EC: Reinforcement Learning **Lecture: 1**

Homework: Approximate Dynamic Programming

Lecturer: *Alessandro Lazaric* *http://researchers.lille.inria.fr/∼lazaric/Webpage/Teaching.html*

Deadline: March 23, midnight

1 The Mountain Car Problem

Description of the Problem

In the mountain car problem, a car is stuck in a valley and the goal is to escape by climbing up the right side. We assume that the gravity is stronger than the cars engine, and even at full throttle the car cannot accelerate up the steep mountain road. Thus the only solution is to first move away from the goal and up opposite slope on the left. Then by applying full throttle the car can build up enough inertia to carry it up the steep slope. This problem can be modeled as finding the optimal policy in the following MDP:

- The state space $S = [-1.2; 0.6] \times [-0.07; 0.07]$ where the first dimension represent the position and the second the velocity of the car.
- The action space $A = -1, 1$ where 1 means full throttle forward, 1 for full throttle reverse.
- The reward is always $r(s, a, s) = r(s) = -1$ except when the position is 0.5. The position 0.5 is a terminal state.
- The dynamic of the system (transition) is described as follows. Assume that you are in state $s_t = (x_t, v_t)$ and you did action a. Then you move to the next state $s_{t+1} = (x_{t+1}, v_{t+1})$ given by :

$$\begin{aligned} v_{t+1} &= \max\big\{\min\{v_t + \epsilon_t + 0.001a_t - 0.0025cos(3x_t), 0.07\}, -0.07\big\} \\ x_{t+1} &= \max\big\{\min\{x_t + v_t, 0.6\}, -1.2\big\}, \end{aligned}$$

  with the extra condition that the position $x = -1.2$ is an inelastic wall: if the car reaches this point, its speed is set to zero.

One can consider both the deterministic case in which $\epsilon = 0$ and the stochastic case in which ϵ is drawn uniformly at random in $[-0.0005, 0.0009]$. This problem is a reinforcement learning problem since the driver is in general not aware of the precise dynamics of the problem and the transition can be regarded as unknown. Note that even in the planning problem (where you know the transition, as we do), the usual algorithms (VI, PI) cannot be applied since the state space is infinite (or very big if discretized). The goal of the TP is to compute and plot the optimal value function V .

Requirements for the Homework

Use the files provided in code.zip for the structure of the homework.

- Implement the test_policy function and use it to test the three baseline policies.

- Implement the LSTD function to evaluate any fixed policy. The objective is to have an approximation of its Q-value over the whole state-action space. In case, you can test the quality of the approximation by running Monte-carlo simulations in some states.
- **BONUS**: Implement the linear fitted Q-iteration algorithm and use the features provided by my_featureQ. Verify its quality by comparing its performance with the hand-coded policy.