Chaouki_Ayman_TP1

November 12, 2018

1 Dynamic Programming

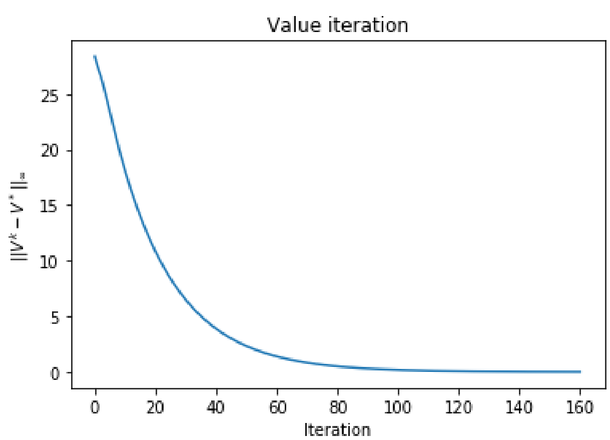
1.0.1 Q1: Implementing the discrete MDP model

There are no rewards in state s_1, therefore we want to avoid being stuck in it. We have $r(s_2, a_2) = \frac{9}{10}$, which is a good reward, thus we want to get stuck in state s_2, thus we take:

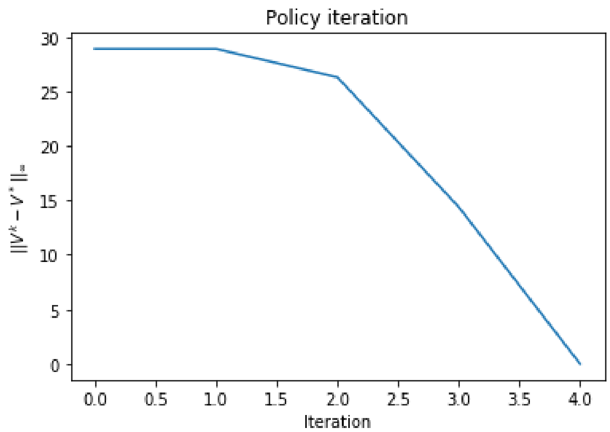
$$\begin{cases} \pi^*(s_0) = a_1 \\ \pi^*(s_1) = a_1 \\ \pi^*(s_0) = a_2 \end{cases}$$

1.0.2 Q2 : Value iteration

We want a 0.01-optimal policy, therefore we take $\frac{2\epsilon\gamma}{1-\gamma} = 0.01$ where ϵ is the stopping criterion of the algorithm.



1.0.3 Q3 : Policy iteration



Timing value iteration and computing the number of iterations it takes until convergence.

```
6.78 ms ± 1.18 ms per loop (mean ± std. dev. of 7 runs, 100 loops each)
```

```
number of iterations taken by value iteration :  160
```

Timing policy iteration and computing the number of iterations it takes until convergence.

```
370 µs ± 49 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
```

```
number of iterations taken by policy iteration :  4
```

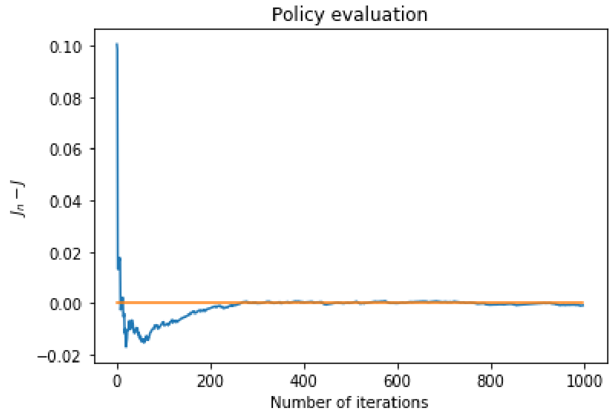
In this example, policy iteration is much faster to converge than value iteration and it takes much less iterations. The problem with policy iteration is that it requires a full policy evaltion in each step, thus if the state space is huge this step becomes expensive and value iteration may become the preferable option.

2 Reinforcement Learning

2.0.1 Q4 : Policy evaluation

We take $T_{max} = 140$ to get an approximation of aroun 10^{-3} and for each state we simulate 1000 trajectories to estimate the optimal value function.

```
Out[13]: array([ 0.87511567,  0.92961951,  0.98980574,  0.        ,  0.66599418,
                -0.99451987,  0.        , -0.82820401, -0.87698198, -0.93385322,
                -0.994725  ])
```



2.0.2 Q5 : Policy optimization

Assume we have some state-action function $Q_t(x, a)$, and we are in the current state $x_t \in \mathcal{X}$. We choose the next state x_{t+1} according to an ϵ-greedy policy, that is we choose $a_t = argmax_{a \in \mathcal{A}} Q_t(x_t, a)$ with probability $1 - \epsilon$ otherwise we choose an action from the rest with a uniform distribution. Now that we have simulated the transition (x_t, a_t, r_t, x_{t+1}), we can compute the temporal difference:

$$\delta_t = r_t + \gamma . max_{a \in \mathcal{A}} \left(Q_t \left(x_{t+1}, a \right) \right) - Q_t \left(x_t, a_t \right) \tag{1}$$

Then we can update the estimate:

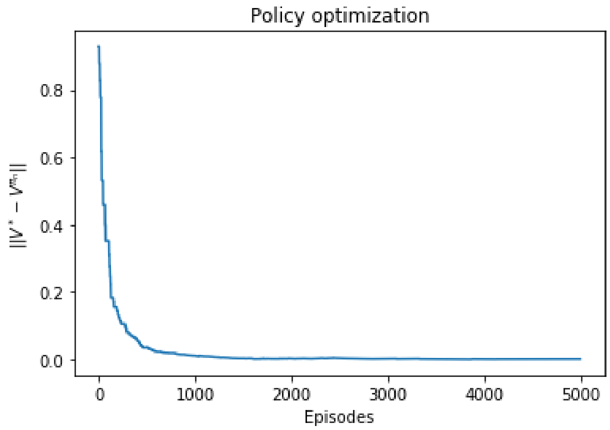
$$Q_{t+1} \left(x_t, a_t \right) = Q_t \left(x_t, a_t \right) + \alpha_{N(x_t, a_t)} \delta_t \tag{2}$$

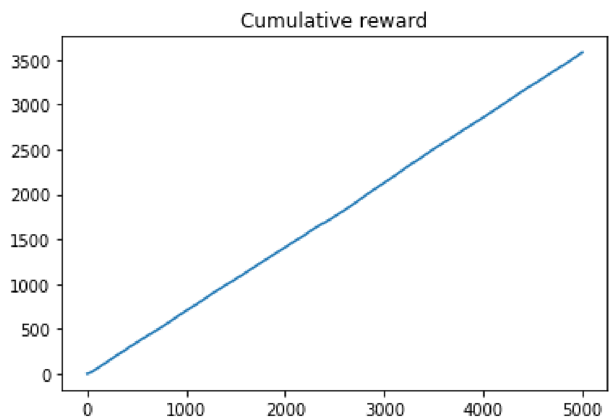
Where $N \left(x_t, a_t \right)$ is the number of times we visited state-action (x_t, a_t), by taking $\alpha_n = \frac{1}{n^\beta}$ where β is a decay coefficient in $]0.5, 1]$, the learning rate satisfies Robbins-Monro conditions:

$$\begin{cases} \sum_n \alpha_n = \infty \\ \sum_n \alpha_n^2 < \infty \end{cases}$$

but in order for the Q-learning algorithm to converge, all states-actions must be visited infinitly often. This is relying on the quality of the exploratory policy. In this case, a good choice of ϵ.

Note: We may need to decrease ϵ to 0 over time with a cooling schedule like SARSA algorithm does with its exploratory policy.
The algorithm we will run uses $\epsilon = 0.3$ with a cooling schedule $\epsilon(x, a) = \frac{1}{\epsilon + N(x,a)}$ where $N(x, a)$ is the number of visits to the state-action (x, a) and in the same fashion $\alpha(x, a) = \frac{1}{N(x,a)^\beta}$ with $\beta = 0.8$





Q-learning algorithm does converge well to the optimal value function and optimal policy.

2.0.3 Q6 :

The initial distribution μ_0 does influence the estimation of the optimal policy, **But not the true optimal policy since it is intrinsic to the MDP**.

- If μ_0 always gives an absorbant state, no trajectories will be produced and thus Robbins-Monro condition of visiting each state-action infinitely often does not hold.
- If the MDP simulator is deterministic with respect to some state-action, if μ_0 is poorly chosen, there is a risk of not exploring all the states. There will be a need for a very good exploratory policy, therefore μ_0 does influence the estimation of the optimal policy.