Reinforcement Learning WS 2018/19
Instructor: Sebastian Blaes `<sblaes@tuebingen.mpg.de>`
Due-date: 14.12.2018 (hand in at beginning of recitation session)
Exercise Sheet 8
December 7, 2018

1 Function approximation

(a) (Exercise 9.1 from Sutton): Show that tabular methods such as presented in Part I of Sutton's book are a special case of linear function approximation. What would the feature vectors be?

2 Feature designing

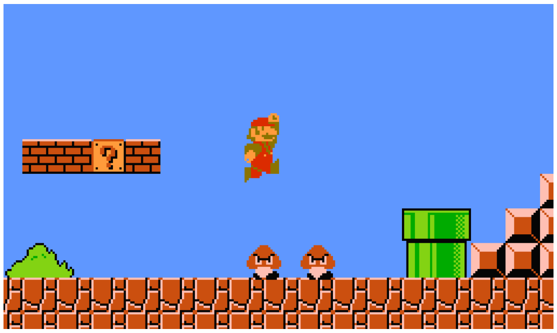

Figure 1: Super Mario Bros., ©NINTENDO 1985

(a) Imagine you want to implement a RL agent to play Super Mario (the original one). For the policy $\mu(x)$ to work, we need to construct features. Please give a example feature vector and what each component encodes.

Hint: A feature vector is given by $x = \langle x_1, x_2, \ldots, \rangle$, where x_1, for example, encodes the x-coordinate of Mario, x_2 encodes the y-coordinate of Mario, etc. Please give a feature vector of at least 5 dimensions.

3 Value function fitting

In this exercise, you will fit an approximated value function for a given policy. `valueapprox.zip` contains the following files:

custompendulumenv.py: Custom gym pendulum environment. This is a classic environment, Mujoco is not needed. (no changes required)
Note: Instead of a rewards, the environment returns a cost that is zero at the optimum and positive away from the optimum. This means that lower values are better.

agent.py This file implements the agent class. For you, the agent is a black box[1] that exposes a method *get_action* which you can use to query an action u for a given state s. (no changes required)

memory.py Implements a replay buffer that provides a method *addTuple* to add new transitions to the buffer and *sample* that, in our case, returns a tuple of the form (s_t, u_t, ct, s_{t+1}) where s_t is the current and s_{t+1} the next state, u_t is the action and c_t the cost. (no changes required)

pendulum_value_plot.ipynb In this notebook, you will implement the value fitting with a function approximator (NN). The notebook provides already routines for data collection and value function visualization. (Changes required)

(a) Use a function approximator, like a NN, to fit the value function. Use an appropriate target (td update) for the value fitting.

(b) Give an intuitive interpretation of the plotted value function. In case you could not solve part one of the exercise, Figure 2 shows a plot of the expected output.

[1] In fact, the agent outputs a linear mapping $u = -Ks$ from inputs s to outputs u. The gain factor K is derived from a linear quadratic regulator (LQR)

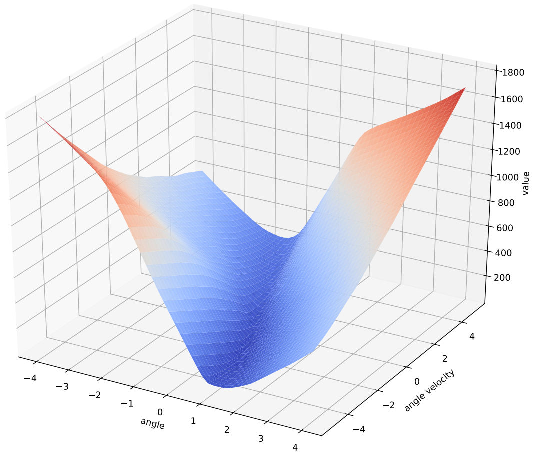

Figure 2: Approximated value function for a linear policy and the pendulum environment. Lower values are better.