University of Liège
School of Engineering

# Reinforcement Learning in a Discrete Domain

INFO8003-1: Optimal decision making for complex problems

Julien Gustin, Joachim Houyon

March 1, 2022

# 1 Implementation of the domain

Our rule based policy is to always go right. More formally it can be described as:

$$\mu(x, y) = (1, 0) \quad \forall x, y \in X \tag{1}$$

Note that in the following for clearness we will represent actions

$$U = \{(1, 0), (-1, 0), (0, -1), (0, 1)\}$$

as follow:

$$U = \{RIGHT, LEFT, UP, DOWN\}$$

Also the state (x, y) does not mean "element at the line x and column y", but element at position [x][y] in the table where (0, 0) is on the top left

| | | | | |
|---|---|---|---|---|
| -3 | 1 | -5 | 0 | 19 |
| 6 | 3 | 8 | 9 | 10 |
| 5 | -8 | 4 | 1 | -8 |
| 6 | -9 | 4 | 19 | -5 |
| -20 | -17 | -4 | -3 | 9 |

Figure 1. Domain instance

Therefore by taking the example of the statement the initial state is at position $(0, 3)$ and not $(3, 0)$.

Other components of the domain remain the same.

## 1.1 Deterministic

In the deterministic domain the action (1) is always applied to the environment.

Here is a simulation through a single trajectory of 10 steps, starting by the initial state $x_0 = (0, 3)$ represented as tuple $(x_0, u_0, r_0, x_1), ..., (x_9, u_9, r_9, x_{10})$

```
--- Deterministic ---

0.  ((0,3), RIGHT, -9, (1,3))
1.  ((1,3), RIGHT, 4, (2,3))
2.  ((2,3), RIGHT, 19, (3,3))
3.  ((3,3), RIGHT, -5, (4,3))
4.  ((4,3), RIGHT, -5, (4,3))
5.  ((4,3), RIGHT, -5, (4,3))
6.  ((4,3), RIGHT, -5, (4,3))
7.  ((4,3), RIGHT, -5, (4,3))
8.  ((4,3), RIGHT, -5, (4,3))
9.  ((4,3), RIGHT, -5, (4,3))
```

Listing 1. Simulated trajectories by applying policy (1) in the deterministic domain.

## 1.2 Stochastic

In the stochastic domain the action (1) may not be applied due to the noise $w \sim \mathcal{U}(0, 1)$. Such that if $w \geq 0.5$ the state is updated to $(x, y) = (0, 0)$.

Here is a simulation through a single trajectory of 10 steps, starting by the initial state $x_0 = (0, 3)$ represented as tuple $(x_0, u_0, r_0, x_1), ..., (x_9, u_9, r_9, x_{10})$

```
--- Stochastic ---

0.  ((0,3), RIGHT, -9, (1,3))
1.  ((1,3), RIGHT, -3, (0,0))
2.  ((0,0), RIGHT, -3, (0,0))
3.  ((0,0), RIGHT, -3, (0,0))
4.  ((0,0), RIGHT, 1, (1,0))
5.  ((1,0), RIGHT, -5, (2,0))
6.  ((2,0), RIGHT, 0, (3,0))
7.  ((3,0), RIGHT, -3, (0,0))
8.  ((0,0), RIGHT, -3, (0,0))
9.  ((0,0), RIGHT, -3, (0,0))
```

Listing 2. Simulated trajectories by applying policy (1) in the stochastic domain.

# 2 Expected return of a policy

As seen on the course there exist a bound of the difference between $J_N^\mu$ and $J^\mu$ such that

$$||J_N^\mu - J^\mu||_\infty \leq \frac{\gamma^N}{1-\gamma} Br$$

where

$$J_N^\mu(x) = \underset{w \sim P_w(\cdot|x,u)}{E} \left[ r(x, \mu(x), w) + \gamma J_{N-1}^\mu(f(x, \mu(x), w)) \right], \quad \forall N \geq 1 \tag{2}$$

with $J_0^\mu(x) = 0$

Therefore we can find a lower bound of $N$ by fixing the error of approximation $\epsilon$, where $\epsilon = ||J_N^\mu - J^\mu||_\infty$.

$$\begin{aligned} \epsilon &\leq \frac{\gamma^N}{1-\gamma} Br \\ \frac{\epsilon(1-\gamma)}{Br} &\leq \gamma^N \\ \log_\gamma \left( \frac{\epsilon(1-\gamma)}{Br} \right) &\leq N \end{aligned}$$

Given $\gamma = 0.99$, $Br = 19$ and $\epsilon = 10^{-3}$ we have:

$$N = \left\lceil \log_\gamma \left( \frac{\epsilon(1-\gamma)}{B_r} \right) \right\rceil = 1439. \tag{3}$$

## 2.1 Deterministic

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 1839.617 | 1857.189 | 1880.999 | 1899.999 | 1899.999 |
| **1** | 990.039 | 997.009 | 998.999 | 999.999 | 999.999 |
| **2** | −779.299 | −779.090 | −791.000 | −800.000 | −800.000 |
| **3** | −471.567 | −467.240 | −476.000 | −500.000 | −500.000 |
| **4** | 849.368 | 875.120 | 888.000 | 900.000 | 900.000 |

Table 1. $J_N^\mu(x, y)$ for all $(x, y) \in X$ in the deterministic domain; $N = 1439$

## 2.2 Stochastic

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | −72.021 | −71.455 | −64.253 | −54.753 | −54.753 |
| **1** | −67.781 | −64.911 | −64.164 | −63.664 | −63.664 |
| **2** | −77.413 | −73.258 | −76.986 | −81.486 | −81.486 |
| **3** | −75.348 | −68.075 | −66.515 | −78.515 | −78.515 |
| **4** | −82.342 | −74.124 | −70.654 | −64.654 | −64.654 |

Table 2. $J_N^\mu(x, y)$ for all $(x, y) \in X$ in the stochastic domain; $N = 1439$

# 3 Optimal policy

We define

$$r(x,u) = \underset{w\sim P_w(\cdot|x,u)}{E}[r(x,u,w)] \quad \forall x \in X, u \in U$$

$$p(x' \mid x,u) = \underset{w\sim P_w(\cdot|x,u)}{E}[I_{\{x'=f(x,u,w)\}}] \quad \forall x, x' \in X, u \in U$$

which defines the structure of a MDP. Thanks to this, we can rewrite the functions $Q_N$ as follows:

$$Q_0(x,u) = 0$$

$$Q_N(x,u) = r(x,u) + \gamma \sum_{x'\in X} p(x' \mid x,u) \underset{u'\in U}{max} Q_{N-1}(x',u') \forall N \geq 1 \tag{4}$$

## 3.1 Choice of N

We wish to find the smallest $N$ such that

$$||Q_N - Q||_\infty = 0$$

because once we reach this equality, we know that the optimal policy $\mu^*$ will be the same as the policy $\mu_N^*$ derived from $Q_N$, ensuring that any increase of $N$ will not modify the policy found.

However, it is not possible to determine such value of $N$ which respects this property, we can only bound the suboptimality of $\mu_N^*$ with respect to $\mu^*$:

$$||J^{\mu_N^*} - J^{\mu^*}||_\infty \leq \frac{2\gamma^N B_r}{(1-\gamma)^2}$$

It follows that

$$\underset{N\to+\infty}{lim} \frac{2\gamma^N B_r}{(1-\gamma)^2} = 0$$

and we can only compute a value $N$ such that the gap from optimality is bounded by a value $\epsilon$.

Using $\epsilon = 10^{-3}$, we get that

$$N = \left\lceil \log_\gamma \left( \frac{\epsilon(1-\gamma)^2}{2B_r} \right) \right\rceil = 1966. \tag{5}$$

Which is very likely to have $\mu_N^* = \mu^*$ because the rewards are quite big numbers compared to the bound $\epsilon$.

With $J^{\mu_N^*}(x) = \max_{u \in U} Q_N(x,u)$ and $\mu_N^* \in arg\ max_{u \in U}\ Q_N(x,u)$

## 3.2 Deterministic domain

In the deterministic domain, it is clear that

$p(x' \mid x, u) = 1\ \forall x, x' \in X, u \in U$ and $r(x,u) = R(g, F(x,u))\ \forall x \in X, u \in U$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 1842.031 | 1857.190 | 1881.000 | 1900.000 | 1900.000 |
| **1** | 1854.576 | 1870.279 | 1881.090 | 1891.000 | 1900.000 |
| **2** | 1842.031 | 1855.576 | 1870.279 | 1881.090 | 1891.000 |
| **3** | 1828.610 | 1849.010 | 1863.646 | 1863.279 | 1864.090 |
| **4** | 1816.324 | 1826.520 | 1849.010 | 1863.646 | 1842.010 |

Table 3. $J_{\mu^*}^N(x,y)$ for all $(x,y) \in X$ in the deterministic domain; $N = 1966$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | DOWN | RIGHT | RIGHT | RIGHT | RIGHT |
| **1** | RIGHT | RIGHT | RIGHT | RIGHT | UP |
| **2** | UP | RIGHT | UP | UP | UP |
| **3** | UP | RIGHT | RIGHT | UP | UP |
| **4** | UP | RIGHT | UP | UP | LEFT |

Table 4. $\mu_N^*(x,y)$ for all $(x,y) \in X$, in the deterministic domain; $N = 1966$

## 3.3 Stochastic domain

In the stochastic domain, we have the following:

$$r(x,u) = wR(g,F(x,u)) + (1-w)R(g,(0,0)) \ \forall x \in X, u \in U$$

$$p(x' \mid x,u) = w(I_{\{x'=F(x,u)\}}) + (1-w)I_{\{x'=(0,0)\}} \quad \forall x,x' \in X, u \in U$$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 159.446 | 159.637 | 163.052 | 172.130 | 172.130 |
| **1** | 159.637 | 163.052 | 164.903 | 167.630 | 172.130 |
| **2** | 159.446 | 160.137 | 163.052 | 167.213 | 167.630 |
| **3** | 159.259 | 162.196 | 167.213 | 162.196 | 167.213 |
| **4** | 159.259 | 155.713 | 162.196 | 167.213 | 162.229 |

Table 5. $J^N_{\mu^*}(x,y)$ for all $(x,y) \in X$ in the stochastic domain; $N = 1966$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | DOWN | DOWN | DOWN | RIGHT | RIGHT |
| **1** | RIGHT | RIGHT | RIGHT | RIGHT | UP |
| **2** | UP | RIGHT | UP | DOWN | UP |
| **3** | LEFT | RIGHT | RIGHT | LEFT | LEFT |
| **4** | UP | RIGHT | UP | UP | RIGHT |

Table 6. $\mu^*_N(x,y)$ for all $(x,y) \in X$, in the stochastic domain; $N = 1966$

# 4 System Identification

In order to estimate $r(x,u)$ and $p(x'|x,u)$ from a given trajectory $h_t = (x_0, u_0, r_0, x_1, u_1, r_1, ..., u_{t-1}, r_{t-1}, x_t)$ one can compute it by

$$\hat{r}(x,u) = \frac{1}{|A_h(x,u)|} \sum_{i \in A_h(x,u)} r_i, \tag{6}$$

$$\hat{p}(x' \mid x,u) = \frac{1}{|A_h(x,u)|} \sum_{i \in A_h(x,u)} I_{\{x_{i+1}=x'\}}, \tag{7}$$

where $A_h(x,u)$ is the set of indices $\{i \mid x_i = x, u_i = u\}$. However, the number of operations to estimate the MDP structure grows linearly with t. And also the memory requirement we have therefore decided to implement the algorithm described at the slide 29 of the course.

At time 0, set $N(x,u) = 0$, $N(x,u,x') = 0$, $R(x,u) = 0$, $p(x'|x,u) = 0$, $\forall x, x' \in X$ and $u \in U$.
At time $t \neq 0$, do
1. $N(x_{t-1}, u_{t-1}) \leftarrow N(x_{t-1}, u_{t-1}) + 1$
2. $N(x_{t-1}, u_{t-1}, x_t) \leftarrow N(x_{t-1}, u_{t-1}, x_t) + 1$
3. $R(x_{t-1}, u_{t-1}) \leftarrow R(x_{t-1}, u_{t-1}) + r_t$
4. $r(x_{t-1}, u_{t-1}) \leftarrow \frac{R(x_{t-1}, u_{t-1})}{N(x_{t-1}, u_{t-1})}$
5. $p(x|x_{t-1}, u_{t-1}) \leftarrow \frac{N(x_{t-1}, u_{t-1}, x)}{N(x_t, u_t)} \quad \forall x \in X$

In order to deal with state action pairs $(x,u) \in X \times U$ that had not been visited and therefore have $N(x,u)$ equal to 0. We have decided to consider that they have a uniform probability to reach any cell of the domain. More formally that

$$p(x'|x,u) = \frac{1}{n * m} \forall x' \in X$$

We have then apply this algorithm from a given trajectory $h_t$ starting from $(0,3)$ and then using a random uniform policy. With increasing value of $t$ to show the convergence, $t = \{10^0, 10^1, 10^2, ...10^7\}$.

Note that in the following

$$J^{\hat{\mu}^*_N}(x) = \max_{u \in U} \hat{Q}_N(x,u)$$

## 4.1 Deterministic

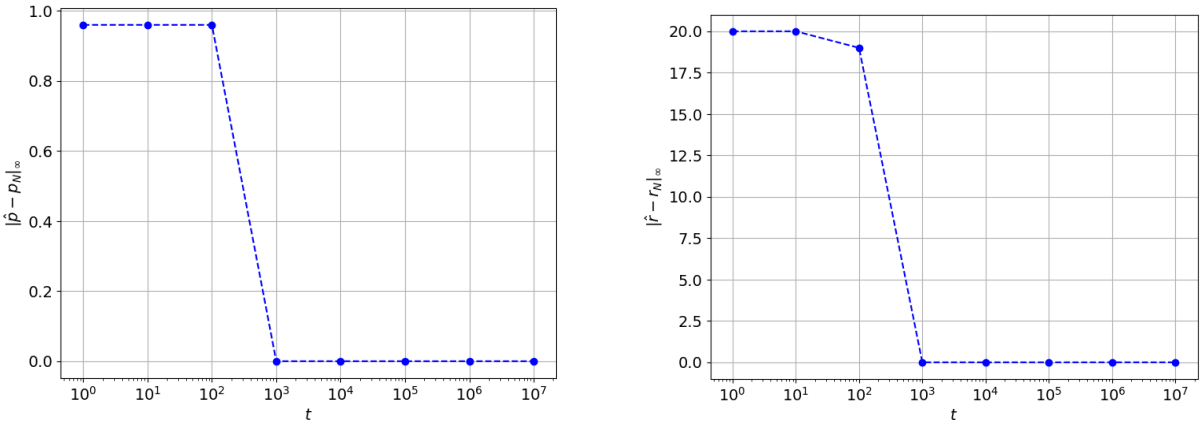



Figure 2. Convergence speed of $\hat{p}$ and $\hat{r}$ towards $p$ and $r$ using the $\infty$ norm through a growing trajectory generated by a random uniform trajectory in a deterministic domain

Clearly in the deterministic domain $\hat{p}$ and $\hat{r}$ converge after a trajectory length of $10^3$. Which are then use to compute the $\hat{Q}_N$-functions like the $Q_N$ (4) one by just replacing $r$ by $\hat{r}$ and $p$ by $\hat{p}$. Therefore if $\hat{p}$ and $\hat{r}$ has converged $\hat{Q}_N$ will also as we can see just below.

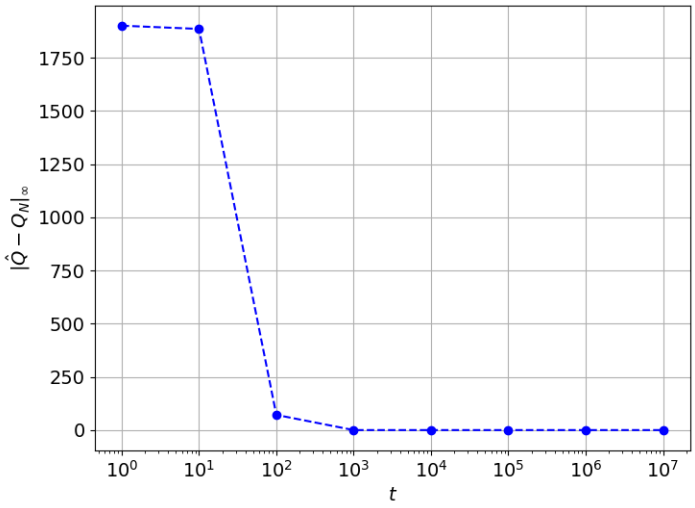



Figure 3. Infinity norm between $\hat{Q}_N$ and $Q_N$ for each trajectory length

The trajectory length $h_t$ had been computed once and then truncated according to each $t$.

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 1842.031 | 1857.190 | 1881.000 | 1900.000 | 1900.000 |
| **1** | 1854.576 | 1870.279 | 1881.090 | 1891.000 | 1900.000 |
| **2** | 1842.031 | 1855.576 | 1870.279 | 1881.090 | 1891.000 |
| **3** | 1828.610 | 1849.010 | 1863.646 | 1863.279 | 1864.090 |
| **4** | 1816.324 | 1826.520 | 1849.010 | 1863.646 | 1842.010 |

Table 7. $J^{\mu^*_N}(x, y)$ for all $(x, y) \in X$ in the deterministic domain; $N = 1966$

Using $\hat{Q}_N$ we can then approximate $\hat{\mu}^*_N$ such that $\hat{\mu}^*_N \in arg\ max_{u \in U}\ \hat{Q}_N(x, u)$. And in the following we computed $\hat{Q}_N$ with a trajectory length of $10^7$ steps.

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 1842.031 | 1857.190 | 1881.000 | 1900.000 | 1900.000 |
| **1** | 1854.576 | 1870.279 | 1881.090 | 1891.000 | 1900.000 |
| **2** | 1842.031 | 1855.576 | 1870.279 | 1881.090 | 1891.000 |
| **3** | 1828.610 | 1849.010 | 1863.646 | 1863.279 | 1864.090 |
| **4** | 1816.324 | 1826.520 | 1849.010 | 1863.646 | 1842.010 |

Table 8. $J^{\hat{\mu}^*_N}(x, y)$ for all $(x, y) \in X$ in the deterministic domain; $N = 1966$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | DOWN | RIGHT | RIGHT | RIGHT | RIGHT |
| **1** | RIGHT | RIGHT | RIGHT | RIGHT | UP |
| **2** | UP | RIGHT | UP | UP | UP |
| **3** | UP | RIGHT | RIGHT | UP | UP |
| **4** | UP | RIGHT | UP | UP | LEFT |

Table 9. $\hat{\mu}^*_N(x, y)$ for all $(x, y) \in X$ in the deterministic domain; $N = 1966$

.

We can see that as expected the result of Table 7. and Table 8. are indeed the same.

## 4.2 Stochastic

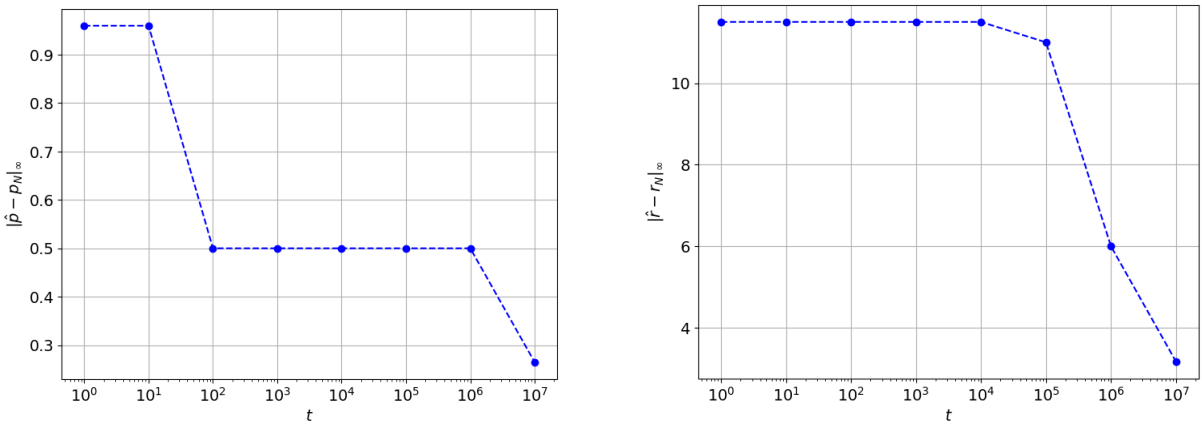



Figure 4. Convergence speed of $\hat{p}$ and $\hat{r}$ towards $p$ and $r$ using the $\infty$ norm through a growing trajectory generated by a random uniform trajectory in a stochastic domain

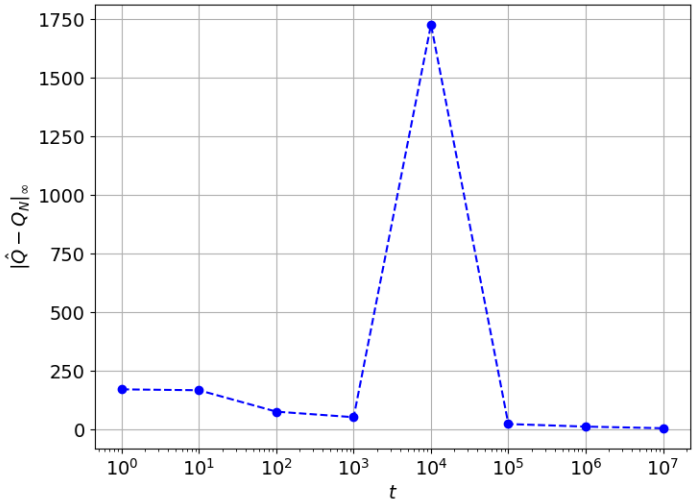



Figure 5. Infinity norm between $\hat{Q}_N$ and $Q_N$ for each trajectory length

Here $\hat{r}$, $\hat{p}$ and $\hat{Q}_N$ has not converged even after a trajectory length of $10^7$. As before in the following we computed $\hat{Q}_N$ with a trajectory length of $10^7$ steps.

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 159.446 | 159.637 | 163.052 | 172.130 | 172.130 |
| **1** | 159.637 | 163.052 | 164.903 | 167.630 | 172.130 |
| **2** | 159.446 | 160.137 | 163.052 | 167.213 | 167.630 |
| **3** | 159.259 | 162.196 | 167.213 | 162.196 | 167.213 |
| **4** | 159.259 | 155.713 | 162.196 | 167.213 | 162.229 |

Table 10. $J^{\mu^*_N}(x, y)$ for all $(x, y) \in X$ in the stochastic domain; $N = 1966$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 157.435 | 157.594 | 160.955 | 170.774 | 170.100 |
| **1** | 157.593 | 160.941 | 162.799 | 165.501 | 170.306 |
| **2** | 157.448 | 158.241 | 161.269 | 166.050 | 166.819 |
| **3** | 157.375 | 160.731 | 166.113 | 161.529 | 167.433 |
| **4** | 157.134 | 154.232 | 161.155 | 162.786 | 158.605 |

Table 11. $J^{\hat{\mu}^*_N}(x, y)$ for all $(x, y) \in X$ in the stochastic domain; $N = 1966$

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | DOWN | DOWN | DOWN | RIGHT | RIGHT |
| **1** | RIGHT | RIGHT | RIGHT | RIGHT | UP |
| **2** | UP | RIGHT | UP | DOWN | UP |
| **3** | LEFT | RIGHT | RIGHT | LEFT | LEFT |
| **4** | UP | RIGHT | UP | UP | UP |

Table 12. $\hat{\mu}^*_N(x, y)$ for all $(x, y) \in X$ in the deterministic domain; $N = 1966$

We can notice that, in the stochastic configuration, the convergence is way slower.

To explain the influence of the length of the trajectory on the quality of the approximations, we can compare the results with the deterministic domain and the stochastic domain.

It is trivial that the longer is the trajectory, the more likely it will contain a transition $(x, u, r) \to x'$ of the domain.

Furthermore, the strong law of large numbers states that the more samples we have (in our case, a sample is a transition $(x, u, r) \to x'$ of the domain), the closer we should get from the real mean of $r(.)$ and $p(.)$.

We can also explain the big difference of convergence between the deterministic domain and the stochastic domain:

In the deterministic domain, it is easier to reach any state $x'$ from a state $x$ than in the stochastic domain, because, at each transition in the stochastic configuration, there is a

probability of 0.5 to be teleported back to the state $(0, 0)$. Therefore, the probability of reaching states like the state $(4, 4)$ is very low, and so it is unlikely for a small trajectory to contain such states, and by the strong law of large number, moreover these state much be reach a certain amount of time in order to be close to the true value, while for the deterministic domain a single pass is enough to get the true value.

# 5 Q-learning in a batch setting

Non-deterministic numerical resuts are averaged through 10 independent runs and displayed with standard deviation. (Including when using an agent that behave randomly)

## 5.1 Offline Q-learning

Given $h_T = (x_0, u_0, r_0, x_1, u_1, r_1, ..., u_{T-1}, r_{T-1}, x_T)$, we can infer an approximate value of the Q-function without identifying the structure of a Markov Decision Process. The algorithm works as follows:

1. Initialize $\hat{Q}(x, u) = 0 \ \forall x \in X, u \in U$ and set $t = 0$
2. Take the one-step transition at time $t$ which is represented by the tuple $(x_t, u_t, r_t, x_{t+1})$, and apply the update rule :

$$\hat{Q}(x_t, u_t) = \hat{Q}(x_t, u_t) + \alpha(r_t + \lambda \underset{u \in U}{max} \hat{Q}(x_{t+1}, u) - \hat{Q}(x_t, u_t)) \tag{8}$$

   where $\alpha$ is the learning rate. In our case $\alpha = 0.05$ is a constant.
3. Move to the next one-step transition $t = t + 1$. If $t = T$, return $\hat{Q}$ and stop. Otherwise, go back to step 2.

Without any prior knowledge of the domain, it is not that easy to determine a good value of $T$, all we can say is that, thanks to the strong law of large numbers, the longest the trajectory is, the better we will get close to $Q$.

Fortunately, we know the domain definition. Therefore, we can compare $\hat{Q}$ with $Q_N$, computed in section 3. $Q_N$ being close to the real $Q$, we can approximately view how far $\hat{Q}$ is from $Q$ by computing $||\hat{Q} - Q_N||_\infty$.

The graphs below represent the convergence of $\hat{Q}$ towards $Q_N$, at each power of 10 number of steps up to $10^6$.

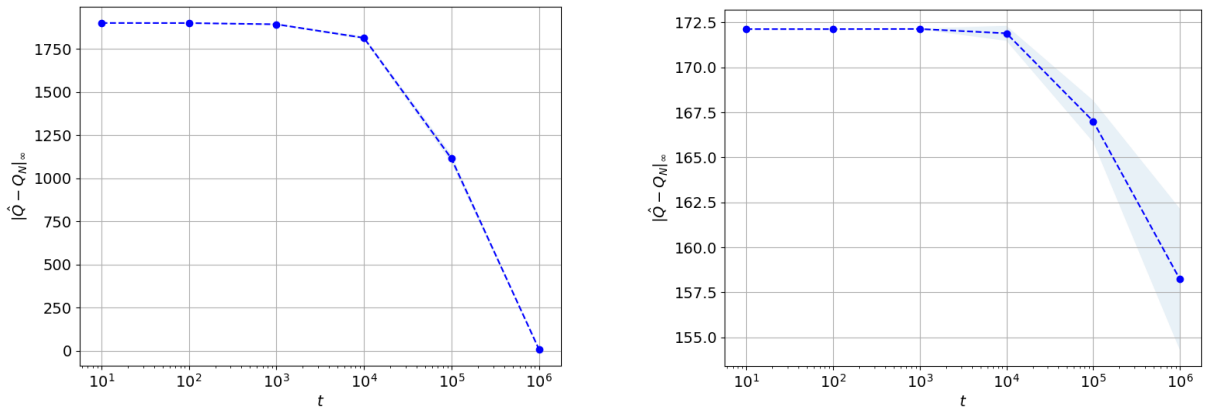



Figure 6. Convergence speed of $\hat{Q}$ towards $Q_N$ using the $\infty$ norm through a growing trajectory generated by a random uniform trajectory in the deterministic domain (left) and the stochastic domain (right)

We can notice that a trajectory size of $10^6$ is enough for the deterministic domain as $\hat{Q}$ converged to $Q_N$. However, in the case of the stochastic domain, it would need to perform more iterations, which has not been done because, unfortunately, the computing time is too long.

We can then infer $\hat{\mu}^* \in \underset{u \in U}{arg\,max} \hat{Q}(x,u)\ \forall x \in X$ using $T = 10^6$, shown in the table below:

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | DOWN | RIGHT | RIGHT | RIGHT | UP |
| **1** | RIGHT | RIGHT | RIGHT | RIGHT | UP |
| **2** | UP | RIGHT | UP | UP | UP |
| **3** | UP | RIGHT | RIGHT | UP | UP |
| **4** | UP | RIGHT | UP | UP | LEFT |

Table 13. $\hat{\mu}^*$ derived from the deterministic domain

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | DOWN | DOWN | DOWN | RIGHT | RIGHT |
| **1** | LEFT | RIGHT | RIGHT | LEFT | UP |
| **2** | DOWN | LEFT | UP | UP | UP |
| **3** | LEFT | LEFT | UP | UP | LEFT |
| **4** | UP | UP | UP | UP | LEFT |

Table 14. $\hat{\mu}^*$ from the stochastic domain

We can notice that, for the deterministic domain, we have that $\hat{\mu}^* = \mu_N^*$ except at the upper right corner: in this state, the actions UP and RIGHT lead to the same reward. Concerning the stochastic domain, this is logical that $\hat{\mu}^* \neq \mu_N^*$ as it did not converge.

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 1842.031 | 1857.190 | 1881.000 | 1900.000 | 1900.000 |
| **1** | 1854.576 | 1870.279 | 1881.090 | 1891.000 | 1900.000 |
| **2** | 1842.031 | 1855.576 | 1870.279 | 1881.090 | 1891.000 |
| **3** | 1828.610 | 1849.010 | 1863.646 | 1863.279 | 1864.090 |
| **4** | 1816.324 | 1826.520 | 1849.010 | 1863.646 | 1842.010 |

Table 15. $J^N_{\hat{\mu}^*}$ in the deterministic domain

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 139.165 | 138.851 | 142.368 | 152.251 | 152.251 |
| **1** | 138.946 | 141.342 | 142.375 | 142.400 | 151.228 |
| **2** | 139.164 | 139.099 | 141.862 | 141.875 | 143.594 |
| **3** | 139.188 | 139.285 | 139.608 | 138.381 | 138.459 |
| **4** | 139.285 | 131.833 | 134.200 | 136.270 | 135.419 |

Table 16. $J^N_{\hat{\mu}^*}$ mean in the stochastic domain

| $y$ \ $x$ | **0** | **1** | **2** | **3** | **4** |
|---|---|---|---|---|---|
| **0** | 8.082 | 7.866 | 7.640 | 7.922 | 7.922 |
| **1** | 8.245 | 7.852 | 7.981 | 8.143 | 7.102 |
| **2** | 8.159 | 7.956 | 7.938 | 8.158 | 6.036 |
| **3** | 8.006 | 7.962 | 7.927 | 7.952 | 12.142 |
| **4** | 7.962 | 7.941 | 9.827 | 13.399 | 11.491 |

Table 17. $J^N_{\hat{\mu}^*}$ standard deviation in the stochastic domain

We observe the same behavior for the expected cumulative reward.

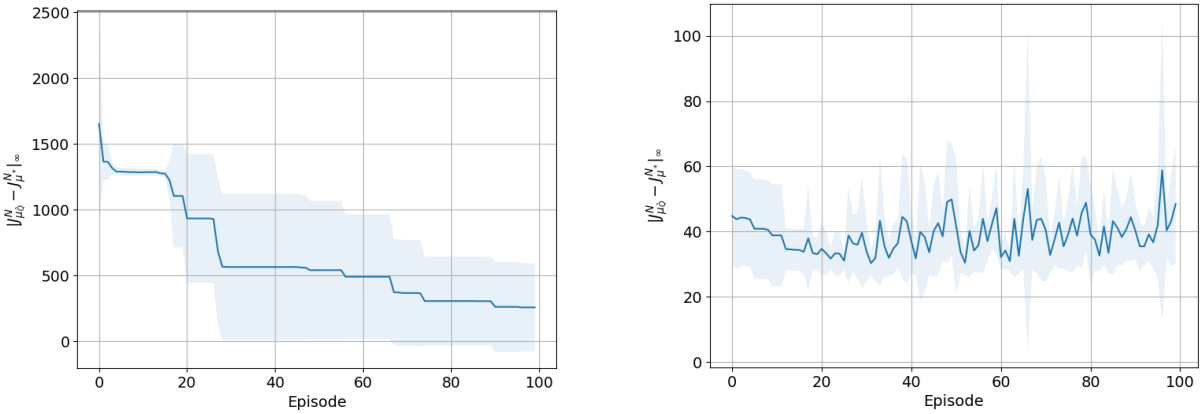
## 5.2 Online Q-Learning

For these experiments we needed to train our agent over 100 episodes having 1000 transitions each using an $\epsilon$-greedy policy. This policy selects with a probability $1 - \epsilon$ actions according to $\mu_{\hat{Q}}$ which is derived from $\hat{Q}$ and with a probability $\epsilon$ to choose action at random. $\hat{Q}$ is continuously updated at each transition using equation 8 .

This policy allows to address the dilemma between exploration and exploitation. For these experiments we will plot after each episode the value of $||J^N_{\mu_{\hat{Q}}} - J^N_{\mu^*}||_\infty$

### 5.2.1 Experiment 1

The first experiment is to set the learning rate $\alpha = 0.05$ and the exploration rate $\epsilon = 0.5$




(a) Deterministic domain

(b) Stochastic domain

Figure 7. Convergence speed of $J^N_{\mu_{\hat{Q}}}$ with $J^N_{\mu^*}$ across episodes, using the infinity norm.

In the deterministic setting, our agent converges slowly and has not converged yet, after 100 episodes. Indeed, to be close to the value of $J^N_{\mu^*}$, our agent would need to visit all the states a large number of times (infinite number of times). However, by using the $\epsilon$-greedy policy half of the time, the agent will perform actions that it knows, from experience, these will make him earn high rewards, and, in the other half, the agent will perform random actions allowing him to discover new (state, action, reward, next_state) tuples.

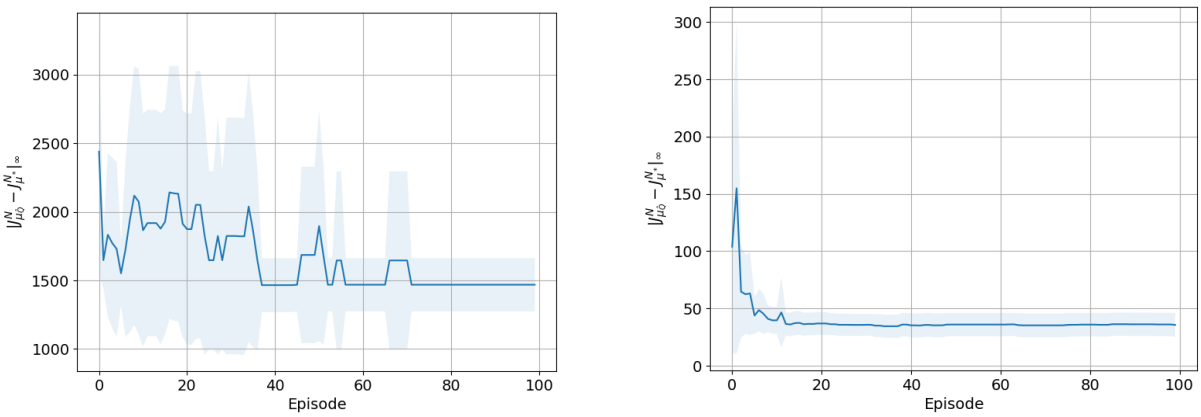
For the stochastic setting, the agent also struggles to converge to the value for the same reason as above, plus the fact that he may be teleported to (0, 0) with a probability of 0.5. It has the consequence that the agent has more difficulties to reach states that are far from the state (0, 0).

### 5.2.2 Experiment 2

The second experiment is exactly like the preceding one, but with a learning rate that evolves over time.

$$\alpha_t = 0.8\alpha_{t-1} \ \forall t > 0$$

with $\alpha_0 = 0.05$ Note that we reset the learning rate at each epoch.




(a) Deterministic domain (b) Stochastic domain

Figure 8. Convergence speed of $J^N_{\mu_{\hat{Q}}}$ with $J^N_{\mu^*}$ across episodes, using the infinity norm.

Note that 8 can be re-written as

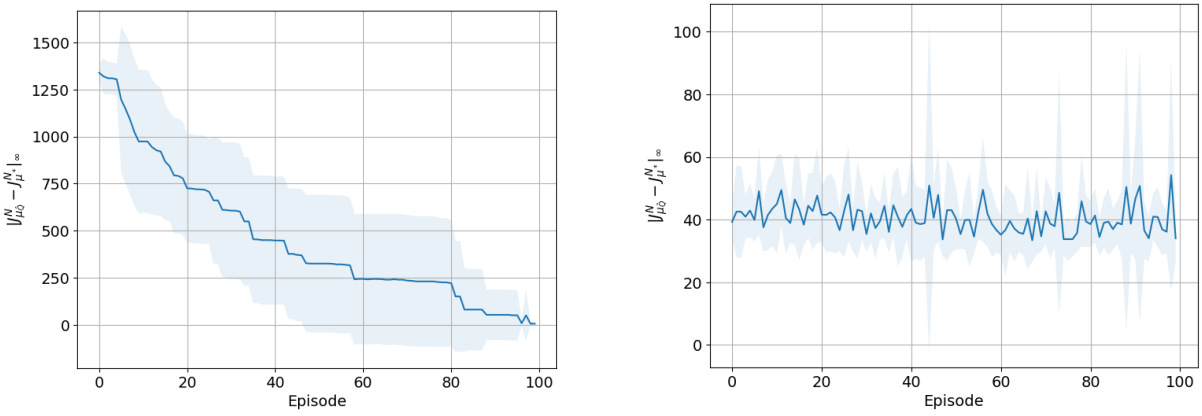
$$\hat{Q}(x_k, u_k) \leftarrow \hat{Q}(x_k, u_k) + \alpha\delta(x_k, u_k) \tag{9}$$

Where $\delta(x_k, u_k)$ is called the temporal difference and is defined as

$$\delta(x_k, u_k) = r_k + \lambda max_{u \in U}\hat{Q}(x_{k+1}, u) - \hat{Q}(x_k, u_k) \tag{10}$$

Using that notation, it is clear that if the learning rate $\alpha$ is really small, $\hat{Q}(x_k, u_k)$ will not change that much. It is exactly what happens here, as $\alpha$ decreases quite fast with respect to $t$ even if our agent visits all the states a large number of times, it will stop learning too early and converges into a local minimum.

### 5.2.3 Experiment 3



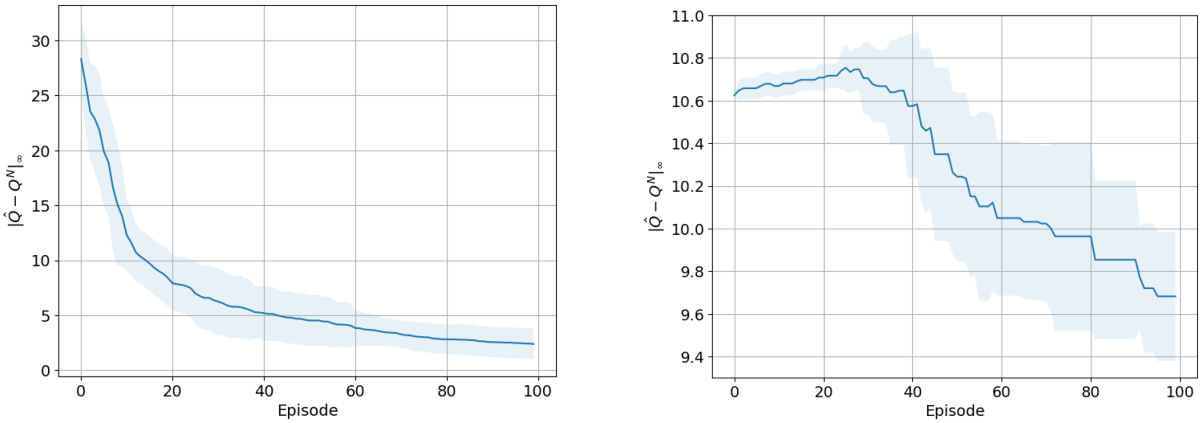

(a) Deterministic domain (b) Stochastic domain

Figure 9. Convergence speed of $J^N_{\mu_{\hat{Q}}}$ with $J^N_{\mu^*}$ across episodes, using the infinity norm.

For that experiment we use a replay buffer that store all previous transitions and at each time-step, the function $\hat{Q}$ is updated ten times by drawing ten transitions at random from the replay buffer.

It is interesting to see that our agent converge way more rapidly to the value of $J^N_{\mu^*}$ that before in the deterministic setting, the replay is quite important because it allow to replay some transition done earlier and to not forget about things that it learned before. However in the stochastic domain it's still struggle.

## 5.3 Discount factor




(a) Deterministic domain

(b) Stochastic domain

Figure 10. Convergence speed of $\hat{Q}$ toward $Q^N$ across episodes, using the infinity norm.

Overall it converge way much faster and easier than before (even for the stochastic), because using a discount factor of $\gamma = 0.4$ will reduce the impact of the farthest states on the optimal policy.

## 5.4 Q-learning with another exploration policy

Proposed in the article **"Learning-Driven Exploration for Reinforcement Learning"** by **Muhammad Usama** and **Dong Eui Chang**, we implemented an exploration policy called `Entropy based exploration`, where, unlike the greedy-policy who uses a constant $\epsilon$, it is state dependant of the entropy of the current state. Formally, we compute the probability distribution, which is a **Boltzman distribution**, as follows:

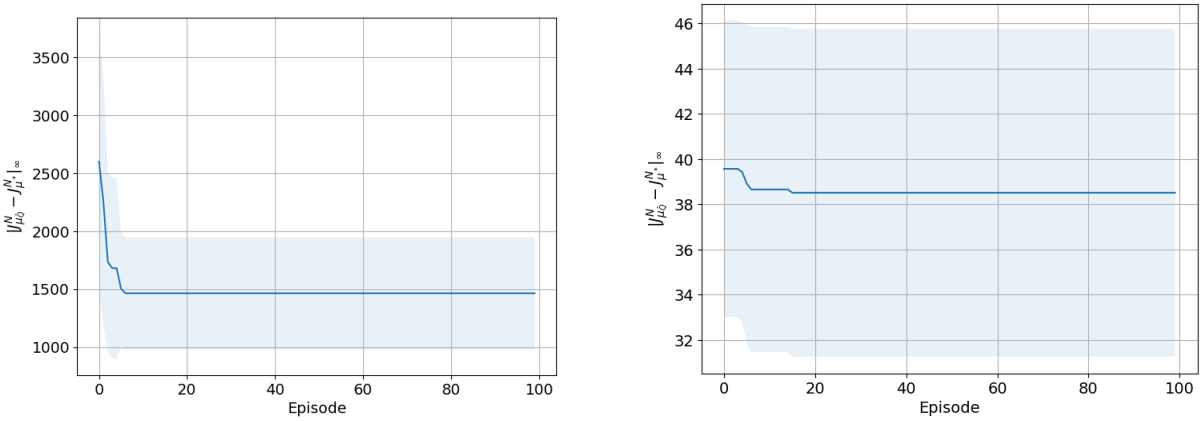
$$p(x,u) = \frac{e^{\hat{Q}(x,u) - max_{u' \in U} \hat{Q}(x,u')}}{\sum_{a \in U} e^{\hat{Q}(x,a) - max_{u' \in U} \hat{Q}(x,u')}}$$

Where $max_{u' \in U} \hat{Q}(x,u')$ is used in order to avoid numerical instability.

The entropy of a state $x$ is computed as follows:

$H(x) = \sum_{u \in U} p(x,u) log_{|U|} p(x,u)$

Ensuring that $H(x)$ is a value in $[0,1]$. This policy is non-parametric, so we do not have any parameters to tune.




(a) Deterministic domain

(b) Stochastic domain

Figure 11. Convergence speed of $J^N_{\mu_{\hat{Q}}}$ with $J^N_{\mu^*}$ across episodes, using the infinity norm.

We can notice that the results are quite unsatisfying as we do not converge to the optimal policy.