Master MVA
Reinforcement Learning

TP 1: Dynamic Programming and Reinforcement Learning

Victor Busa
victor.busa@ens-paris-saclay.fr

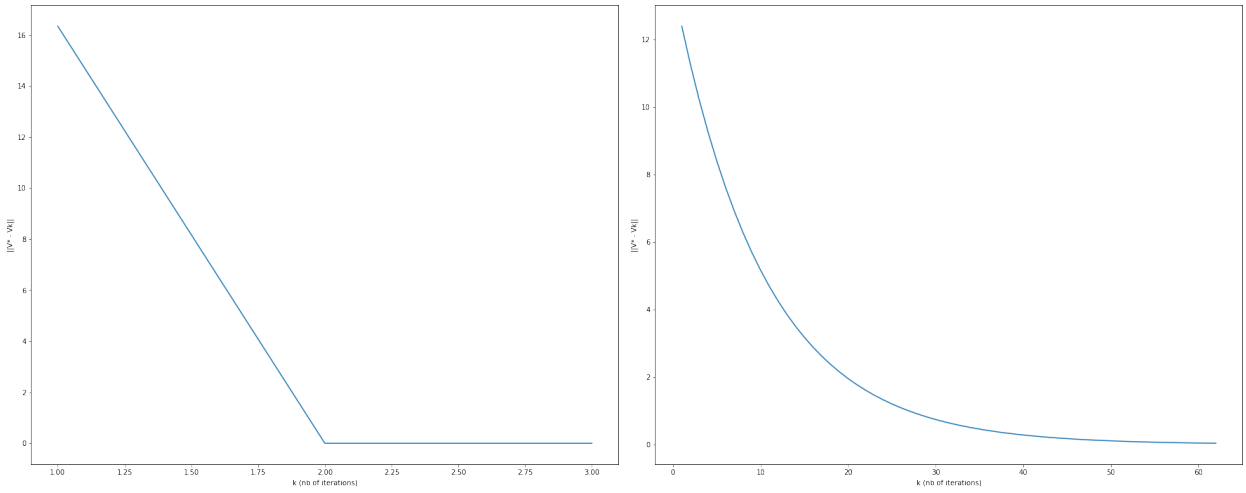
November 11, 2017

1 Dynamic Programming

Q1 See code

Q2 See code

Q3 For Policy Iteration, I used the Bellman operator $\mathcal{T}$ to evaluate the optimal policy. The Figure 1.1 depicts the error rates for both Value Iteration and Policy Iteration.



(a) Policy Iteration (b) Value Iteration

Figure 1.1: error rates $||V^* - V_k||_\infty$ in function of the number of iterations

Policy Iteration converges faster than Value iteration. In our experiment, Policy iteration converges in only **2 iterations** while Value Iteration needed **60 iterations**. However, each Policy iteration step is much more costlier than Value Iteration as, for each step of Policy Iteration, the algorithm needs to evaluate the current policy.

2 Reinforcement Learning

2.1 A Review of RL Agent/Environment Interaction

Q4 Let $J_n = \sum_{x \in \mathcal{X}} \mu_0(x) V_n(x)$, and $J^\pi = \sum_{x \in \mathcal{X}} \mu_0(x) V^\pi(x)$ where μ_0 is the uniform distribution, then, using first-visit Monte-Carlo method, we can plot $||J_n - J^\pi||$ in function of the number of iterations. Here I choose E (the number of Episode) to be **1000** and T_{max} the maximum of step in a trajectory to be **100**. We can see that $J_n - J_\pi$ quickly converges towards 0 which is what we want.

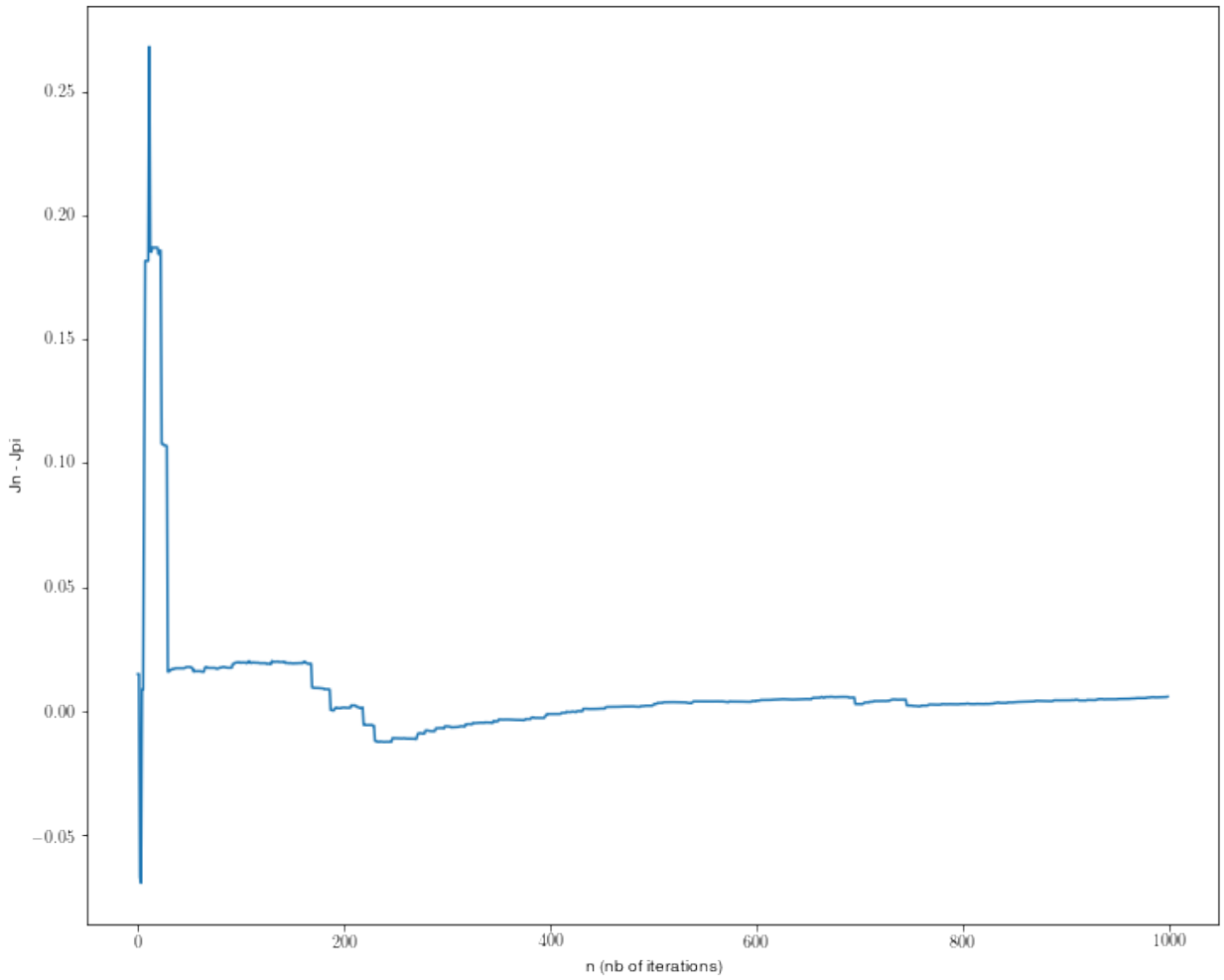


Figure 2.1: $J_n - J^\pi$ in function of the number of iterations using first-visit Monte-Carlo

Q5 I implemented Q-learning algorithm using and ϵ-greedy policy with $\epsilon = 0.1$. I set E, the number of episodes to be **20000**, T_{max}, the maximum number of step per trajectory to be **100**, and I used the learning rate:

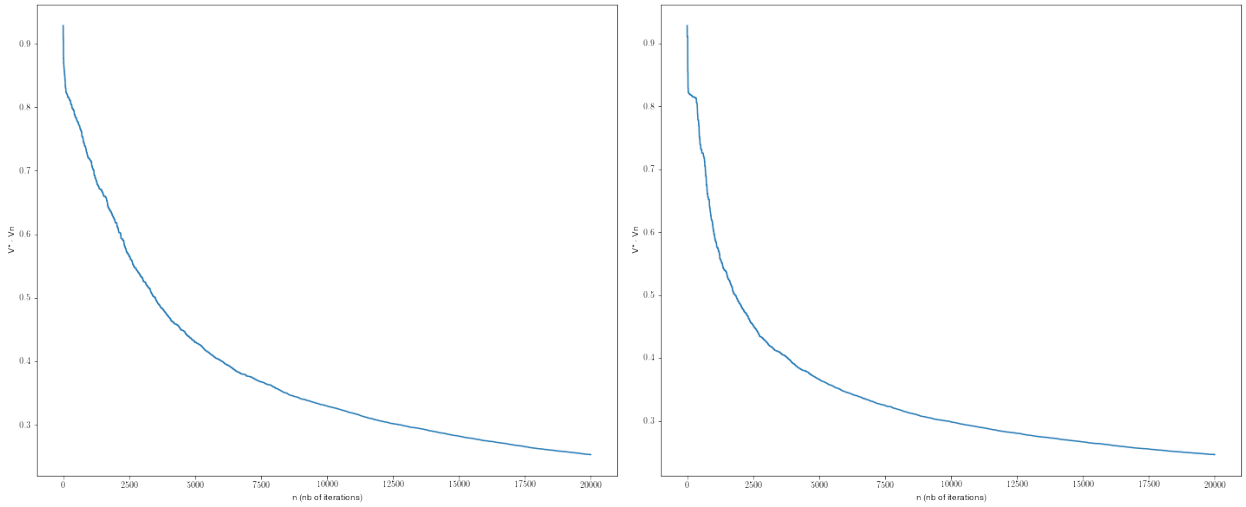
$$\alpha_i(x, a) = \frac{1}{N_i(x, a)}$$

where $N_i(x, a)$ is the number of time we visited the state-action pair (x, a) up to iteration i. This

choice satisfies the Robbins-Monro conditions:

$$\sum_i \alpha_i(x,a) = \sum_i 1/i = +\infty \quad \text{and} \quad \sum_i \alpha_i^2(x,a) = \frac{\pi^2}{6} < +\infty$$

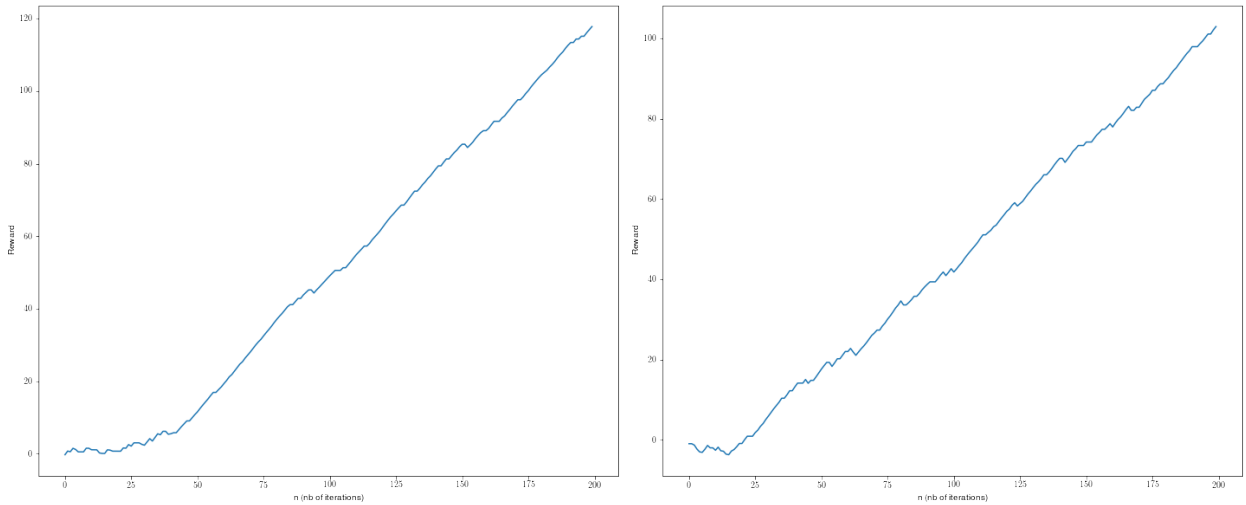
Figure 2.2 shows the error rates in function of the number of iterations for $\epsilon = 0.1$ and $\epsilon = 0.25$



(a) Q-learning with $\epsilon = 0.1$ (b) Q-learning with $\epsilon = 0.25$

Figure 2.2: error rates $||V^* - V^{\pi_n}||_\infty$ in function of the number of iterations

Figure 2.3 displays the cumulated reward in function of the number of iterations (Here $E = 200$).



(a) Q-learning with $\epsilon = 0.1$ (b) Q-learning with $\epsilon = 0.25$

Figure 2.3: Cumulated reward in function of the number of iterations

The previous Figures match our intuition. During the first iteration of the algorithm Q-learning doesn't have a very good idea of the optimal policy and so the cumulated reward will oscillate

and not increase much. Then after a few iteration the agent have explored most of the state and have a good idea of the action it should do in every step and hence the curve of the cumulated reward will be linear as the agent will most of the time retrieve the same nearly optimal reward. We can also notice that the curve still oscillate a bit. This is due to the epsilon greedy policy. A good idea would be to decay the epsilon with the number of episode. I also added in Table 1, the approximated optimal policy computed using Q-learning for different number of iterations. Here I choose $T_{max} = 100$.

Iterations	**Approximated optimal policy**
10^2	[0.645, 0.869, 0.963, 0, 0.276, 0.898, 0, 0.420, 0.678, 0.836, 0.481]
10^3	[0.832, 0.907, 0.982, 0, 0.748, 0.905, 0, 0.660, 0.584, 0.815, 0.430]
10^4	[0.880, 0.930, 0.988, 0, 0.827, 0.921, 0, 0.779, 0.715, 0.847, 0.575]
10^5	[0.876, 0.928, 0.988, 0, 0.817, 0.926, 0, 0.763, 0.805, 0.871, 0.790]
10^6	[0.876, 0.928, 0.988, 0, 0.822, 0.928, 0, 0.775, 0.820, 0.876, 0.823]
$v*$	[0.877, 0.928, 0.988, 0, 0.824, 0.928, 0, 0.778, 0.824, 0.877, 0.828]

Table 1: Approximated optimal policy computed for various number of iteration using Q-learning

Q6 The optimal policy of an MDP is not affected by the change of the initial distribution. This is due to the fact that:

- We explore all the states due to the ϵ-greedy policy
- the **optimal action** that should be chosen in a state is independent of the number of times we ended in that state