Reinforcement Learning (SS18) - Exercise 7

Daniel Hennes

25.06.2018 (due 04.07.2018)

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

2. Give the update rule for Sarsa(λ) for:

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

3. Consider the mountain car problem described in the book and introduced in the lecture: 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 $\boldsymbol{x}_t = (x_t, \dot{x}_t)$, 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. At the beginning of each episode, the car starts at the random initial state $x_0 \in [-0.6, -0.4)$ and $\dot{x} = 0$, i.e., close to the bottom of the valley. The car moves according to the following equations:

   $$x_{t+1} = bound[x_t + \dot{x}_{t+1}]$$

   $$\dot{x}_{t+1} = bound[\dot{x}_t + 0.001 \cdot a_t - 0.0025 \cos(3x_t)]$$

   The *bound* operator enforces above state limits for $\boldsymbol{x}_{t+1}$. In addition, when the car reached the inelastic left bound, $\dot{x}_{t+1}$ is reset to zero. The reward in this problem is -1 on all time steps until the car moves past its goal position $x_{t+1} \geq 0.5$ at the top of the mountain, which ends the episode.

   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.

   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.

   You may use the OpenAI gym environment:

   ```
   # https://gym.openai.com/envs/MountainCar-v0/
   # https://github.com/openai/gym/blob/master/gym/envs/classic_control/mountain_car.py
   ```

4. Implement Sarsa(λ) with linear function approximation (e.g. tile–coding or RBFs) and compare the learning curves agains the ones from 1(b).