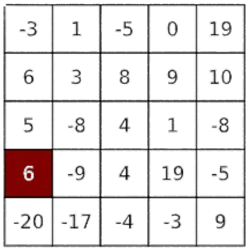
LIÈGE université
School of Engineering

INFO8003-1: Optimal Decision Making for Complex Problems



| | | | | |
|---|---|---|---|---|
| -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 |

# Assignment 1: Reinforcement Learning in a Discrete Domain (Part 2)

*Staff :*
ERNST Damien, *Teacher*
LOUETTE Arthur, *Teaching Assistant*
MIFTARI Bardhyl, *Teaching Assistant*

*Group :*
LAMBERMONT Romain
LOUIS Arthur

March 21, 2024

# Contents

# List of Figures

# List of Tables

# 5 Q-Learning in a Batch Setting

## 5.1 Offline Q-Learning

For this part of the project, we needed a function inside our `Q_Learning` class that was able to generate trajectories using an random uniform policy. This is implemented in the function `generate_trajectory` that also takes into account the wanted length of the trajectory, stochasticity of the environment as well as the initial state. As we re-implemented the fourth section that we were not able to deliver in the last milestone, we found out that a $T = 10^6$ was a good trade-off between performance and speed for our algorithms by looking at the infinite norm $\|Q_N - \hat{Q}_N\|_\infty$ and the computation times for different values of T.

Once we created this trajectory generating function, we implemented the routine seen during the course:

1. Initialisation of $\hat{Q}(s, a)$ to 0 everywhere. Set $k = 0$
2. $\hat{Q}(s_k, a_k) \leftarrow (1 - \alpha_k)\hat{Q}(s_k, a_k) + \alpha_k(r_k + \gamma \max_{a \in \mathcal{A}} \hat{Q}(s_{k+1}, a))$
3. $k \leftarrow k + 1$. If $k = t$, return $\hat{Q}$ and stop. Otherwise, go back to 2.

The only conditions that are needed to ensure the convergence of $\hat{Q}$ towards $Q$ when $t \to \infty$:

1. The learning ratio $\alpha$ must decrease but not to abruptly:
$$\lim_{n\to\infty} \sum_{k=0}^{t-1} \alpha_k = \infty \text{ and } \lim_{n\to\infty} \sum_{k=0}^{t-1} \alpha_k^2 < \infty$$
2. The trajectory $h_t$ needs to visit every state-action pair an infinite number of times when its length tends to infinity.

Now that the routine is implemented in the function `offline_learning` of our class `Q_Learning`, we can apply it to both the environments. The policies extracted from the Q-Learning algorithm for the deterministic and stochastic environments can be seen in Tables 1 and 3. The expected return of these policies can be seen respectively in Tables 2 and 4.

| | | | | |
|---|---|---|---|---|
| ↓ | → | → | → | ↑ |
| → | → | → | → | ↑ |
| ↑ | → | ↑ | ↑ | ↑ |
| ↑ | → | → | ↑ | ↑ |
| ↑ | → | ↑ | ↑ | ← |

Table 1: Offline Q-Learning Policy for the Deterministic Domain

| | | | | |
|---|---|---|---|---|
| 1842.030546 | 1857.190000 | 1881.000000 | 1900.000000 | 1900.000000 |
| 1854.576309 | 1870.279100 | 1881.090000 | 1891.000000 | 1900.000000 |
| 1842.030546 | 1855.576309 | 1870.279100 | 1881.090000 | 1891.000000 |
| 1828.610240 | 1849.009846 | 1863.646309 | 1863.279100 | 1864.090000 |
| 1816.324138 | 1826.519747 | 1849.009846 | 1863.646309 | 1842.009846 |

Table 2: Expected Return of the Offline Q-Learning for the Deterministic Domain

| | | | | |
|---|---|---|---|---|
| ↓ | ↓ | ↓ | → | → |
| ← | → | → | ← | ↑ |
| ↑ | ↑ | ↑ | ↑ | ← |
| ↑ | ← | ↑ | ← | ← |
| ↑ | ↑ | ↑ | ← | ← |

Table 3: Offline Q-Learning Policy for the Stochastic Domain

| | | | | |
|---|---|---|---|---|
| 600.000 | 599.284943 | 604.616047 | 625.742574 | 625.742574 |
| 600.000 | 604.616047 | 605.284943 | 604.616047 | 625.742574 |
| 600.000 | 599.284943 | 604.616047 | 605.284943 | 597.616047 |
| 599.000 | 599.505000 | 600.284943 | 598.141047 | 612.079818 |
| 599.505 | 584.754975 | 598.141047 | 589.079818 | 585.594510 |

Table 4: Expected Return of the Offline Q-Learning Policy for the Stochastic Domain

## 5.2 Online Q-Learning

### 5.2.1 Experimental Protocols

In our experimental setup, we implemented an intelligent agent utilizing Q-learning with an $\epsilon$-greedy policy, where the policy is greedy with a probability of $(1-\epsilon)$ and random otherwise. A replay buffer, defined as a data structure of previously seen transitions $(s_t, a_t, r_t, s_{t+1})$, was employed.

1. **First experimental protocol**: The agent underwent training across 100 episodes, each consisting of 1000 transitions. Each episode began from the initial state $s_0 = (3, 0)$. The learning rate $(\alpha)$ was set to 0.05, and the exploration rate $(\epsilon)$ to 0.5, with both values held constant over time. The function $\hat{Q}$ was updated after every transition, with transitions utilized only once for updating $\hat{Q}$.

2. **Second experimental protocol**: This protocol mirrored the first one, except for the learning rate. Here, $\alpha_0 = 0.05$, and $\forall t > 0, \alpha_t = 0.8\alpha_{t-1}$, signifying an adaptive learning rate scheme. The evolution of the learning rate can be see in Figure 1.

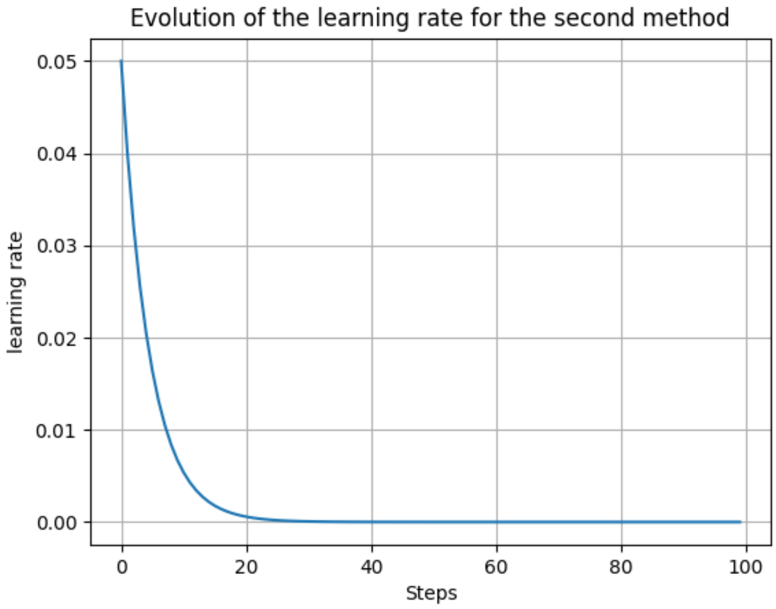


Figure 1: Evolution of the learning rate following the second experimental protocol

3. **Third experimental protocol**: Similar to the first one, but with modifications. One-step system transitions were stored in a replay buffer, and at each time-step, the function $\hat{Q}$ was updated ten times by drawing ten transitions randomly from the replay buffer.

We ran all three experimental protocols and recorded, after each episode, the value of $\|J_N^{\mu^{\hat{Q}}} - J_N^{\mu^*}\|_\infty$, where $\mu^{\hat{Q}}$ represents the policy derived from $\hat{Q}$, and compared the results.

#### 5.2.2 Deterministic case

Here's a summary of the performance of the three methods in the deterministic case:

1. **First method**: Converges to 40 in 60 steps, as seen in Figure 2.
2. **Second method**: Converges to 95 in 30 steps, as seen in Figure 4.
3. **Third method**: Converges to 1 in 40 steps, as seen in Figure 6.

In terms of convergence speed:

- The second method is the fastest, converging in only 30 steps.
- The third method is slower than the second but still faster than the first, converging in 40 steps.
- The first method is the slowest, requiring 60 steps to converge.

In terms of precision:

- The third method achieves the highest precision, estimating the true expected return almost perfectly.
- The second method sacrifices precision for speed, resulting in a less accurate estimate.
- The fist method is the simplest but seems to work pretty good since the result is not the best but also not the worst.

#### 5.2.3 Stochastic case

Here's a summary of the performance of the three methods in the stochastic case:

1. **First method**: Converges to 56 in 80 steps, as seen in Figure 3.
2. **Second method**: Converges to 27.5 in 40 steps, as seen in Figure 5.
3. **Third method**: Converges to 56.5 in 17 steps, as seen in Figure 7.

In terms of convergence speed compared to the first method:

- The second method achieves convergence twice as fast and with twice the precision.
- The third method achieves convergence much faster, but does not improve upon the performance of the first method.

In terms of precision:

- The third method achieves the highest precision, reaching a value close to the true expected return.
- The second method achieves faster convergence but sacrifices precision compared to the third method.
- The first method, while not the fastest or most precise, performs reasonably well, with its result in the same order of the third method.

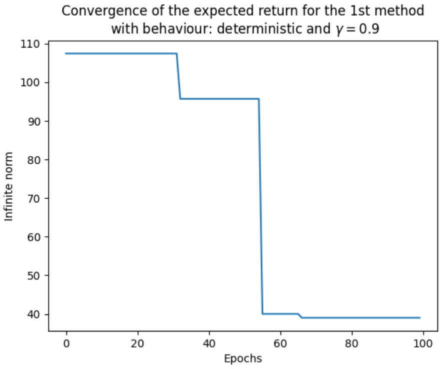


Figure 2: $\|J_N^{\mu_{\hat{Q}}} - J_N^{\mu^*}\|_\infty$ for the first method in the deterministic case

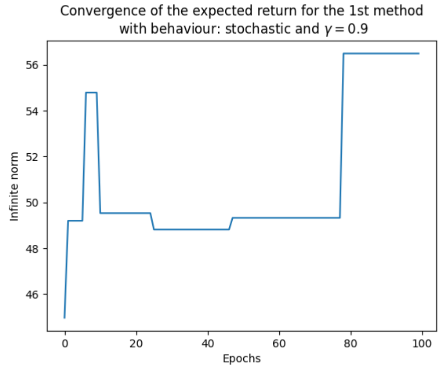


Figure 3: $\|J_N^{\mu_{\hat{Q}}} - J_N^{\mu^*}\|_\infty$ for the first method in the stochastic case

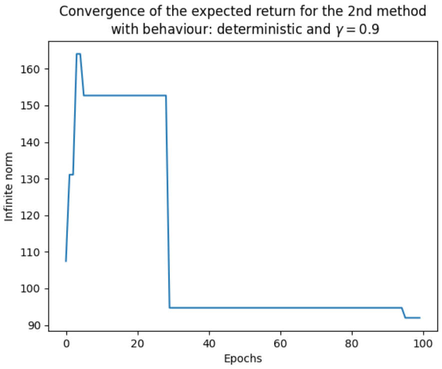


Figure 4: $\|J_N^{\mu_{\hat{Q}}} - J_N^{\mu^*}\|_\infty$ for the second method in the deterministic case

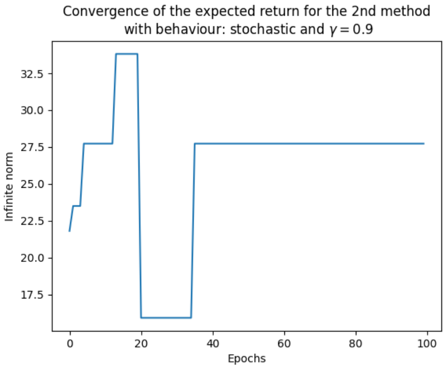


Figure 5: $\|J_N^{\mu_{\hat{Q}}} - J_N^{\mu^*}\|_\infty$ for the second method in the stochastic case

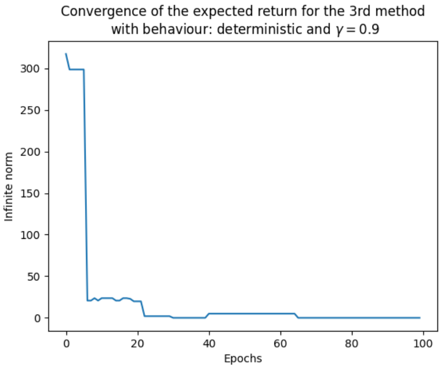


Figure 6: $\|J_N^{\mu_{\hat{Q}}} - J_N^{\mu^*}\|_\infty$ for the third method in the deterministic case

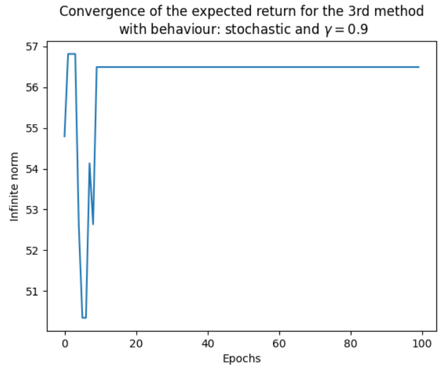


Figure 7: $\|J_N^{\mu_{\hat{Q}}} - J_N^{\mu^*}\|_\infty$ for the third method in the stochastic case

## 5.3 Discount Factor

### 5.3.1 Approximation of the Q Function with $\gamma = 0.4$ in the Deterministic Case

In this section, we discuss the approximation of the Q function with a discount factor $\gamma = 0.4$, using the same three experimental protocols as before.

1. **First method**: The first method converges to 4 within 100 steps, as seen in Figure 8.
2. **Second method**: The second method converges to 30.6 within 100 steps, as seen in Figure 10.
3. **Third method**: The third method converges to 0 within 2 steps, as seen in Figure 12.

In terms of convergence speed:

- The third method demonstrates the fastest convergence, achieving the target value in just 2 steps.
- The first and second methods have not yet fully converged within the specified number of steps.

In terms of precision:

- The third method achieves high precision, reaching the target value of 0 within a very small number of steps.
- The first and second methods, while not yet fully converged, demonstrate promising precision by approaching their respective target values within the given number of steps.

These results provide insights into the performance of each method in approximating the Q function under the given discount factor in the deterministic case.

#### 5.3.2 Approximation of the Q Function with $\gamma = 0.4$ in the Stochastic Case

In this section, we discuss the approximation of the Q function with a discount factor $\gamma = 0.4$, in the stochastic case.

1. **First method**: The first method converges to 21 within 2 steps, as seen in Figure 9.
2. **Second method**: The second method seems to approach infinity but reaches at 18 by step 100, as seen in Figure 11.
3. **Third method**: The third method reaches 21 within 3 steps, as seen in Figure 13.

In terms of convergence speed:

- The first method demonstrates very fast convergence, achieving the target value in just 2 steps.
- The third method also shows rapid convergence, reaching the target value within 3 steps.
- The second method appears to converge slowly and does not seem to reach the target value within the specified number of steps.

In terms of precision:

- The first and third methods achieve high precision, converging to the target value of 21 within a small number of steps.
- The second method, while not reaching the target value, stabilizes at 18, indicating some level of precision.

These results provide insights into the performance of each method in approximating the Q function under the given discount factor in the stochastic case.

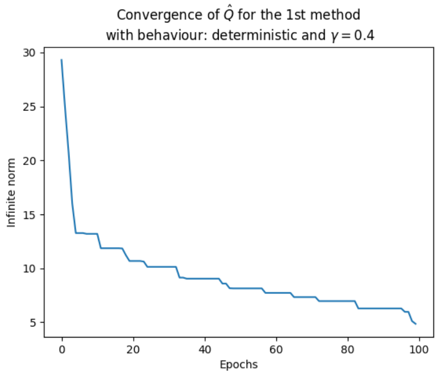


Figure 8: $\|\hat{Q} - Q\|_\infty$ for the first method in the deterministic case

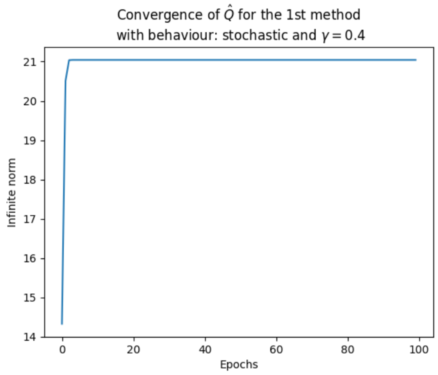


Figure 9: $\|\hat{Q} - Q\|_\infty$ for the first method in the stochastic case

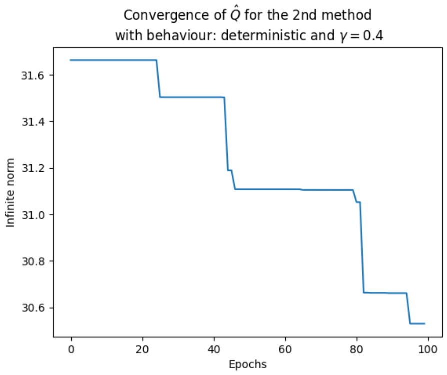


Figure 10: $\|\hat{Q} - Q\|_\infty$ for the second method in the deterministic case

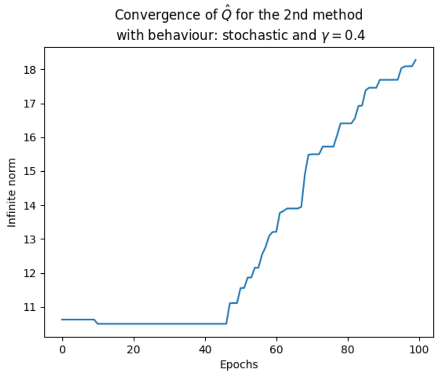


Figure 11: $\|\hat{Q} - Q\|_\infty$ for the second method in the stochastic case

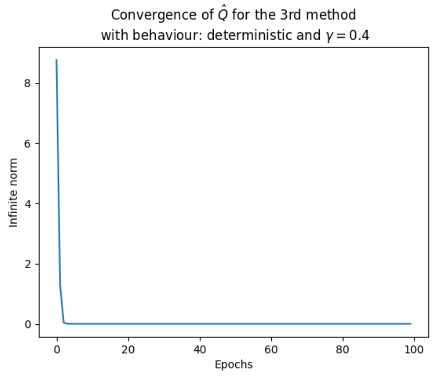


Figure 12: $\|\hat{Q} - Q\|_\infty$ for the third method in the deterministic case

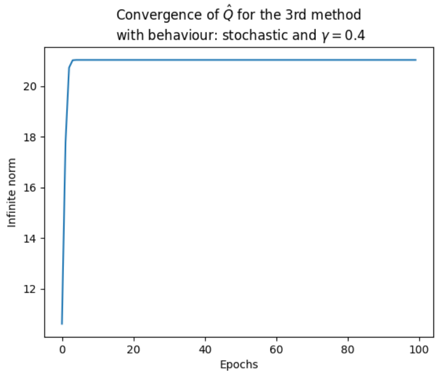


Figure 13: $\|\hat{Q} - Q\|_\infty$ for the third method in the stochastic case