}^k$ . In these cases, the approximate value function at a state  $s$  can be expressed as the inner product of a feature vector with a vector of weights  $\theta \in \mathbb{R}^k$ :

$$\hat{V}(s) = \phi(s)^T \theta \quad (3.1)$$

which can be written as  $\hat{V} = \Phi \theta$  in vector notation, being  $\Phi \in \mathbb{R}^{n \times k}$  the matrix of row vectors. The *semi-gradient update rule* for TD learning[5] learns an estimation of  $V^\pi$  from sample transitions. Given a starting state  $s \in \mathcal{S}$ , a successor state  $s' \sim P_\pi(\cdot|s)$ , and a step-size parameter  $\alpha > 0$ , this update is

$$\theta \leftarrow \theta + \alpha [r_\pi(s) + \gamma \phi(s')^T \theta - \phi(s)^T \theta] \phi(s) \quad (3.2)$$

The expected behaviour of this update rule is described by the *projected Bellman operator*  $\Pi_d \mathcal{T}_\pi$ , a combination of the usual Bellman operator with a projection  $\Pi_d$  in norm  $\|\cdot\|_d$ - for some  $d \in \Delta(\mathcal{S})$ - onto the span of  $\Phi$ [6]. In fact, the stationary point of 3.2, if it exists, is the solution of the *projected Bellman equation*  $\hat{V}^\pi = \Pi_d \mathcal{T}_\pi \hat{V}^\pi$ . As stated in [3], not only convergence is proved for  $d = d_\pi$ , but this choice also seems optimal in terms of the quality of the fixed point under off-policy data.

Supposing that stationary distributions  $d_\pi$  and  $d_\mu$  are known, and that states are updated according to  $s \sim d_\mu$ , the covariate shift approach presented in [2] uses importance sampling to redefine 3.2 so that the semi-gradient update rule can be considered *under the sampling distribution*  $d_\pi$ :

$$\theta \leftarrow \theta + \alpha \frac{d_\pi(s)}{d_\mu(s)} [r(s, a) + \gamma \phi(s')^T \theta - \phi(s)^T \theta] \phi(s) \quad (3.3)$$

with  $a \sim \mu(\cdot|s)$ ,  $s' \sim P(\cdot|s, a)$  as before.

Both the original COP-TD learning rule[2] and its discounted version[3] seek to learn the ratio  $d_\pi/d_\mu$  from samples -by bootstrapping from a previous prediction, similar to temporal difference learning. For instance, given a sample transition  $(s_t, a_t, s_{t+1}) = (s, a, s')$  drawn from  $d_\mu$ ,  $\mu(\cdot|s)$  and  $P(\cdot|s, a)$ , respectively, and for  $c \in \mathbb{R}^n$ , step size  $\alpha > 0$  and discount factor  $\hat{\gamma} \in [0, 1]$ , the Discounted COP-TD provide us with the following update

$$c(s') \leftarrow c(s') + \alpha \left[ \hat{\gamma} \frac{\pi(a|s)}{\mu(a|s)} c(s) + (1 - \hat{\gamma}) - c(s') \right] \quad (3.4)$$

This update rule learns “in reverse” compared to TD learning, its expected behaviour being captured by the operator  $Y_{\hat{\gamma}}$

$$(Y_{\hat{\gamma}}c)(s') := \mathbb{E}_{s \sim d_\mu, a \sim \mu(\cdot|s)} \left[ \hat{\gamma} \frac{\pi(a|s)}{\mu(a|s)} c(s) + (1 - \hat{\gamma}) \middle| s' \right] = \hat{\gamma}(Yc)(s') + (1 - \hat{\gamma}) \quad (3.5)$$

where  $Y$  is the original COP operator

$$(Yc)(s') := \mathbb{E}_{s \sim d_\mu, a \sim \mu(\cdot|s)} \left[ \frac{\pi(a|s)}{\mu(a|s)} c(s) \middle| s' \right] \quad (3.6)$$

which corresponds to the undiscounted case,  $Y_1 = Y$ .

Note that the condition  $s_{t+1} = s'$  in the expectation of 3.6 forces to take into account the distribution of previous state-action pairs  $(s, a)$  according to policy  $\mu$ . The distribution of the possible previous states  $s$  is given by the time-reversal transition function  $\bar{P}_\mu$ , whose entries are:

$$\begin{aligned} \bar{P}_\mu(s|s') &:= \text{Prob}_\mu(s_t = s | s_{t+1} = s') \\ &= \frac{\text{Prob}_\mu(s_{t+1} = s' | s_t = s) \text{Prob}_\mu(s_t = s)}{\text{Prob}_\mu(s_{t+1} = s')} \\ &= \frac{P_\mu(s'|s) d_\mu(s)}{d_\mu(s')} \end{aligned} \quad (3.7)$$

Or, equivalently,  $\bar{P}_\mu = D_{d_\mu}^{-1} P_\mu^T D_{d_\mu}$  in vector notation. Regarding the distribution of the possible actions that lead to  $s'$  from a certain state  $s$  by following policy  $\mu$ , it will be represented by function  $\bar{\mu}$ :

$$\begin{aligned} \bar{\mu}(a|s, s') &:= \text{Prob}_\mu(a_t = a | s_t = s, s_{t+1} = s') \\ &= \frac{\text{Prob}_\mu(a_t = a, s_t = s, s_{t+1} = s')}{\text{Prob}_\mu(s_t = s, s_{t+1} = s')} \\ &= \frac{\text{Prob}_\mu(s_{t+1} = s' | a_t = a, s_t = s) \text{Prob}_\mu(a_t = a | s_t = s) \text{Prob}_\mu(s_t = s)}{\text{Prob}_\mu(s_{t+1} = s' | s_t = s) \text{Prob}_\mu(s_t = s)} \\ &= \frac{P(s'|s, a) \mu(s|a)}{P_\mu(s'|s)} \end{aligned} \quad (3.8)$$

With the introduced notation, the expectation in 3.6 can be rewritten and expanded in

the following way:

$$\begin{aligned}
\mathbb{E}_{s \sim d_\mu, a \sim \mu(\cdot|s)} \left[ \frac{\pi(a|s)}{\mu(a|s)} c(s) \middle| s' \right] &= \mathbb{E}_{s \sim \bar{P}_\mu(\cdot|s'), a \sim \bar{\mu}(\cdot|s, s')} \left[ \frac{\pi(a|s)}{\mu(a|s)} c(s) \right] \\
&= \sum_s \bar{P}_\mu(s|s') \sum_a \bar{\mu}(a|s, s') \frac{\pi(a|s)}{\mu(a|s)} c(s) \\
&= \sum_s \left( \frac{P_\mu(s'|s) d_\mu(s)}{d_\mu(s')} \right) \sum_a \left( \frac{P(s'|s, a) \mu(s|a)}{P_\mu(s'|s)} \right) \frac{\pi(a|s)}{\mu(a|s)} c(s) \quad (3.9) \\
&= \frac{1}{d_\mu(s')} \sum_s d_\mu(s) c(s) \sum_a \pi(a|s) P(s'|s, a) \\
&= \frac{1}{d_\mu(s')} \sum_s P_\pi(s'|s) d_\mu(s) c(s)
\end{aligned}$$

Thus, the COP and DCOP operators can be expressed in vector notation, respectively, as

$$Yc = D_{d_\mu}^{-1} P_\pi^T D_{d_\mu} c \quad (3.10)$$

$$Y_{\hat{\gamma}} c = \hat{\gamma} D_{d_\mu}^{-1} P_\pi^T D_{d_\mu} c + (1 - \hat{\gamma}) e \quad (3.11)$$

## 4 Distributional COP-TD

TODO: INTRO

Now we are interested in going beyond the notion of value and consider the estimation of the CS ratio from a distributional perspective, similarly to what was done in [1] within the reinforcement learning setting. Our starting point could be

$$X(s') \stackrel{D}{=} \frac{\pi(A_{s,s'}^\mu | S_{s'}^\mu)}{\mu(A_{s,s'}^\mu | S_{s'}^\mu)} X(S_{s'}^\mu) \quad (4.1)$$

where

- $X$  is the random ratio between distributions of achieving a certain state, its expectation being the covariate shift ratio

$$\frac{d_\pi}{d_\mu}(s) = c(s) = \mathbb{E}[X(s)] \quad (4.2)$$

- $(S_{s'}^\mu, A_{s,s'}^\mu)$  is the previous state-action random variable:
  - The random variable  $S_{s'}^\mu$  represents the states from which state  $s'$  is achievable by following policy  $\mu$ ;  $S_{s'}^\mu = s$  with probability  $\bar{P}_\mu(s|s')$
  - $A_{s,s'}^\mu$  encodes the random action that can be taken to get state  $s'$  from a state  $s \sim S_{s'}^\mu$  according to policy  $\mu$ , so  $A_{s,s'}^\mu = a$  with probability  $\bar{\mu}(a|S_{s'}^\mu, s')$

Recalling equation 4.1, note that it intrinsically expresses the random ratio of a state  $s_{t+1}$ ,  $X(s_{t+1})$ , as a mixture distribution with the 'corrected' previous state random ratios  $\rho(s_t, a_t)X(s_t)$  as mixing components, and the previous state-action random variable  $(S_{s_{t+1}}^\mu, A_{s_t, s_{t+1}}^\mu)$  as the mixing distribution:

$$f_{X(s_{t+1})}(x) = \sum_{s_t} \bar{P}_\mu(s_t|s_{t+1}) \sum_{a_t} \bar{\mu}(a_t|s_t, s_{t+1}) f_{\frac{\pi(s_t, a_t)}{\mu(s_t, a_t)} X(s_t)}(x) \quad (4.3)$$

**Notation.** So as to reduce the complexity and increase the readability of the formulation, we introduce the following notation:

- Let define  $\rho$  the policy ratio  $\pi/\mu$ :

$$\rho(a, s) := \frac{\pi(a|s)}{\mu(a|s)}$$

- Note in equation 4.3 that there are as many mixture components as state-action pairs  $(s_t, a_t)$ ; thus, we can iterate the summation over all these possible pairs and define each corresponding mixture weight  $\alpha$  as

$$\alpha(s_t, a_t; s_{t+1}) = \bar{P}_\mu(s_t|s_{t+1}) \bar{\mu}(a_t|s_t, s_{t+1})$$

Considering the previous notation, the expression of the density function 4.3 can be rewritten and expanded in the following way:

$$\begin{aligned}
f_{X(s_{t+1})}(x) &= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot f_{\rho(s_t, a_t)X(s_t)}(x) \\
&= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot f_{\rho(s_t, a_t)\rho(S_{s_t}^\mu, A_{s_{t-1}, s_t}^\mu)X(S_{s_t}^\mu)}(x) \\
&= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \sum_{(s_{t-1}, a_{t-1})} \alpha(s_{t-1}, a_{t-1}; s_t) \cdot f_{\rho(s_t, a_t)\rho(s_{t-1}, a_{t-1})X(s_{t-1})}(x) \\
&= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdots \sum_{(s_0, a_0)} \alpha(s_0, a_0; s_1) \cdot f_{(\prod_{i=t}^0 \rho(s_i, a_i))X(s_0)}(x)
\end{aligned} \tag{4.4}$$

Regarding its expectation, we can see that:

$$\begin{aligned}
\mathbb{E}[X(s_{t+1})] &= \int_{-\infty}^{\infty} x f_{X(s_{t+1})}(x) dx \\
&= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \int_{-\infty}^{\infty} x f_{\rho(s_t, a_t)X(s_t)}(x) dx \\
&= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \mathbb{E}[\rho(s_t, a_t)X(s_t)] \\
&= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \rho(s_t, a_t) \cdot \mathbb{E}[X(s_t)]
\end{aligned} \tag{4.5}$$

## Discounted Distributional CS Ratio

Having already develop the previous formulation, it is straightforward to turn equation 4.1 into its discounted version as it is done in [3] in the value-based case:

$$X(s_{t+1}) \stackrel{D}{=} \hat{\gamma} \rho(S_{s_{t+1}}^\mu, A_{s_t, s_{t+1}}^\mu)X(S_{s_{t+1}}^\mu) + 1 - \hat{\gamma} \tag{4.6}$$

as well as compute its corresponding expectation:

$$\mathbb{E}[X(s_{t+1})] = 1 - \hat{\gamma} \left( 1 - \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \rho(s_t, a_t) \cdot \mathbb{E}[X(s_t)] \right) \tag{4.7}$$

Given that  $\mathbb{E}[X(s)] = c(s)$ , the previous equation is equivalent to the result of applying of the DCOP operator, whose convergence is analyzed and proved in [3]. This can help us prove the convergence to a fixed point in the distributional setting, whose expectation matches with the covariate shift ratio estimate  $c$  in the value-based case.

## 5 Logarithmic Approach to Distributional COP-TD

Note that the Distributional Covariate Shift equation is purely multiplicative, and so it is the associated update rule in the learning setting.

Let's consider

$$Y(s_{t+1}) := \log(X(s_{t+1})) = \log(\rho(S_{s_{t+1}}^\mu, A_{s_t, s_{t+1}}^\mu)) + Y(S_{s_{t+1}}^\mu) \quad (5.1)$$

Its density function being

$$\begin{aligned} f_{Y(s_{t+1})}(x) &= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot f_{\log(\rho(s_t, a_t)X(s_t))}(x) \\ &= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot f_{\log(\rho(s_t, a_t)) + Y(s_t)}(x) \end{aligned} \quad (5.2)$$

Recalling **Property 3** of Mixture Distributions, we can express the expectation of  $Y$  as:

$$\begin{aligned} \mathbb{E}[Y(s_{t+1})] &= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \mathbb{E}[\log(\rho(s_t, a_t)X(s_t))] \\ &= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot \mathbb{E}[\log(\rho(s_t, a_t)) + Y(s_t)] \\ &= \sum_{(s_t, a_t)} \alpha(s_t, a_t; s_{t+1}) \cdot (\log(\rho(s_t, a_t)) + \mathbb{E}[Y(s_t)]) \end{aligned} \quad (5.3)$$

TODO: ADDITIVE FIXED POINT

And if

$$U(s') = \exp(Y(s')) \quad (5.4)$$

expectation:

$$\begin{aligned} \mathbb{E}[U(s_{t+1})] &= \int_{-\infty}^{\infty} \exp(y) f_{Y(s_{t+1})}(y) dy \\ &= \int_{-\infty}^{\infty} \exp(\log(x)) f_{X(s_{t+1})}(x) dx \\ &= \int_{-\infty}^{\infty} x f_{X(s_{t+1})}(x) dx \\ &= \mathbb{E}[X(s_{t+1})] \end{aligned} \quad (5.5)$$

## 6 Implementation

### 6.1 Categorical COP-TD

Analogously to [1], we can attempt to model the CS ratio distribution through a discrete parametric distribution with a certain set of atoms  $\{x_i\}_{0 \leq i < M}$ ,  $M \in \mathbb{N}$ , as its support. The atom probabilities would be given by a parametric model  $\theta : \mathcal{X} \rightarrow \mathbb{R}^M$

$$X_\theta(s) = x_i \quad w.p. \quad p_i(s) := f_\theta^i(s) \quad (6.1)$$

The DCOP update  $Y_{\hat{\gamma}}X_\theta$  and this parametrization  $X_\theta$  almost always would have disjoint supports. This could be tackled in practice projecting the sample DCOP update  $\hat{Y}_{\hat{\gamma}}X_\theta$  onto the support of  $X_\theta$  (i.e. given a sample transition  $(s, a, s')$ , we compute the DCOP update  $\hat{Y}_{\hat{\gamma}}x_j = \frac{\pi(a|s)}{\mu(a|s)}x_j + (1 - \hat{\gamma})$  for each atom  $x_j$ , then distribute its probability  $p_j(s)$  to the immediate neighbours of  $\hat{Y}_{\hat{\gamma}}x_j$ ).

The corresponding pseudocode is detailed in Algorithm 2, analogous to that of [1]:

---

**Algorithm 2:** Categorical CS Algorithm

---

**input:** A transition  $s_{t-1}, a_{t-1}, s_t$   
 $m_i = 0, i \in \{0, \dots, M-1\}$   
**for**  $j \in \{0, \dots, M-1\}$  **do**  
    #Compute the projection  $\tilde{T}x_j$  onto the support  $\{x_i\}$   
     $\tilde{T}x_j \leftarrow \left[ \hat{\gamma} \frac{\pi(a_{t-1}|s_{t-1})}{\mu(a_{t-1}|s_{t-1})}x_j + 1 - \hat{\gamma} \right]_{C_{min}}^{C_{max}}$   
     $b_j \leftarrow (\tilde{T}x_j - C_{min})/\Delta x$     #note that  $b_j \in [0, M-1]$   
     $l \leftarrow \lfloor b_j \rfloor, u \leftarrow \lceil b_j \rceil$   
    #Distribute probability of  $\tilde{T}x_j$   
     $m_l \leftarrow m_l + p_j(s_{t-1})(u - b_j)$   
     $m_u \leftarrow m_u + p_j(s_{t-1})(b_j - l)$   
**end for**  
**output:** Cross-entropy loss  $-\sum_i m_i \log(p_i(s_t))$

---

Observations:

- $C_{min}, C_{max} \in \mathbb{R}$  are the predefined lower and upper value limits of the covariate shift ratio. In this case,  $C_{min} \geq 0$ .
- The support is evenly spaced in  $[C_{min}, C_{max}]$ ; we consider the set of atoms  $\{x_i = C_{min} + i\Delta x : 0 \leq i < M\}$ , with  $\Delta x = \frac{C_{max} - C_{min}}{M-1}$
- The *behavioural policy*  $\mu$  is simply the uniformly random policy, so

$$\mu(a|s) = \frac{1}{|\mathcal{A}|} \quad \forall a \in \mathcal{A} \quad (6.2)$$

for any state  $s \in \mathcal{S}$ .

- The *target policy*  $\pi$  is the  $\epsilon$ -greedy policy with respect to the estimated state-action values of the model. Hence,

$$\pi_\theta(a|s) = \begin{cases} (1 - \epsilon) + \epsilon \frac{1}{|\mathcal{A}|} & \text{if } a = \arg \max_a Q_\theta(s, a) \\ \epsilon \frac{1}{|\mathcal{A}|} & \text{otherwise} \end{cases} \quad (6.3)$$

for any state  $s \in \mathcal{S}$ .

- Considering equations 6.2 and 6.3, we have that

$$\frac{\pi_\theta(a|s)}{\mu(a|s)} = \begin{cases} |\mathcal{A}|(1 - \epsilon) + \epsilon & \text{if } a = \arg \max_a Q_\theta(s, a) \\ \epsilon & \text{otherwise} \end{cases} \quad (6.4)$$

## 6.2 Exponential Bins

## 6.3 Log-Categorical COP-TD



## 7 Evaluation of the proposal

## 8 Conclusions

# Appendix

## A.1 Useful Distributional Notation

**Remark.** All random variables presented in this document are considered to be real-valued, i.e. their measurable space is  $E = \mathbb{R}$ .

### Mixture Distributions

A random variable  $Y$  is a mixture distribution if it is derived from a collection of other random variables  $\{X_i\}$ ,  $i \in \{1, \dots, N\}$ , (named mixture components) in such a way that the combination of these parent distributions is driven according to a certain distribution  $A$  (called mixing distribution).  $A$  encapsulates the mixture weights  $\alpha_i \sim A$ ,  $i \in \{1, \dots, N\}$ , which represent the probabilities of each individual mixture component  $X_i$ .

The mixture distribution  $Y$  can be defined in terms of its density function  $f_Y$ , which is the resulting  $\alpha$ -convex combination of the mixture components' density functions:

$$f_Y(x) = \sum_{i=1}^N \alpha_i f_{X_i}(x) \quad (\text{A.1})$$

Let's present some interesting properties of mixture distributions:

**Property 1.** *The expectation of the mixture distribution  $Y$  is the convex combination of expectations of each mixture component:*

$$\begin{aligned} \mathbb{E}[Y] &= \int_{-\infty}^{\infty} x f_Y(x) dx = \int_{-\infty}^{\infty} x \sum_{i=1}^N \alpha_i f_{X_i}(x) dx \\ &= \sum_{i=1}^N \alpha_i \int_{-\infty}^{\infty} x f_{X_i}(x) dx \\ &= \sum_{i=1}^N \alpha_i \mathbb{E}[X_i] \end{aligned} \quad (\text{A.2})$$

**Property 2.** *Let be  $Z$  a mixture distribution with mixture components  $\{g_i(X_i)\}$ ,  $i \in \{1, \dots, N\}$  and mixing weights  $\alpha_i \sim A$*

$$\mathbb{E}[Z] = \sum_{i=1}^N \alpha_i \mathbb{E}[g_i(X_i)] \quad (\text{A.3})$$

**Property 3.** *Let be  $Z = g(Y)$ , being  $Y$  a mixture distribution with mixture components  $\{X_i\}$ ,  $i \in \{1, \dots, N\}$ , and  $g$  a monotonic, invertible and differentiable function. Then we have that  $Z$  is a mixture distribution whose expectation is*

$$\begin{aligned} \mathbb{E}[Z] &= \int_{-\infty}^{\infty} g(x) f_Y(x) dx \\ &= \int_{-\infty}^{\infty} g(x) \left( \sum_{i=1}^N \alpha_i f_{X_i}(x) \right) dx = \sum_{i=1}^N \alpha_i \int_{-\infty}^{\infty} g(x) f_{X_i}(x) dx \\ &= \sum_{i=1}^N \alpha_i \mathbb{E}[g(X_i)] \end{aligned} \quad (\text{A.4})$$

*Note that in both the first and last steps the so-called Law of the Unconscious Statistician has been applied, which states that*

$$\mathbb{E}[g(X)] = \int_{-\infty}^{\infty} y f_{g(X)}(y) dy = \int_{-\infty}^{\infty} g(x) f_X(x) dx \quad (\text{A.5})$$

We emphasize the relevance of **Property 3**. In distributional TD learning, the distribution mixture plays the role of the expectation in expected TD. But while  $\mathbb{E}[g(x)] \neq g(\mathbb{E}[x])$ , we can interchange mixtures and functions. This allows us to circumvent Jensen’s inequalities.

## Sum of Distributions

The sum of two independent random variables  $X_1$  and  $X_2$ ,  $Y = X_1 + X_2$ , results in a random variable whose density function is the convolution of the density functions of each summand

$$f_Y(y) = \int_{-\infty}^{\infty} f_{X_1}(y-x) f_{X_2}(x) dx = (f_{X_1} * f_{X_2})(y) \quad (\text{A.6})$$

In the general case, considering a collection  $\{X_i\}$ ,  $i \in \{1, \dots, N\}$ , of independent random variables, the density function of the random variable  $Y = \sum_{i=1}^N X_i$  can be expressed as the convolution of all the individual density functions:

$$f_Y(x) = (f_{X_1} * \dots * f_{X_N})(x) \quad (\text{A.7})$$

Convolutions are linear operators.

## A.2 Metrics over Distributions

1. Kullback-Leibler divergence
2. Wasserstein
3. Cramér

## A.3 Contraction Mappings

## A.4 Implementation Details

TODO!!!

Our baseline is the C51 distributional reinforcement learning agent[1] within Dopamine framework[7]. We use published hyperparameters unless otherwise noted. We augment the C51 network by adding an extra head, the distributional ratio model  $X(s)$ , to the final convolutional layer, whose role is to predict the distribution of the ratio  $d_\pi/d_\mu$ . This model consists of a two-layer fully-connected network, with as many outputs as the number of atoms  $M$  of the parametric model. A final softmax layer transforms the resulting logits into probabilities.

## References

- [1] M. G. Bellamare, W. Dabney and R. Munos. A Distributional Perspective on Reinforcement Learning.
- [2] A. Hallak and S. Mannor. Consistent On-Line Off-Policy Evaluation.
- [3] C. Gelada and M. G. Bellamare. Off-Policy Deep Reinforcement Learning by Bootstrapping the Covariate Shift.
- [4] S. P. Meyn and R. L. Tweedie. Markov chains and stochastic stability. 2012.
- [5] Sutton, R. S., and Barto, A. G. 2018. Reinforcement learning: An introduction. MIT Press, 2nd edition.
- [6] Tsitsiklis, J. N., and Van Roy, B. 1997. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control* 42(5):674–690.
- [7] Dopamine: A Research Framework for Deep Reinforcement Learning. <https://github.com/google/dopamine>
- [8] Bertsekas, D. and Tsitsiklis, J. *Neuro-Dynamic Programming*. Athena Scientific, 1996.
- [9] Sutton, R. S., Mahmood, A. R., and White, M. An emphatic approach to the problem of off-policy temporal- difference learning. *arXiv:1503.04269*, 2015.
- [10] Hallak, Assaf, Tamar, Aviv, Munos, Remi, and Mannor, Shie. Generalized emphatic temporal difference learning: Bias-variance analysis. *arXiv preprint arXiv:1509.05172*, 2015.
- [11] Marc G. Bellemare, Nicolas Le Roux, Pablo Samuel Castro and Subhodeep Moitra. Distributional reinforcement learning with linear function approximation
- [12] Rowland, Mark, Bellemare, Marc G, Dabney, Will, Munos, Rémi, and Teh, Yee Whye. An analysis of categorical distributional reinforcement learning. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2018.
- [13] Jaquette, Stratton C. Markov decision processes with a new optimality criterion: Discrete time. *The Annals of Statistics*, 1(3): 496–505, 1973.
- [14] Bellemare, Marc G, Naddaf, Yavar, Veness, Joel, and Bowling, Michael. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.