Reinforcement Learning
Exercise 7

Jim Mainprice, Philipp Kratzer
Machine Learning & Robotics lab, U Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

June 15, 2020

Submission Instructions:

The submission deadline for this exercise sheet is 23.06., 23:55.

Put your answers into a single pdf. Your python code should be a single python script. Upload both files to ilias. Make sure that the code runs with *python3 yourscript.py* without any errors.

Group submissions of up to three students are allowed.

1 Linear function approximation (4P)

a) Show that tabular methods are a special case of linear function approximation. What does the feature vectors look like? (2P)

b) Give the update rule for Sarsa (λ) for: (2P)

- the tabular case
- with function approximation
- with linear function approximation

2 Mountain Car (8P)

The code template can be found on github (`https://github.com/humans-to-robots-motion/rl-course`) in *ex07-fa/ex07-fa.py*. For this exercise we will use the MountainCar environment from gym: `https://gym.openai.com/envs/MountainCar-v0/`

Starting from the bottom of a valley, an underpowered car has to gain enough momentum to reach the top of a mountain. The objective is to minimize the number of time steps to reach the goal. There are three possible values of action a:

- full throttle forward (+1)
- full throttle reverse (-1)
- and zero throttle (0)

The continuous state space is defined by $x_t = (x_t, \dot{x}_t)$, where the bounded state variables $x_t \in [-1.2, 0.6]$ and $\dot{x}_t \in [-0.07, 0.07]$ are respectively the position and velocity of the car. The reward in this problem is -1 on all time steps. An episode terminates when the car moves past its goal position $x_{t+1} \geq 0.5$ at the top of the mountain, or when the episode length is greater than 200.

state space grid

a) Implement Q(λ) with state-aggregation, e.g., 20 intervals for x and 20 for $\dot{x}$. Plot the value function at regular intervals (e.g., every 20 episodes). Pick reasonable parameters α, γ, and ϵ; tune for λ. (3P)

b) Repeat the process 10 times; plot the averaged cumulative number of successes (reaching goal state),and the averaged number of steps per episode (y-axis) against number of episodes (x-axis). Analyze the learning curves. (2P)

c) Implement Sarsa(λ) with linear function approximation (e.g. tile-coding or RBFs) and compare the learning curves against the ones from b). (3P)