ULiège
Prof. Damien Ernst
TA : Samy Aittahar

INFO8003-1
Optimal decision making
for complex problems
11th February 2018

Assignment 2

Reinforcement Learning in a Continuous Domain

1 Domain

We describe the domain below :

- State space : $X = \{(p, s) \in \mathbb{R}^2 | |p| \leq 1, |s| \leq 3\}$ and a *terminal state*[1].
  - A terminal state is reached if $|p_{t+1}| > 1$ or $|s_{t+1}| > 3$.
- Action space : $U = \{4, -4\}$.
- Dynamics : $\dot{p} = s$, $\dot{s} = \frac{u}{m(1+Hill'(p)^2)} - \frac{gHill'(p)}{1+Hill'(p)^2} - \frac{s^2 Hill'(p)Hill''(p)}{1+Hill'(p)^2}$, where $m = 1$, $g = 9.81$ and

$$Hill(p) = \begin{cases} p^2 + p & \text{if} \quad p < 0 \\ \frac{p}{\sqrt{1+5p^2}} & \text{otherwise.} \end{cases}$$

  - The discrete-time dynamics is obtained by discretizing the time with the time between t and $t + 1$ chosen equal to $0.100s$.
- Integration time step : 0.001.
- Reward signal :

$$r(p_t, s_t, u_t) = \begin{cases} -1 & \text{if} \quad p_{t+1} < -1 \quad \text{or} \quad |s_{t+1}| > 3 \\ 1 & \text{if} \quad p_{t+1} > 1 \quad \text{and} \quad |s_{t+1}| \leq 3 \\ 0 & \text{otherwise.} \end{cases}$$

- Discount factor : $\gamma = 0.95$.

[1] A terminal state can be seen as a regular state in which the system is stuck and for which all the futur rewards obtained in the aftermath are zero.

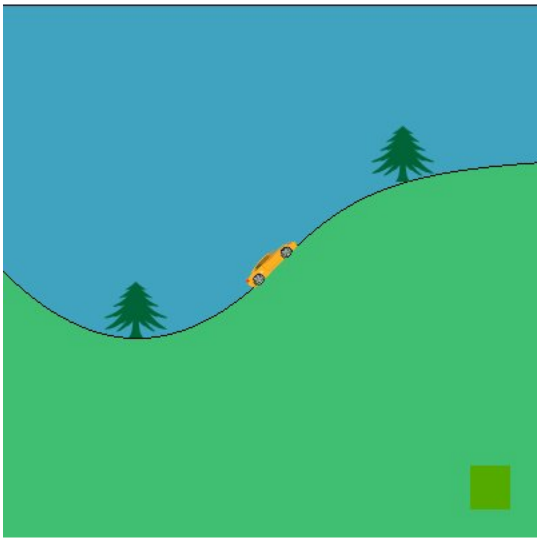

Figure 1: Display of the position $p = 0$ and the speed $s = 1$ of the car.

- Time horizon : $T \to +\infty$.

This domain is a *car on the hill* problem, and will be referred by this name from now.

Make sure you (i) rigorously test your implementation at each stage of the assignment and (ii) store any intermediate result to avoid redundant calculations.

2 Implementation of the Domain

Implement the different components of the *car on the hill* problem. Your implementation of the dynamics should exploit the Euler integration method. Make sure your implementation handles the terminal state case. Test your results by simulating a simple policy.

3 Visualization

Implement a routine which produce a video from any *car on the hill* trajectory. Your routine needs (i) to use the function, from this *Python script*, which produces an image of a given state from the *car on the hill* problem and (ii) to store the sequence of images of the video separately. The latter will be needed for the Assignment 3.

4 RL WITH SUPERVISED LEARNING ALGORITHMS

Implement a routine which compute $\widehat{Q}_N$ for $N = 1, 2, 3 ...$ using *Fitted-Q-Iteration*. Use the following supervised learning techniques :

- Linear/logistic regression,
- Extremely Randomized Trees,
- Neural networks.

All of them are already implemented in the *scikit-learn* and *Keras* programming libraries. Propose several strategies for generating sets of one-step system transition that will be used in your experiments. Display $\widehat{Q}_N$ for each supervised learning algorithm. You need to tune (some of) the parameters according to the supervised learning models. Derive the policy $\widehat{\mu}^*_N$ from $\widehat{Q}_N$. Compute and display $J^{\widehat{\mu}^*_N}$.

Extend your *Q-learning* implementation from Assignment 1 to handle parametric Q-functions in the form $\widehat{Q}(x, u, a^*)$. You need to consider as well parametric function approximators that are proposed here or in the scientific literature. Derive the policy $\widehat{\mu}_*$ from $\widehat{Q}(x, u, a^*)$. Compute and display $J^{\widehat{\mu}^*_N}$. Design an experiment protocol to compare *FQI* and *parametric Q-learning*.