Master MVA / Mathématiques, Vision, Apprentissage
École Normale Supérieure de Paris-Saclay

Internship 2021–2022
Alexandre Marthe

Distributional Reinforcement Learning and Quantile Optimization

Abstract : *We studied how the results of the theory of Reinforcement Learning can be transfered when trying to maximize a quantity such as the quantile of the return, instead of the mean. For that we used the recent work on Distributional Reinforcement Learning introduced by Bellemare et al., and tried to build upon it. We first showed that it was possible to get approximations of the quantile of the return for evaluating policies. We then showed how algorithms such as Value Iteration worked here. We exhibited a counter-exemple to the Bellman Optimality Principle, which doesn't hold anymore in this framework, preventing any guarantees from Dynamic algorithms. We finally showed that in some restricted sets of MDPs, optimal policies could still be chosen deterministic.*

Mots clefs : *Reinforcement Learning, Distributionnal Reinforcement Learning, Risk-sensitive, Quantile Optimization.*

Internship supervised by:
Aurelien Garivier
aurelien.garivier@ens-lyon.fr / tél. (+33) 4 72 72 81 08
UMPA, ENS de Lyon
46, allée d'Italie
69364 Lyon Cedex 07, FRANCE
http://www.umpa.ens-lyon.fr

September 27, 2022

Acknowledgment

I would like to first thank Aurelien Garivier for supervising me, guiding me throughout all the internship, but also for invitation to the summer school and the statistical conference. I got the opportunity to meet many people working close to my research area, and to discover tangent research subjects. I would also like to thank the team I was with at the ENS, Antoine, Aymen and Hugues, with who I shared many great moment, be it social or studious. And finally I would also like to thank the UMPA team as a whole, for welcoming me and making me immediatly feel at ease in the lab.

Contents

1 Introduction

In the last decade, Machine Learning has become a major field of interest in current scientific research, and has allowed for uncountable technological breakthroughs. Its techniques allows for automisation of many task that usually had to be done by hand by a human. We can find its application everywhere in today's world. Its strengh is to be able to solve complex problems for which we don't know an exact solution, by learning through examples and/or training. The main challenges of the field is to find what can be learned, and how it can be learned. Depending on the kind of tasks we want to solve, there are different subdomain of Machine Learning. In this report, we will focus on what is called *Reinforcement Learning*.

Reinforcement Learning is the subfield that investigates how to learn and adapt in real time. The principle is the same as how humans learn to do certain tasks: through trials and errors. It's for instance the paradigm used in recommandation systems or in the programs that beat humans in board games such as Chess or Go. The main ideas are quite old[1][2], but it gained a lot of hype when it started being associated with deep learning methods, which made it scalable to real world applications. Among the breakthroughs that brought a lot of interest in the field, we can find the development of Deepmind's Alphago, an algorithm that beat the world's best Go players, one of the last board games on which humans were still better than machines. Reinforcement Learning methods have been increasingly popular in many different domains, such as scheduling, robot control and telecommunications.

All those algorithms work on the same principle: optimizing a reward. Similarly to what happens in our brain, where our actions are mainly chosen in a way to maximise the dopamine produced, the algorithms chose actions depending on which will give the best reward. In the example of Chess or Go, the algorithm will receive a positive reward when winning the game, and a penalty when loosing. It could also be designed so that it gain/loses reward when taking/losing a piece. The reward is usually defined by hand depending on the objective. It is then for the algorithm to find the best sequence of actions in order to maximize the reward.

However, in many real world applications, the reward can be stochastic (for instance with only partial observation of the context) and thus, performing the same sequence of actions may not always lead to the same reward. This is why most of the research focus the maximisation of the expected reward, which works very well in practice. Though, some Reinforcement Learning algorithms can also be found in quite sensitive applications, like software driving autonomous vehicles. The issues with those software is that depending on the action chosen and the context, there could be incidents, such as a car crash and or death of person. As this should be avoided as much as possible, it is mandatory to make the safest algorithm possible. Doing so by defining specific rewards is particularly difficult, as it's most of the time impossible to predict the exact behavior of the algorithm solely with the rewards. This justifies a need of research in minimizing some risks in Reinforcement Learning. This branch is called Risk-sensitive Reinforcement Learning, and is almost as old as the first formalization of Reinforcement Learning.

There are different approaches in Risk-Sensitive RL. The one that we will get interest in have been tried the first time by [3][Morimura et al.], which will be detailed later.

In this report we will start by recalling the main results in Reinforcement Learning for known environement, later we will sum up some of the latest advancement in Distributional RL. We will then use those results to try to understand how the theory changes when the optimization is done on a different quantity, such as the quantile. Our goal is to undertand how the behaviors differ, and what founding results remain or disappear.

2 Related work

2.1 General RL Framework

Reinforcement Learning is most of all a constant interaction between an *Agent*, and an *Environement*. The agent can observe the environement and act at all time. The environement is always evolving and can give *rewards* to the agent depending on the current state and the action performed by the agent. The goal of the agent is to learn and find the best actions to perform in order to maximize the reward obtained.

More formally, we consider a sequence of discrete time steps $t = 1, 2, \dots$. At each timestep, the Agent will observe the environement state x_t. It will choose to perform an action a_t. The agent will receive the reward r_t and the environement will evolve to state x_{t+1}. The sequence $x_0, a_0, r_0, x_1, a_1, r_1, \dots$ is called *trajectory*

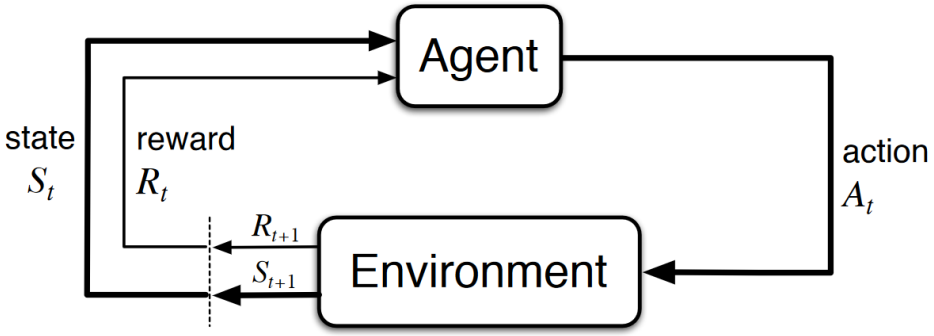


Figure 1: The agent-environement interaction[4]

In this report we will only consider the Framework of Countable MDPs and the discounted total reward criterion. We will first start by introducing the general framework of Markov Decision Process (MPD) and the basic results on dynamic programming.

Markov Decision Process

Definition 1 (Markov Decision Processes [5])**.** *An MPD is a tuple $\mathcal{M}(\mathcal{X}, \mathcal{A}, P, \gamma)$, where:*

- *$\mathcal{X}$ is a finite state space*
- *$\mathcal{A}$ a finite action space*
- *P a transition probability kernel that assigns to each pair $(x, a) \in \mathcal{X} \times \mathcal{A}$ a probability measure on $\mathcal{X} \times \mathbb{R}$*
- *$\gamma \in [0, 1[$ the discount*

The transition probability kernel P works as follow: consider a state x and an action a, P gives the probabilities to the next states and the reward received at this timestep, denoted by $P(\cdot|x, a)$. In what follows, we will write $p(x\prime|x, a) = P(\{x\} \times \mathbb{R}|x, a)$ (resp. $p(x\prime, r|x, a) = P(\{x\} \times r|x, a)$) the probability to reach state $x\prime$ (resp. and receiving reward r) when performing action a in state x. We also also define the random reward function r so that $r(x, a)$ gives a realisation of reward obtained following $P(\cdot|x, a)$. This reward will be considered bounded almost surely. It is important to notice that the probability to reach the next depends solely on the current state and the current action. What has happened before has no influence, it is the *Markov property*, that is always assumed is this Framework.

The MDPs gives the formalisation of an environnent, we then need to formalize what it means for the agent to learn. We already defined a reward, that we said we wanted to maximise. However, the

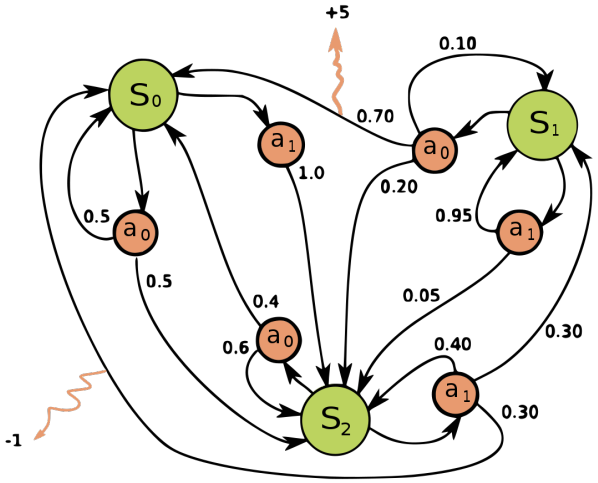


Figure 2: An exemple of MDP[6]

sequence of reward $r_0, r_1, r_2, \dots$ is random, the question is what do we maximise on this sequence. What is usually done, and what we will consider here, is to consider the sum of rewards, called the *Return*:

$$\mathcal{R} = r(x_0, a_0) + r(x_1, a_1) + \dots$$

There are two issues that arise with this definition, making it not really well defined. The first is the return being random, which is dealt with by considering the *mean* of the return, giving the *expected return*. The second is that, even with the expected return, we only assumed the reward to be bounded, so that return may not always be finite. This is where the discount γ from the definition of the MDP comes to play: we consider the *Discounted Expected Return* (later called simply "Return"). The goal of the agent will be to maximize this precise quantity:

$$R = \mathbb{E}\left[r(x_0, a_0) + \gamma r(x_1, a_1) + \gamma^2 r(x_2, a_2) + \dots\right] = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t)\right]$$

It is now well defined and bounded: if M is a bound on the reward almost surely, then :

$$|R| = \left|\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t)\right]\right| \leq \sum_{t=0}^{\infty} \gamma^t M \leq \frac{1}{1-\gamma} M$$

The discount is important to make the sum converging, but it also has some practical implications. Because of how it is defined, the later the reward is obtained, the less worth it is. The same reward received k step later is considered γ^k less. Implicity, it encourages the agent to get this reward as soon as possible, to accomplish a certain task quicker. This choice is quite arbitrary but leads to a rich theory because of its computational properties and ease to manipulate.

We also need to emphasize on the choice of the mean. While it is a quite intuitive choice to make here and will also lead to a rich theory, it reduces greatly the information on all the possiblities, and doesn't really take into account what happens in the extreme cases. It will require the study of the so-called "risk-sensitive" Reinforcement Learning.

Policies and value functions

We have a formalization of the environment and a quantity we want to optimize through actions. Before dwelling in the first results of the theory, we last need to formalize the choice of actions of the agent. For that we introduce *decision rules* and *policies*.

Definition 2. *A decisision rule d is a function that maps each state to a probability distribution on the action space :*

$$d : \mathcal{X} \mapsto \mathscr{P}(\mathcal{A})$$

It is said deterministic *if it of the form d:* $\mathcal{X} \mapsto \mathcal{A}$

Definition 3. *A policy is a sequence of decision rule:*

$$\pi = (d_0, d_1, d_2, \dots)$$

It is said stationnary *if it uses a unique decision rule.*

When exploring the environement, the agent will use a policy to choose its action. It works as follows: at each time step t, the agent will perform an action a sampled from $\pi(\cdot|x_t)$, which will lead to a reward and a next state sampled from $P(\cdot|x, a)$, then the policy gives another action, and so on.

To be able to evaluate each policies, we define the *value functions*:

Definition 4. *The Value function V is defined by:*

$$V^\pi(x) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) | x_0 = x\right]$$

with $x_t \sim p(\cdot|x_{t-1}, a_{t-1})$ *and* $a_t \sim \pi(\cdot|x_t)$

This value function $V(x)$ simply represents the expected discounted reward received following policy π and starting at state x. With this definition, we can now formalize the problem of RL as an optimization problem, the problem of finding an *optimal policy*:

Definition 5 (Optimal Policy)**.** *An optimal policy is a policy that verifies:*

$$\pi^\star \in \underset{\pi}{arg\,max}\ V^\pi(x) \quad \forall x \in \mathcal{X}$$

By convenience for the future, we also define the action-state value function :

Definition 6. *The Q-value function (also called action-state value function) is defined by:*

$$Q^\pi(x, a) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) | x_0 = x, a_0 = a\right]$$

with $x_t \sim p(\cdot|x_{t-1}, a_{t-1})$ *and* $a_t \sim \pi(\cdot|x_t)$

A first result on the policies

Now that the framework have been properly formalized, and before trying to compute the Value function and the optimal policies, this first result will simplify the theory

Definition 7 (The Discounted Occupancy Measure[2]). *Consider the MDP $\mathcal{M}(\mathcal{X}, \mathcal{A}, P, \gamma)$. We define the* Discounted Occupancy Measure p_γ^π *by:*

$$\forall x \in \mathcal{X}, a \in \mathcal{A}, \quad p_\gamma^\pi(x,a) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \mathbb{P}\left[x_t = x, a_t = a\right]$$

We then have the following rewriting of the value function:

$$V^\pi(x) = \frac{1}{1-\gamma}\sum_{x,a} p_\gamma^\pi(x,a)\mathbb{E}\left[r(x,a)\right]$$

This new writing of the value function, while seaming a bit more complicated, allows to summarize the influence of the policy in a unique term. It is used in the following results by Bertsekas:

Theorem 1 (Bertsekas, 2007). *Consider a MDP $\mathcal{M}(\mathcal{X}, \mathcal{A}, P, \gamma)$ with the previous assumptions. For any* non-stationnary policy π*, there exists a* stationnary policy $\bar{\pi}$ *such that*

$$\forall x \in \mathcal{X}, a \in \mathcal{A}, \quad p_\gamma^{\bar{\pi}}(x,a) = p_\gamma^\pi(x,a)$$

Combined with the rewriting of the value function, this mean that for any value function obtainend with a non-stationary policy, we can obtain the same with a stationary one.

Corollary 1. *If an optimal policy exists, then the optimal policy can be choosen to be stationnary.*

We can now restrain ourself to stationary policy when studying the theory further. From now on, except mentioned otherwised, all policies will be assumed to be stationary.

2.2 Dynamic Programming

Policy Evaluation

The first problem when considering a policy, is to evaluate it, i.e. compute the Value function. The issue with its definition, being a mean on a infinite sum of random variables, makes significantly hard to compute directly as it is. Fortunately by using the linearity of the mean and the infinite sum, we find that the value function (and the Q-value function) verifies a recursive formula, called *Bellman Equation*

$$V^\pi(x) = \sum_{a\in\mathcal{A}} \pi(a|x)\left(\mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x'\in\mathcal{X}} p(x'|x,a)V^\pi(x')\right)$$
$$Q^\pi(x,a) = \mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x',a'\in\mathcal{X}\times\mathcal{A}} p(x'|x,a)\pi(a'|x')Q^\pi(x',a')$$

Those equations give a linear system of equation, with as many constraints as unknown ($|\mathcal{X}|$ for the value function, $|\mathcal{X}||\mathcal{A}|$ for the Q-value function). Which means it can be solved through matrix inversion: with $V^\pi \in \mathbb{R}^{|\mathcal{X}|}, r^\pi \in \mathbb{R}^{|\mathcal{X}|}, P\pi \in \mathbb{R}^{|\mathcal{X}|\times\mathcal{X}}$,

$$V^\pi = r^\pi + \gamma P^\pi V^\pi \Longrightarrow V^\pi = \left(I - \gamma P^\pi\right)^{-1} r^\pi$$

This is very useful to compute the exact values in small instances, but there is another way to compute it, by iteration. For that, we will need to introduce the *Bellman Operator*

Definition 8 (Bellman Operator)**.** *Let $V : \mathcal{X} \mapsto \mathbb{R}$ or $Q : \mathcal{X} \times \mathcal{A} \mapsto \mathbb{R}$, π a policy. The Bellman operator $\mathcal{T}^\pi$ is defined by:*

$$\forall x \in \mathcal{X}, \qquad \mathcal{T}^\pi V(x) = \sum_{a \in \mathcal{A}} \pi(a|x) \left(\mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x' \in \mathcal{X}} p(x'|x,a) V(x') \right)$$

$$\forall x, a \in \mathcal{X} \times \mathcal{A}, \qquad \mathcal{T}^\pi Q(x,a) = \mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x', a' \in \mathcal{X} \times \mathcal{A}} p(x'|x,a)\pi(a'|x')Q(x',a')$$

Those operators are directly linked to the Bellman equation, which can be rewrited simpler this way :

$$\mathcal{T}^\pi V^\pi(x) = V^\pi(x)$$
$$\mathcal{T}^\pi Q^\pi(x,a) = Q^\pi(x,a)$$

From this we get that the value functions are fixed points of the Bellman Operators. Fixed points are essential in mathematics are allow for, under the right assumptions, for computation or proof or existence of object hard to handle. Most of it comes from the following result:

Theorem 2 (Banach fixed point[7])**.** *Let (X, d) be a non-empty complete metric space with a contraction mapping $T : X \mapsto X$. Then T has admits a unique fixed-point $x^\star \in X$ and*

$$\forall x \in X, \quad T^n(x) \longrightarrow x^\star \text{ exponentially}$$

The theorem is applicable here because we get the following property:

Proposition 1. *The bellman operators are a γ–contractions:*

$$\forall V_1, V_2 \in \mathbb{R}^{|\mathcal{X}|}, \quad ||\mathcal{T}^\pi V_1 - \mathcal{T}^\pi V_2||_\infty \leq \gamma ||V_1 - V_2||_\infty$$
$$\forall Q_1, Q_2 \in \mathbb{R}^{|\mathcal{X}| \times \mathcal{A}}, \quad ||\mathcal{T}^\pi Q_1 - \mathcal{T}^\pi Q_2||_\infty \leq \gamma ||Q_1 - Q_2||_\infty$$

The fixed point algorithm thus gives a very simple iteration algorithm to get the value function, called *Value Evaluation*: apply the Bellman Operator up to convergence with any starting values, up to a choosen point of convergence.

This algorithm will also give rise to 2 iteration algorithms to find optimal policies in a MPD.

Control

The second and main problem in Reinforcement Learning is to find a policy that maximizes the expected return. We to do so, the strategy will be quite similar to how we got to an algorithm for policy evaluation. First, we start by introducing the *Optimal Value functions*

Definition 9. *The optimal value function are defined by:*

$$V^\star(x) = \max_\pi \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) | x_0 = x \right] = V^{\pi^\star}(x)$$
$$Q^\star(x,a) = \max_\pi \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(x_t, a_t) | x_0 = x, a_0 = a \right] = Q^{\pi^\star}(x,a)$$

The definition of those functions can also be derived to show that they verify a recursive equation. Those one are called *Bellman Optimality Equations*:

$$V^\star(x) = \max_{a\in\mathcal{A}} \mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x'\in\mathcal{X}} p(x'|x,a)V^\star(x')$$
$$Q^\star(x,a) = \mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x'\in\mathcal{X}} p(x'|x,a) \max_{a'\in\mathcal{A}} Q^\star(x',a')$$

This equation is important not only for defining operators, but also because it leads to another theoretical results on policy. By looking at it, we notice that an optimal policy takes the action that maximizes the value function of the next state. An optimal policy can thus be chosen deterministic: this is the *Bellman optimality principle*

Proposition 2 (Bellman optimality principle[1])**.** *An optimal policy has the property that whatever the initial state and initial decision are, the remaining decisions must constitute an optimal policy with regard to the state resulting from the first decision*

Corollary 2. *If an optimal policy exists, then it can be chosen to be deterministic.*

Combining with the previous result on policies, although we started with very general policies, we find that the theory can be reduced to only using stationary and deterministic ones. This a specificity of the framework with the different choice that were, such as optimizing on a mean of a discounted return. We will see later that it may not be so easy in a different framework.

The previous results also suggests that there would only be $|\mathcal{X}||\mathcal{A}|$ policies to check, but evaluating and checking them all is still quite costly. Instead, we will proceed by introducing an optimal version of the Bellman Operator, adapted to the Optimal Value Functions.

Definition 10 (Optimal Bellman Operator)**.** *Let $V : \mathcal{X} \mapsto \mathbb{R}$ or $Q : \mathcal{X}\times\mathcal{A} \mapsto \mathbb{R}$, π a policy. The Bellman operator $\mathcal{T}^\star$ is defined by:*

$$\forall x \in \mathcal{X}, \qquad \mathcal{T}^\star V(x) = \max_{a\in\mathcal{A}} \mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x'\in\mathcal{X}} p(x'|x,a)V^\star(x')$$

$$\forall x,a \in \mathcal{X}\times\mathcal{A}, \qquad \mathcal{T}^\star Q(x,a) = \mathbb{E}\left[r(x,a)\right] + \gamma \sum_{x'\in\mathcal{X}} p(x'|x,a) \max_{a'\in\mathcal{A}} Q^\star(x',a')$$

These operators happen to verify the same contraction properties as the previous case:

Proposition 3. *The optimal bellman operators are a γ–contractions:*

$$\forall V_1, V_2 \in \mathbb{R}^{|\mathcal{X}|}, \quad ||\mathcal{T}V_1 - \mathcal{T}V_2||_\infty \leq \gamma||V_1 - V_2||_\infty$$
$$\forall Q_1, Q_2 \in \mathbb{R}^{|\mathcal{X}|\times\mathcal{A}}, \quad ||\mathcal{T}Q_1 - \mathcal{T}Q_2||_\infty \leq \gamma||Q_1 - Q_2||_\infty$$

We can now deduce an iterative algorith, called *Value iteration*: apply the Bellman Operator up to convergence with any starting values, up to a choosen point of convergence.

2.3 Metrics

In the next sections we will introduce the Framework for Distributional Reinforcement Learning. Before, we were only dealing with real numbers, now we will work on a space of Distribution. To do it properly, we need to introduce a few metrics.

Wasserstein Metric

Definition 11 (Wasserstein Metric[8]). *Let $p \geq 1$ and $\mathscr{P}_p(\mathbb{R})$ the space of distributions with finite p^{th} moment. Let $\nu_1, \nu_2 \in \mathscr{P}_p(\mathbb{R})$ and $\Lambda(\nu_1, \nu_2)$ the set of distribution on $\mathbb{R}^2$ with marginals ν_1 et ν_2. The p-Wasserstein distance d_p is then defined as :*

$$d_p(\nu_1, \nu_2) = \left(\inf_{\lambda \in \Lambda(\nu_1, \nu_2)} \int_{\mathbb{R}^2} |x - y|^p \, \mathrm{d}\lambda(x, y) \right)^{\frac{1}{p}}$$

Let $\eta_1, \eta_2 \in \mathscr{P}_p(\mathbb{R})^{\mathcal{X} \times \mathcal{A}}$. We also define the supremum-p-Wasserstein distance $\overline{d}_p$ by:

$$\overline{d}_p(\eta_1, \eta_2) = \sup_{(x,a) \in \mathcal{X} \times \mathcal{A}} d_p(\eta_1^{(x,a)}, \eta_2^{(x,a)})$$

This is the general definition, but it will not be practical to compute with. Instead, in our case, we have a simpler expression:

Lemma 1 ([9]). *Let $\nu_1, \nu_2 \in \mathscr{P}_p(\mathbb{R})$ with respective cumulative distribution F and G. Let $\mathcal{U}$ be a uniform random variable on $[0, 1]$. then*

$$d_p(\nu_1, \nu_2) = \left\| F^{-1}(\mathcal{U}) - G^{-1}(\mathcal{U}) \right\|_p$$

which, in the case $p < \infty$ simplifies to:

$$d_p(\nu_1, \nu_2) = \left(\int_0^1 \left| F^{-1}(u) - G^{-1}(u) \right|^p du \right)^{\frac{1}{p}}$$

d_1 is the one that will be used the most.

Cramer Distance

Definition 12 ([8]). *Let $\nu_1, \nu_2 \in \mathscr{P}(\mathbb{R})$. We define the family of metrics ℓ_p by :*

$$\ell_p(\nu_1, \nu_2) = \left(\int_{\mathbb{R}} (F_{\nu_1}(x) - F_{\nu_2}(x))^p \mathrm{d}x \right)^{\frac{1}{p}}$$

ℓ_2 is called the Cramer distance.
We also define the suppremum version of the ℓ_p norm:

$$\overline{\ell}_p(\eta_1, \eta_2) = \sup_{(x,a) \in \mathcal{X} \times \mathcal{A}} \ell_p(\eta_1^{(x,a)}, \eta_2^{(x,a)})$$

Remark 1. $l_1 = d_1$

2.4 Distributional Reinforcement Learning

In 2017, [9][Bellemare et al.] introduces the Distributional Reinforcemennt Learning framework. The idea is to compute the full distribution of the return instead of just the expected return. In his paper, they introduce the distributional Bellman operators and prove theoretical results on their properties.

The random return is the sum of the discounted random reward:

$$Z(x,a) = \sum_{t=0}^{\infty} \gamma R_t \mid X_0 = x, A_0 = a \tag{1}$$

The idea is that the distribution of the reward would follow similar Bellman equations:

$$Z(x,a) \overset{\mathrm{D}}{=} R(x,a) + \gamma Z(X', A') \tag{2}$$

with X', A' the random next state-action.

Policy Evaluation

Let's consider a policy π. The distribution of the random return under π will be written as follows:

$$\eta_\pi^{(x,a)} = \mathrm{Law}_\pi \left(\sum_{t=0}^{\infty} \gamma R_t \mid X_0 = x, A_0 = a \right)$$

and we will write η_π as the collection of distribution $(\eta_\pi^{(x,a)})_{(x,a)\in\mathcal{X}\times\mathcal{A}}$.

What makes the distributional framework worth studying, is the generalization of the Bellman equation and its properties: The random return associated to policy π verifies the *distributional Bellman equation*:

$$\eta_\pi = \mathcal{T}^\pi \eta_\pi$$

where $\mathcal{T}^\pi$ is the Bellman operator defined by:

$$\mathcal{T}^\pi \eta^{(x,a)} = \int_{\mathbb{R}} \sum_{(x',a')\in\mathcal{X}\times\mathcal{A}} (f_{r,\gamma})_\# \eta^{(x',a')} \pi(a'|x') p(r,x'|x,a) dr$$

with $(f_{r,\gamma})_\#\eta$ is the pushforward measure define by $f_\#\eta(A) = \eta(f^{-1}(A))$ for all Borel sets $A \subseteq R$ and $f_{r,\gamma}(x) = r + \gamma x$ for all $x \in R$.

While this operator seems more cumbersome that the non-distributional one, it just comes down to rewriting equation 2 for distributions. The proof use the exact same idea as in the non-distributional case, but in this new formalism.

In the tabular case, it is possible to solve this fixed point equation by matrix inversion. However, it doesn't seem possible to do so when dealing with distribution. To solve it, we will use the following result, that is same used to solve the non-tabular non-distributional case.

Similarly as in the non-distributional case, this operator is a γ-contraction under a well chosen metric: the maximal p-Wasserstein metric $\overline{d}_p$ (for $p \geq 1$).

This result is very important in the sense where it gives an theoretical algorithm to compute the value distribution of a policy.

$$\forall \eta \in \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}, \quad (\mathcal{T}^\pi)^n \eta \underset{n\to\infty}{\longrightarrow} \eta_\pi$$

The Wasserstein metric is important here because the same operator is not always a contraction under the total variation distance, the Kolmogorov distance or the Kullback-Liebler divergence. (ref in article de Bellemare)

Even though this algorithm seems promising, there are several issues that arise in practice, that make it difficult to implement: It is impossible to represent exactly all the space of distributions, which requires a parametrisation of the distribution and a projection step, then we most of the time don't have access to the exact transition of the MDP, which requires a stochastic estimation of the Bellman Operator. Those issues will be tackled in the next subsections.

Control

Here, the goal still is to find an optimal policy. However, we will consider the full distributions of return to reach it.

We define by optimal distribution a distribution associated to an optimal policy: $\eta^\star \in \{\eta_{\pi^\star} \mid \pi^\star \in \arg\max_\pi \mathbb{E}_{R\sim\eta_\pi}[R]\}$. One of the first difference that we notice is the fact that there can be several different optimal distribution. Those optimal distribution all have the same mean, but a distribution having the optimal mean may not be an optimal distribution, because some distributions may not come from any (stationary) policy. [mettre les exemples]

As expected, the optimal distributions verify the optimal distributional Bellman equation: $\eta^\star = \mathcal{T}\eta^\star$ with

$$\mathcal{T}\eta^{(x,a)} = \int_{\mathbb{R}} \sum_{(x',a')\in\mathcal{X}\times\mathcal{A}} (f_{r,\gamma})_\# \eta^{(x',a^\star(x'))} p(r,x'|x,a)dr$$

where $a^\star(x') = \arg\max_{a'\in\mathcal{A}} \mathbb{E}_{R\sim\eta^{(x',a')}}[R]$

The first interesting Control result is the fact that this operator is a contraction in average:

Lemma 2. *Let $\eta_1, \eta_2 \in \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}$, we write $\mathbb{E}[\eta] := \mathbb{E}_{Z\sim\eta}[Z]$. Then:*

$$\|\mathbb{E}[\mathcal{T}\eta_1] - \mathbb{E}[\mathcal{T}\eta_2]\|_\infty \leq \gamma \|\mathbb{E}[\eta_1] - \mathbb{E}[\eta_2]\|_\infty$$

Which means that $\mathbb{E}[\mathcal{T}^n\eta] \underset{n\to\infty}{\longrightarrow} Q^\star$ exponentially quickly.

As before this leads to a theoretical algorithm to find the optimal value function using the whole distribution. We have another result regarding the convergence of the distribution itself:

Theorem 3. *Let $\mathcal{X}$ and $\mathcal{A}$ be finite. Let $\eta \in \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}$. Assume that there is a single policy $\pi^\star$. Then:*

$$\mathcal{T}^n\eta \underset{n\to\infty}{\longrightarrow} \eta_{\pi^\star} \text{ uniformly in } \overline{d}_p,\ p \geq 1$$

This theorem is very important to understand how distributions behave. At first it seems really convenient, with the convergence of the distribution, but there are especially two points which are worth emphasizing. The first one is that there is no exponential convergence anymore and, in fact, the speed of the convergence is unknown. The second is the condition of unicity of optimal policy. While this condition seems reasonable, it is still possible to do without, at the cost of stationarity: if there are several optimal policy, the distribution converges uniformly to one associated to a possibly nonstationary optimal policy.

The non stationarity of the optimal policy isn't an issue when the goal is solely to maximize the mean reward, as the greedy policy associated to its distribution will still be optimal. However, it can be more problematic if we try to find policy that takes acount of the whole distribution, such a safer or riskier policy.

The two previous properties are weaker that what we found in the Policy Evaluation case. To emphasize more on the differences, here are some more results that underline the pathologic cases that arise in Distributional Control:

Proposition 4. *The optimality operators are not always contractions.*

Proposition 5. *The optimality operators do not always have fixed points.*

The lack of contraction is the precise result that prevents us to get the same properties as in the non-distributional case or as in the distributional, especially the existence and unicity of a fixed point, and the exponential convergence.

2.5 Distribution Parametrization

One of the main issue when dealing with distribution in practice, is the question of representation. It is not possible for a computer to represent the full extent of the distribution space. It is then necessary to restrain ourself on a parametrized space.

In their papers, [3, Morimura et al.] decide to parametrize the return distribution as a Gaussian, a Laplace or a skewed Laplace distribution. Later, [9, Bellemare et al.] and then [10, Dabney et al.] developed the theory for a richer class of parametric distributions, discrete ones, that are much more convenient. There two main approaches for that: the categorical approach, and the quantile regression approach.

Categorical

This is the approach introduced by [9, Bellemare et al.] which led to the C51 algorithm that reached state of the art result for ALE. However, the theoretical properties of such approach were mainly developed later, by [8, Rowland et al.].

The idea is to use the hypothesis of bounded reward to use evenly spread diracs on that reward support, and use the diracs weight as the parameters.

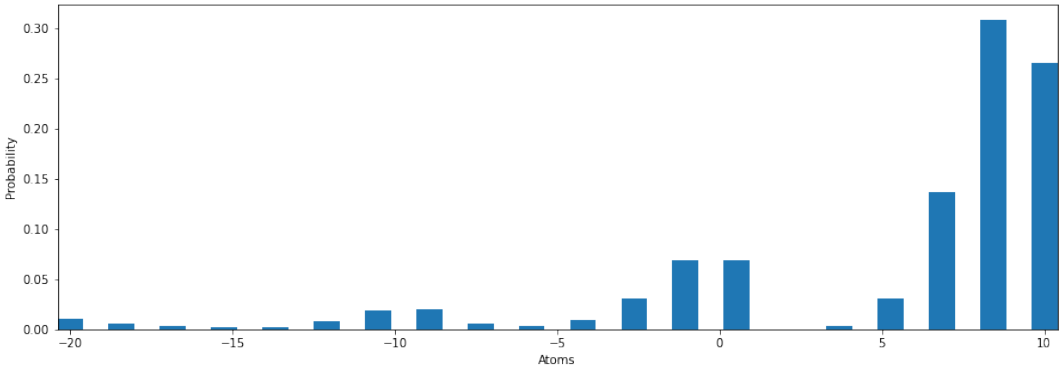


Figure 3: Example of a distribution projected by with the categorical approach

More formally, let's consider $V_{\text{MIN}}, V_{\text{MAX}}$ the bounds for the reward, N the number of diracs (the resolution) to use, $\Delta z = \frac{V_{\text{MAX}} - V_{\text{MIN}}}{N-1}$ the steps between diracs. The support of the distributions will be $\{z_i = V_{\text{MIN}} + i\Delta z \mid 0 \leq i < N\}$. The parametric family then is $\{\sum_{i=0}^{N-1} q_i \delta_{z_i} \mid \sum_{i=0}^{N-1} q_i = 1,\ 0 \leq q_i \leq 1\}$.

We define the projection operator $\Pi_C : \mathscr{P}(\mathbb{R}) \to \mathcal{P}_C$ by :

$$\Pi_C(\delta_y) = \begin{cases} \delta_{z_0} & y \leq z_0 \\ \frac{z_{i+1}-y}{z_{i+1}-z_i}\delta_{z_i} + \frac{y-z_i}{z_{i+1}-z_i}\delta_{i+1} & z_i < y < z_{i+1} \\ \delta_{z_{N-1}} & y \geq z_{N-1} \end{cases} \tag{3}$$

The idea behind is, for each atoms, to distribute their weight on the two closest of the projection, proportionnaly to how close they are.

Bellemare et al. introduced this projection step as an heuristic, without any theoretical motives or results related to the Wasserstein metric. It is Rowland et al. that later, found deep connection between this projection and another metric: the Cramer distance.

In fact, for the Wasserstein metric, we have the following result.

Proposition 6. *$\Pi_C \mathcal{T}^\pi$ is not a contraction for $\overline{d}_p$ with $p > 1$.*

For the case $p = 1$ it is however true, but only because it is the same as the ℓ_p distance, for which we have much more properties:

Proposition 7. *For a specific subset of $\mathscr{P}(R)$ and appropriate Hilbert space structure with ℓ_2, Π_C is the orthogonal projection of that subset onto $\mathcal{P}_C$*

Even though the result is quite Abstract, it gives a beginning of explication on why the definition of the projection makes sense, and why it gives decent results. It also highlight the fact that this projection is closely related to the cramer distance. That distance indeed leads to better result and gives a contracting operator:

Proposition 8. *$\Pi_C \mathcal{T}^\pi$ is a $\sqrt[p]{\gamma}$-contraction in $\overline{\ell}_p$.*

The Banach fixed point theorem thus provides with a proof a convergence of the iterated projected Bellman operators:

$$\exists! \eta_C \in \mathcal{P}_C^{\mathcal{X}\times\mathcal{A}},\ \forall \eta_0 \in \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}, \quad (\Pi_C \mathcal{T}^\pi)^m \eta_0 \underset{m\to\infty}{\longrightarrow} \eta_C \quad \text{exponentially quickly in } \overline{\ell}_p \tag{4}$$

It is important to notice that this does not have to converge to η_π, for the simple reason that this operator is convergent in the parametrized space $\mathcal{P}_C$, which may not contain η_π. The question that arises next, is how well does η_C approximates η_π, question which was answered by [8, Rowland et al.].

Lemma 3. *Let η_C defined as in (4). Assume that η_π is supported on $[z_0, z_{N-1}]$. Then:*

$$\overline{\ell}_2(\eta_C, \eta_\pi) \leq \frac{1}{1-\gamma}\Delta z$$

Quantile regression

This approach was first introduced by [10, Dabney et al.] and led to the QR-DQN algorithm that outperformed C51.

The idea is to do the opposite of the categorical approach: instead of having fixed reward support with variable weight, it considers fixed weight for variable support.

More formally, let's consider N the resolution. The parametric family is $\{\frac{1}{N}\sum_{i=0}^{N-1}\delta_{z_i} \mid (z_i) \in \mathbb{R}^n\}$

As the Wasserstein metric seems to be a metric of choice for this framework, it seems natural to try to project a distribution on the parametrized space by minimizing the Wasserstein distance between both. In this subsection we will use the 1-Wasserstein distance. The projection operator $\Pi_{d_1} : \mathscr{P}(\mathbb{R}) \to \mathcal{P}_Q$ is thus defined by:

$$\Pi_{d_1}\nu = \underset{\nu_Q \in \mathcal{P}_Q}{\arg\min}\, d_1(\nu, \nu_Q) \tag{5}$$

This is actually possible to compute, and the minimizers are exactly :

$$\Pi_{d_1}\nu = \frac{1}{N}\sum_{i=0}^{N-1}\delta_{z_i}, \quad F_\nu(z_i) = \frac{2i+1}{2N}$$

where F is the cumulative distribution funcition of ν.

This projection has the quality of being compatible with the Wasserstein metric, with which it gives a contracting operator.

Proposition 9. *$\Pi_{d_1}\mathcal{T}^\pi$ is γ-contraction in $\overline{d}_\infty$:*

$$\overline{d}_\infty(\Pi_{d_1}\mathcal{T}^\pi\eta_1, \Pi_{d_1}\mathcal{T}^\pi\eta_2) \leq \gamma\overline{d}_\infty(\eta_1, \eta_2)$$

This again leads to a convergence of the Bellman operator coupled with the projection step. However, no result have been found so far about a bound of distance between the true distribution of the policy and the policy it this operator converges to. The benefits of this projection is that it doesn't require to have prior knowledge on the reward distribution, where the categorical approach requires bounds. It is thus more convenient to use.

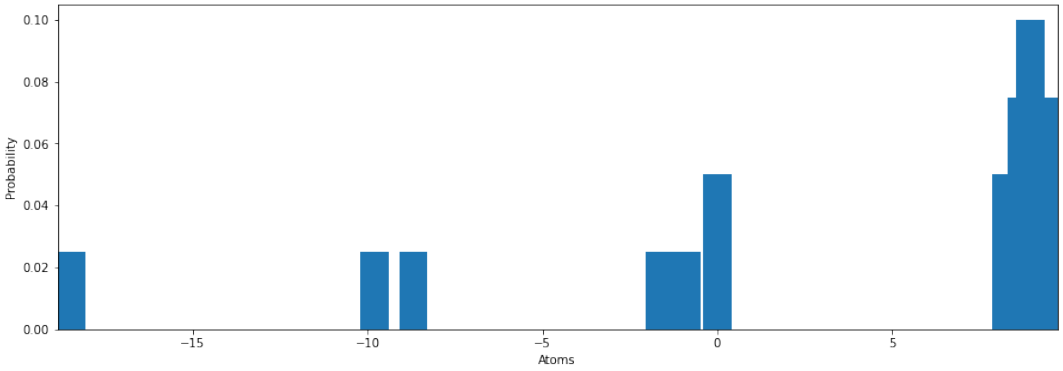


Figure 4: Example of a distribution projected with the quantile approach

3 Research Work and Results: the quantile Framework

In this part, we are going to change the goal, and try to maximise a quantile of the distribution, instead of the mean. Indeed, sometimes we may have specific applications where the mean doesn't matter so much, but where it is very important to have a safe policy, i.e. to have higher quantiles, even at the cost of lower mean. In the same way, it can be interesting to find more risky behavior on some environment, with higher possible reward, but at the cost of failing more often.

Quantile Optimization is a topic well studied in finance, in the framework of portfolios. However very few tackled this issue in the case of MDP. In their paper [3, Morimura et al.] try to apply their first distributional approach on an Q-learning algorithm trying to optimise quantiles. Even though they obtained empirically promising results, no theoretical results have been obtained se far. A main reason for that is because quantiles are particularly hard to compute. Fearless, we will still try to tackle this topic.

First it is important to notice that, as the goal changed, many assumptions that were made in the original case are to be studied again in this case. The theory as to be redone from 0.

3.1 Framework

We are still considering MDPs of the form $\mathcal{M}(\mathcal{X}, \mathcal{A}, P, R, \gamma)$, but with another value to optimize. We consider $x \in \mathcal{X}$ a specific state, and $\tau \in [0, 1]$ the quantile of interest. Our objective is:

$$\max_{\pi} V_\tau(x) = q_\tau \left(\sum_{t=0}^{\infty} \gamma R_t \mid X_0 = x \right)$$

Among the result that we are interested in finding back are the Bellman Optimality Principle, the sufficiency of deterministic policies and the computation of the different quantiles. Those are all results and algorithms that used some properties of the mean, mainly its linearity, which we do not have anymore.

Policy Evaluation

The first question, just as before, is how to evaluate a policy. Let π a policy. We want to compute:

$$Q_\tau(x, a) = q_\tau \left(\sum_{t=0}^{\infty} \gamma R_t \mid X_0 = x, A_0 = a, \pi \right)$$

When working with the mean, we would profit of its linearity to find an equation verified by this quantity, and solve this equation. Here, there is no linearity. In fact, is not even possible to compute a quantile solely knowing the quantiles for the next statest and action. We require the full distribution of the reward and the full distribution of reward and next state-action return.

With the development of the distributional approach, we have a way to compute the full distribution of the return for a specific policy. And once the distribution is known, so is the quantile.

In the case where it only requires a finite number of (distributional) Bellman operator application to get the exact distribution, we get to compute the quantile exactly. This happens for instance in MDP that ends after a finite number of steps.

if

$$\exists k \in \mathbb{N},\ \forall \eta_0 \in \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}, (\mathcal{T}^\pi)^k \eta_0 = \eta_\pi$$

then

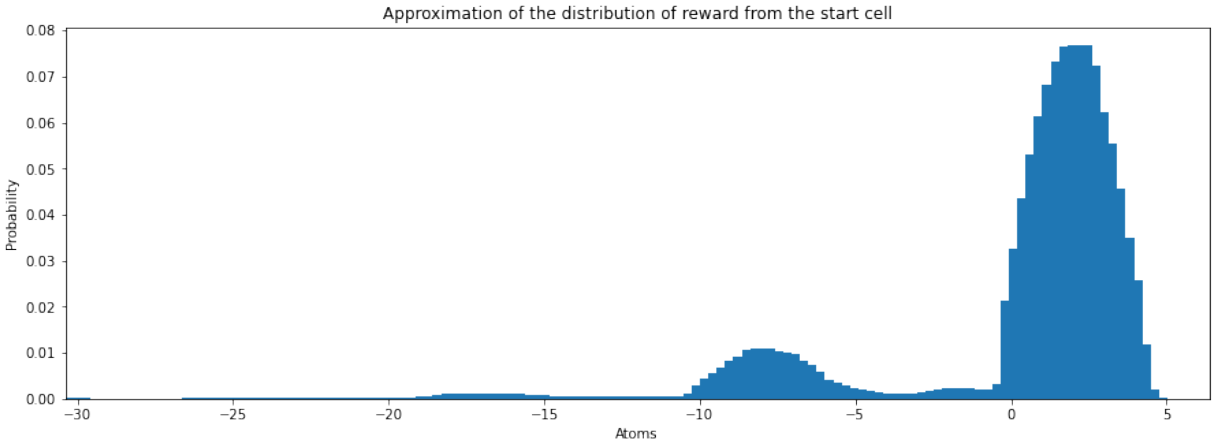
$$\forall \eta_0 \in \mathscr{P}(\mathbb{R})^{\mathcal{X}\times\mathcal{A}}, q_\tau \left((\mathcal{T}^\pi)^k \eta_0 \right) = q_\tau(\eta_\pi)$$

In the general case, even though we have the convergence of the distribution,it may not be enough for the convergence of the quantile.

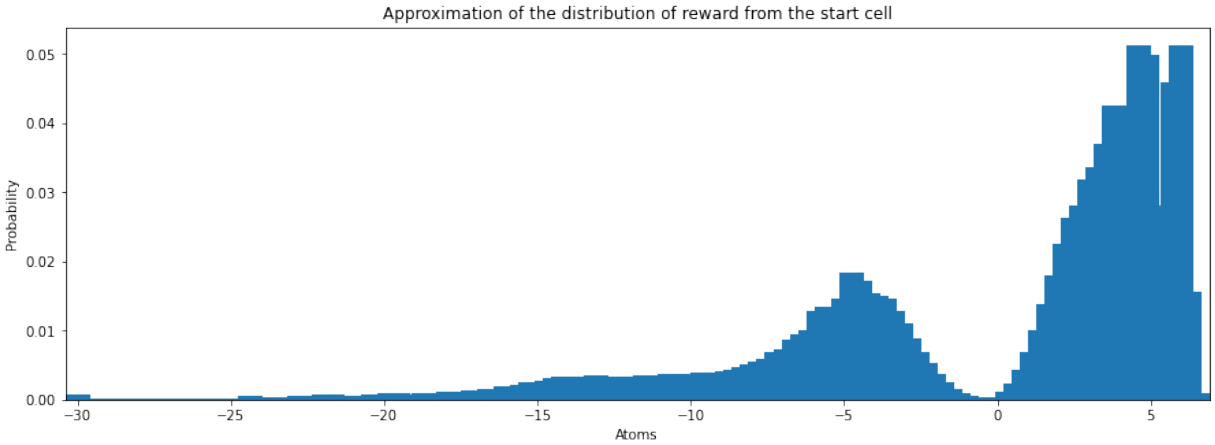
$$(\mathcal{T}^\pi)^n \eta \underset{n\to\infty}{\longrightarrow} \eta_\pi \quad \not\Rightarrow \quad q_\tau\left((\mathcal{T}^\pi)^n \eta\right) \underset{n\to\infty}{\longrightarrow} q_\tau(\eta_\pi)$$

The result on the convergence of the operators tells us that it converges for the Wasserstein Metrics. By definintion of that metric, it means that the quantile function converges for the L_p norm. However we cannot guarantee the point-wise convergence that we would need here. The closest we get is with the Riesz-Fischer theorem, that tells us that we can extract a sub-sequence converging almost everywhere. We can hope that in practice it will work well enough, which it does on the simulations.

Below are exemples of distributions evaluated for the MDP and the policies that will be introduced in the next section. They distribution parametrization was set the categorical one, with a resolution of 500. The discount parameter was 0.9. For the convergence, we applied the Bellman Operator 50 times.



(a) Safe policy distribution



(b) Risky policy distribution

Figure 5: Examples of return distributions

3.2 Environment of tests

To try to understand the behavior of this new Framework we had to choose an test environment. We chose to experiment on the Cliff environment, introduced by [Sutton and Barto], which is also the one used in the article of [Morimura et al.].

This environement is represented by a grid with a starting state (S), and a terminal state (E). There is also a pit between the two. The goal of the agent is to go from S to E without falling. The

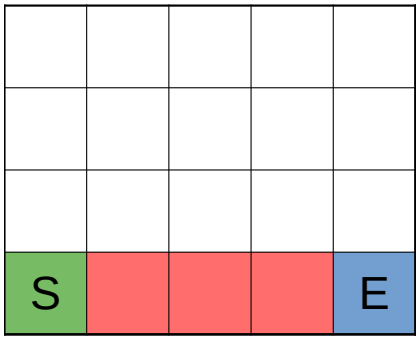
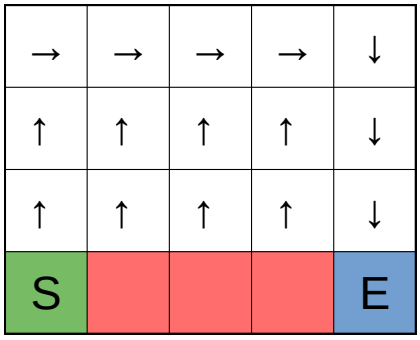


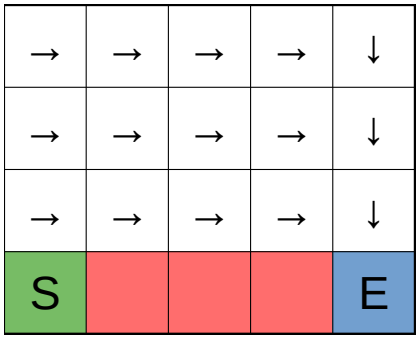
Figure 6: State space of the Cliff environment

reward received when reaching E is set to 10. The reward received when falling is set to -10 and when the agent does, it is teleported back to the start and has to start again. The agent can move in the 4 directions. When it hits a wall, it doesn't move. To add some random in it, at each step the agent has only 0.7% chances to go in the chosen direction, and has 0.1% chances to go any other direction.

This environment is particularly well suited for risk-sensitive RL, as it is easy to define safe and risky policy on it, making it easier to evaluate those new algorithms.



(a) Safe policy



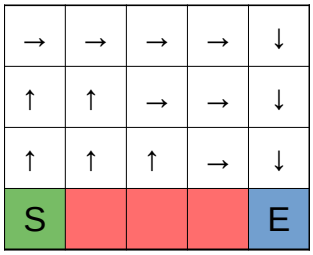
(b) Risky policy

Figure 7: Example of policies

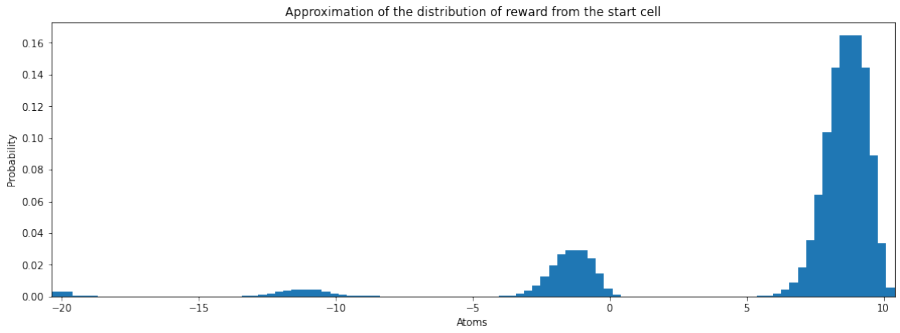
We chose a fixed set of parameters, and tested policy iteration on it, with different optimization criterion: optimization on the mean, on the median, on the quantile 20, on the quantile 0.8. We chose the quantile projection, with a resolution of 500, we each time applied the Bellman operator 50 times (except for the quantile 0.2), the discount γ was set to 0.99.

The results for the mean, is the following. The algorithm converges, as expected. The optimal policy for the chosen parameters is an in-between of the safely and the safe policy. At first it acts greedily, but reaching the end, the agent find it better to just risk it and trying to get sooner, with higher risks of falling, rather than playing it safe and arriving later with a discounted reward. The importance of the discounted is highlighted when comparing with a lower discount: the agent acts more risky as taking detours decreases much more the rewards.

The case with median is more complicated. The algorithm runs but does not converge. It ends up stuck in a loop of period 4. The following policy are what it is stuck with. On the policy we can see that at some cells, the algorithm find it better to go down. It doesn't seem to make sens since, be it for a quantile or for the mean, it should always be better to go the right first: it gets as close to the end cell, but stay further from the edges. By analysing the value distribution function obtained, it was shown that this behavior was due to the two actions leading to the same quantile, even though going to the right was better mean-wise. In our implementation, in case of equality, the

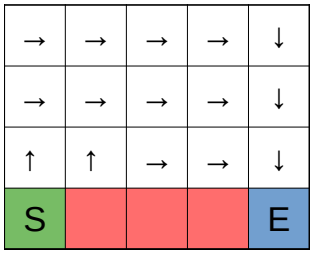


(a) Optimal policy

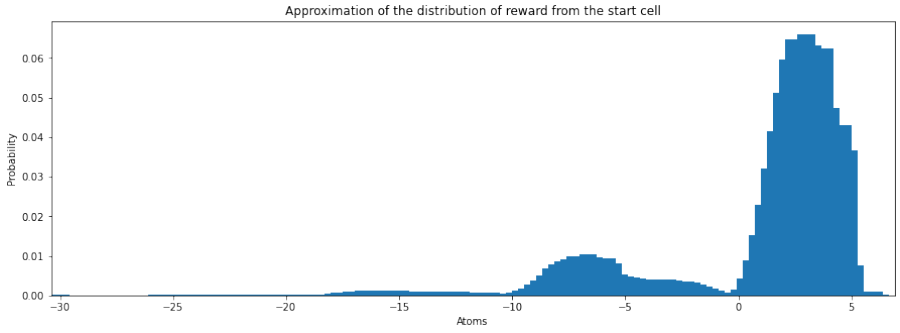


(b) Optimal distribution of return

Figure 8: Behavior on mean optimization $\gamma = 0.99$



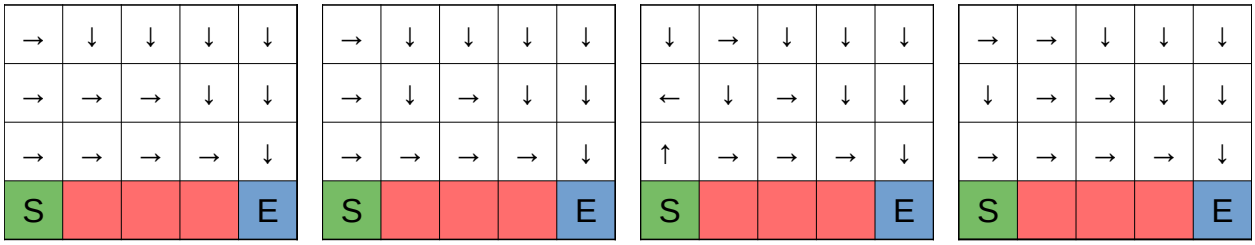
(a) Optimal policy



(b) Optimal distribution of return

Figure 9: Behavior on mean optimization, $\gamma = 0.9$

first action with maximum objective would always be chosen, which here was the down direction. In terms of theory, the quantile was equal because of the approximation necessary in the distribution parametrization, but should not be otherwise. Unfortunately, increasing the resolution would not fix the issue. Interestingly enough, this strange behavior would still result in a better median reward than the mean case.



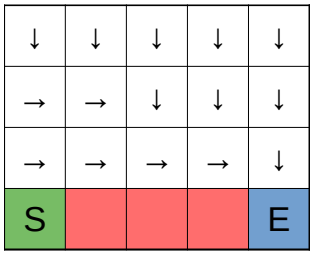
(a) 1st output policy (b) 2nd output policy (c) 3rd output policy (d) 4th output policy

Figure 10: Policies output by median optimisation

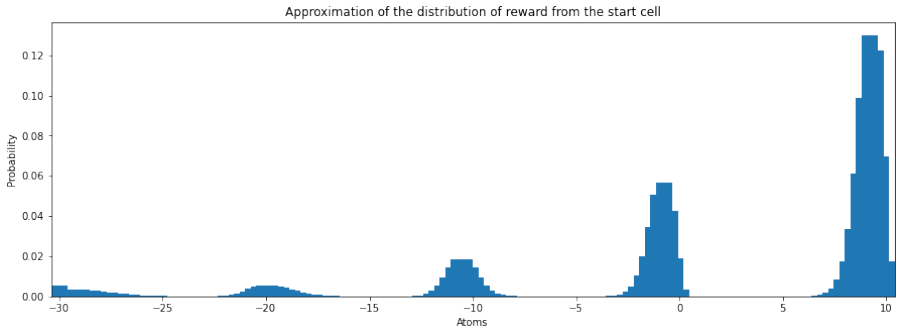
For the 0.8 quantile, it converged and also led to a very greedy policy, with a higher 0.8 quantile than in the mean case. However we still observe the issue with the agent willing to go down instead of right.

For the 0.2 quantile, we observe a much slower convergence, needing twice as many bellman operator applications, but it did converge. We observed a policy quite similar to the one obtained with the mean. However, for this case, the algorithm output a policy that gave a lower 0.2 quantile than the mean case.

In conclusion, except in a few instances, the algorithms converges and give results quite consistent

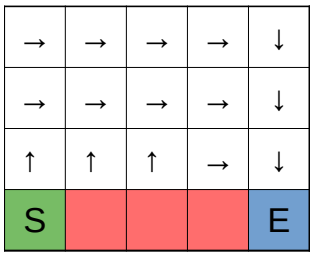


(a) output policy

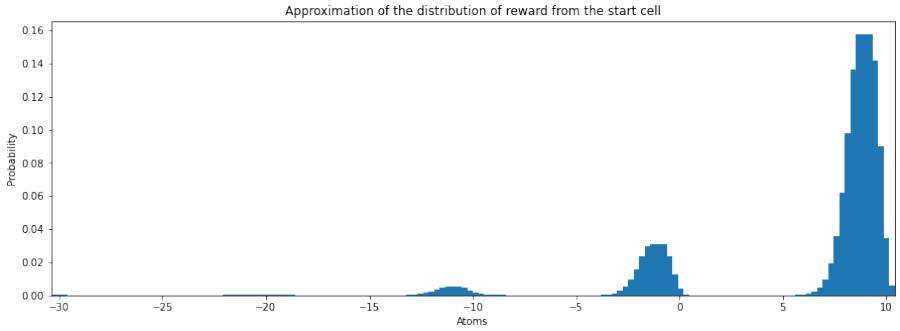


(b) Distribution of return

Figure 11: Behavior on 0.8 quantile optimazation



(a) output policy



(b) Distribution of return

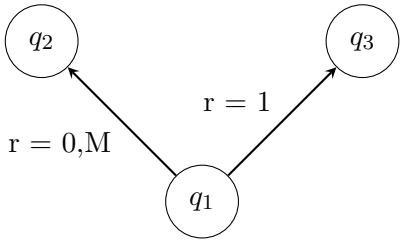
Figure 12: Behavior on 0.2 quantile optimization

with what is expected: greedy with a high quantile, safer with a low quantile. However there are cases where it does not converge. The issue seems to be the approximations in the distribution parametrization. Those approximations also lead to specific cases when quantiles are equals for differente action, which can lead to some illogical action choices. Moreover, this issue couldn't be solved simply by increasing the resolution in the parametrization.

3.3 Median versus Mean

When trying to tackle to RL with quantiles, one of our question was to know how does the mean differ to the median. Often in the distribution observed in practice, both are quite close. The previous tests showed that the policy optimizing the mean could differ to the one optimizing the mean, but the median and mean obtained wasn't significantly different. We wanted how large the difference could to know theoretically. The following example show that the difference in mean can be as high as wanted:

Here a simple example of environment where trying to maximize the median can give an arbitrary difference in mean, compared to the policy maximizing the mean. It also be adjusted to give a significant difference in the median.



The example consist in an initial state, and two terminal states. There are two actions, one for each terminal states. The first action leads to a reward of 0 with probability 0.51, and M with probability

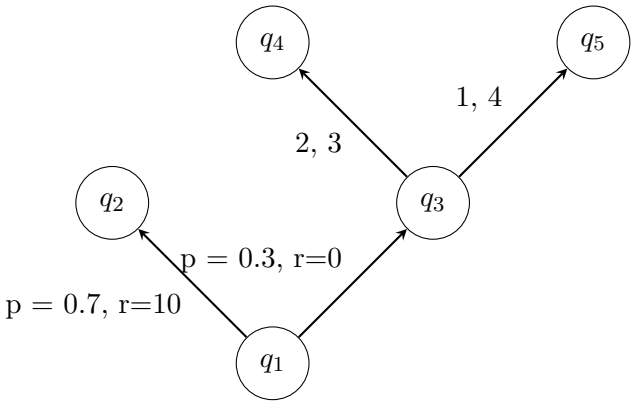
0.49 (M can be chosen as high as possible). The second action leads to a reward of 1 with probability 1. In this MDP, the optimal action for optimizing the median would be the second action, with mean 1, while the first action would lead to a mean of $M/2$

This example is a great example of the dilemna that is studied when trying to minimize risks: depending on what represents the reward and the context, is it better to have low reward 100% of the time, is it better to risk not having anything (or worse, having a penalty), but having low chances to get very high reward.

By tweaking the reward and probabilities in this example, it is possible to show that we can obtain any behavior with any quantile. Hence, in the general case, it is not possible to bound the gap between polices maximizing different quantiles and the mean.

3.4 About dynamic programming

Even though the approximation issues could have explained why the algorithm wouldn't work, dynamic programming will not be able to work properly. Indeed, it worked in the general Framework because, with the mean, we had the Bellman Optimality Principle. Unfortunately, this is not the case anymore. The following MDP is a counter-example:



On the first state, there is a single action, that have probability 0.7 to lead to state q_2 with reward 10, and probability 0.3 to lead to state q_3 with no reward. Then on $q3$ There are 2 actions possible, the first one having probabilities 0.15 and 0.85 to get reward 2 and 3, and the second one, the same probabilities to get rewards 1 and 4. We will consider the optimisation on the quantile 0.1. Optimizing this quantile from state q_3 would require to use to go to state q_4, with a quantile equal to 2. However this would lead to a quantile equal to 3 starting from state q_1. Doing the other action leads to a quantile equal to 4 if starting from state q_1. Thus, depending on the state we start from, the actions will be different.

This confirms the observation that policy iteration and value iteration algorithms would not work very well in practice. There will be no guarantee that they will work on some instances of choice, and the behavior might be unpredictable. Yet, it doesn't prevent the algorithms to output policies with better quantiles on some instances, and they could still be worse using.

Also, in the mean framework the Bellman Optimality Principle was quite convenient: even when wanting to optimize the return starting to a single beginning state, we would optimize it for every state simultaneously. Here, there is a choice to made, a choice on which state we want the quantile of the return optimized.

3.5 About policies

The first result we had on policies in the general framework, was that non-stationnary policies had the same expressive power as stationnary ones and thus, we could restrain ourself the latter. Yet, [Bellemare et al.] show that it was not the case anymore in Distributional RL, and that some distribution

could only be obtained through non-stationnary policies.

Then, investigating the theoretical properties of the quantile framework was particularly har because of the quantile itself. Among the few nice properties that it possess, we found this one, which has some implications with how optimal policies behave:

Lemma 4. *Let $n \in \mathbb{N}$, let $0 \leq \lambda_1, \lambda_2, \dots, \lambda_n \leq 1$ such that $\sum_{i=0}^{n} \lambda_i = 1$, and $\mu_1, \dots, \mu_n$ n distributions. Let q_τ the quantile function for $\tau \in [0,1]$. We have:*

$$q_\tau \left(\sum_{i=0}^{n} \lambda_i \mu_i \right) \leq \max_{1 \leq i \leq n} q_\tau(\mu_i)$$

Proof. Let $\mu_1, \dots, \mu_n$ probability measures. Let $0 \leq \lambda_1, \lambda_2, \dots, \lambda_n \leq 1$ such that $\sum_{i=0}^{n} \lambda_i = 1$. Let $\tau \in [0,1]$. We denote $q_i = q_\tau(\mu_i)$ and $q_{max} = \max_i q_\tau(\mu_i)$.

Recall that, $\forall \mu$ probability measure,

$$\begin{aligned} q_\tau(\mu) &= \inf\{x \in \mathbb{R} : \tau \leq \mu(]-\infty, x])\} \\ &= \inf\{x \in \mathbb{R} : \mu(]x, +\infty[) \leq 1 - \tau\} \end{aligned}$$

Hence,

$$\forall i, \mu_i \left(]q_{max}, +\infty[\right) \leq \mu_i \left(]q_i, +\infty[\right) \leq 1 - \tau$$

since $q_{max} \leq q_i$ and by monotonicity of the measure. Hence,

$$\left(\sum_{i=1}^{n} \lambda_i \mu_i \right) \left(]q_{max}, +\infty[\right) \leq \sum_{i=1}^{n} \lambda_i \mu_i \left(]q_i, +\infty[\right) \leq \sum_{i=1}^{n} \lambda_i (1 - \tau) \leq 1 - \tau$$

Therefore,

$$\left(\sum_{i=1}^{n} \lambda_i \mu_i \right) \left(]q_{max}, +\infty[\right) \leq 1 - \tau$$

which means $q_{max} \geq q_\tau \left(\sum_{i=1}^{n} \lambda_i \mu_i \right)$ □

What this result mean is that, when trying to optimize a quantile, if we have the choice between different distributions and combinations of them, it is always better to only choose the distribution with the highest quantile. This is precisely the situation where are in with the MDP, on one state, we try to choose the policy that will lead to the highest quantile. This suggests that, in value iteration, we can indeed restrain ourselves to an optimal action and not having to look for all convexe combination. This also make us believe in the fact that in any MDP, we should be able to find a deterministic policy that optimizes the quantile of choice.

This first result is encouraging, but it is not enough on itself to show that all optimal policies can be chosen to be deterministic. We conjectured it but have only been able to prove it to some restricted MDPs:

Proposition 10. *Consider a finite MDP where no state can be visited twice (i.e, without any loops). Consider a state $x \in X$, and $\tau \in [0,1]$. There exist an deterministic policy π_x^* that optimizes the τ quantile for state x :*

$$V_\tau^{\pi_x^*(x)} = \max_\pi V_\tau^\pi(x)$$

Sketch of proof. The idea is to write the distribution of reward starting at this state, depending on the distribution of all transition possible and with the coefficient of the policy. This lead to a convexe combination of distribution, where a distribution here would correspond to a *path* in the MDP, and its coefficient would be a multiplication of several policy parameters.

The previous result tells us that, to optimize the quantile, we can restrain ourself to only one *path*. It coefficient can be set to 1 and the others to 0. This leads to all the policy coefficients being either 0 or 1, which means we have a deterministic policy. □

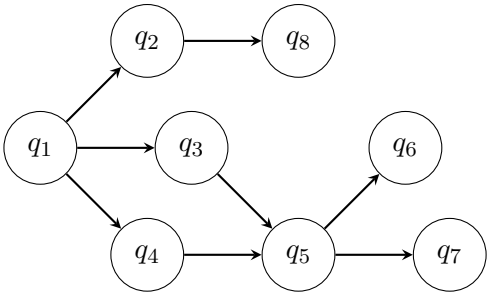


Figure 13: Example of an MDP on which the proposition applies

We also tested the conjecture on several simple MDPs that included loops, and everytime it seemed that the optimal policy could be chosen to be deterministic. No counter-example have been found so far. This conforts us in believing in the conjecture. However, the proof has yet to be found. The issue mainly comes from the fact that, with loops, the policies parameters appear several time multiplied to themselves, and the sum is not a convexe combination anymore, preventing us from using our only inequality on quantiles.

Having the most general result would be a theoretical advancement in the understanding of the framework, but may not bring much in the practical point of view. Currently, with no dynamic programming, the problem of finding the policy that maximizes a certain quantile is intractable, because of the continuum of possibilites on the policies. Looking only for a deterministic policy would help making the problem exponential in complexity (without considering the approximations necessary when dealing with distributions), which is usually too much for any real application.

4 Conclusion

As our first contributions, we implemented a little python librairy to experiment in an easy way with Distributional Reinforcement Learning (the code is available at https://github.com/amarthe/stageM2). This is the code used to obtain all the results mentioned in the report and, using the same parameters, they should all be reproductible. In practical point of view, this code helped illustrated how the usual tabular algorithms behaved in that new Framework, and helped us understand better the framework in general. We also contributed on the theory, by proving results and exhibiting counter-examples. We showed that Dynamic Programming was not possible on this kind of framework, when it was only assumed before, and we partly proved a conjecture stating that deterministic policy are suffiencient, which hadn't be questioned before.

The internship was also the opportunity to try evaluate how difficult working with quantile was, and that in the future it would be better to work with difference quantities such as the expectile, which displays much better properties.

For the futur, there could be different ways to continue this work further. The first one would indeed be to prove or disprove the conjecture on deterministic optimal policies. Ideally it would also be to find algorithms with theoretical guarantees to evaluate and optimize the quantiles, but the theory seem currently to far behind for it to be possible. Another one would be to re-do the work of the internship but with expectiles, and improving the results further.

References

[1] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[2] D. Bertsekas, *Dynamic programming and optimal control: Volume I*, vol. 1. Athena scientific, 2012.

[3] T. Morimura, M. Sugiyama, H. Kashima, H. Hachiya, and T. Tanaka, "Parametric Return Density Estimation for Reinforcement Learning," *arXiv:1203.3497 [cs, stat]*, Mar. 2012. arXiv: 1203.3497.

[4] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[5] C. Szepesvári, "Algorithms for reinforcement learning," *Synthesis lectures on artificial intelligence and machine learning*, vol. 4, no. 1, pp. 1–103, 2010.

[6] Wikipedia contributors, "Markov decision process — Wikipedia, the free encyclopedia," 2022.

[7] W. Rudin, *Functional Analysis*. International series in pure and applied mathematics, McGraw-Hill, 1991.

[8] M. Rowland, M. Bellemare, W. Dabney, R. Munos, and Y. W. Teh, "An Analysis of Categorical Distributional Reinforcement Learning," in *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*, pp. 29–37, PMLR, Mar. 2018. ISSN: 2640-3498.

[9] M. G. Bellemare, W. Dabney, and R. Munos, "A Distributional Perspectiv e on Reinforcement Learning," *arXiv:1707.06887 [cs, stat]*, July 2017. arXiv: 1707.06887.

[10] W. Dabney, M. Rowland, M. G. Bellemare, and R. Munos, "Distributional Reinforcement Learning with Quantile Regression," *arXiv:1710.10044 [cs, stat]*, Oct. 2017. arXiv: 1710.10044.