# Exercises for “Deep Reinforcement Learning”

## Winter Term 2024 Sheet 3

Introduction to On-Policy Algorithms

**Issued:** 04.11.2024, **Due:** 17.11.2024, **Discussion:** 20.11.2024

This assignment sheet introduces you to the basics of On-Policy algorithms for estimating a value function and the corresponding policy. You will apply the algorithms of dynamic programming, Monte Carlo methodology, and temporal difference learning.

### Aufgabe 3.1 Dynamic Programming (DP): (4 points) = 2 + 1 + 1

This task is based on the Grid World environment from the previous exercise (Figure 1).

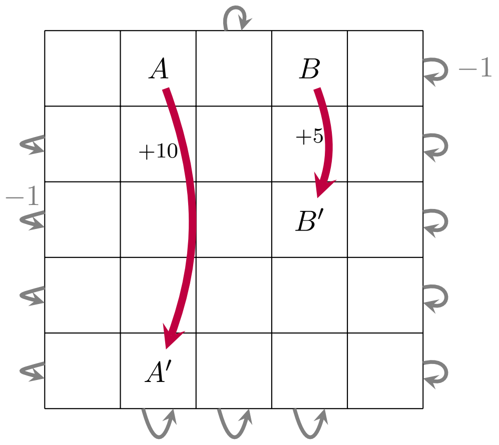


Abbildung 1: Grid World environment from previous exercise (see exercise sheet Value Functions).

Previously, you iteratively determined the state-value function of the Grid World environment using the Bellman equation and a random policy (uniform distribution) (Policy Evaluation). This provided a solution but not necessarily the optimal one. Now the policy is to be improved (Policy Improvement) by the agent using the knowledge of the expected values from a state (1-Step Lookahead) or through the action-state pair (Action-Value Function).

(a) **Implementation GPI**

- *Algorithm*: Implement the GPI algorithm. Start with a random policy and determine the state-value function through repeated application of the Bellman equation (Policy Evaluation). Use the computed state values for policy improvement. Repeat the cycle until the policy experiences no further improvements and converges.
- *Visualization*: Visualize the resulting state-value functions and the final optimal policy in a grid. Use directional arrows or symbols to depict the actions in the various states.

(b) **Modification to Value Iteration**

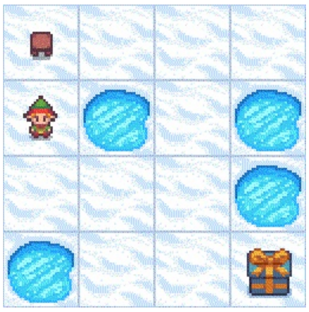
- *Algorithm*: Modify the GPI algorithm to a Value Iteration algorithm. Calculate the state-value function iteratively but conduct the policy improvement after each evaluation step, instead of waiting for full convergence.
- *Visualization*: Visualize the progress of the state-value function and the policy over the course of the iterations using the previously used diagram methods.
- *Analysis*: Write a short analysis of the results of both methods (GPI and Value Iteration) in which you examine and compare the convergence speed and stability of the value functions and the policy. What impact does the shorter evaluation strategy have?

(c) **GPI with Action Value Function**

- *Reflection Question*: Conceptually describe the differences that arise when using the Action-Value Function compared to the State-Value Function.
- *Algorithm*: Modify the GPI algorithm from task (a) to use Action-State Values.
- *Visualization*: Visualize the resulting Action-Value functions using a heatmap or grid, where each cell represents the Q-values of the possible actions (e.g., North, East, South, West) for a state.
- *Analysis*: Discuss the differences in the convergence rate compared to the use of the State-Value Function. Explain for which applications each of the two types of Value Functions is better suited.

**Aufgabe 3.2 Monte Carlo (MC) Algorithm:** (3 points) = 2 + 1

In real applications, we often lack full knowledge of the environmental dynamics. In this task, we will explore a method for solving finite Markov decision processes (MDP) that does not require knowledge of the environmental dynamics (state transition and reward probability). Instead, we will focus on learning a value function and a policy through experience by collecting sequences of states, actions, and rewards that result from interactions with the environment.



(a) Visualization of the environment.

```
[S, F, F, F,
 F, H, F, H,
 F, F, F, H,
 H, F, F, G]
```

(b) Actual representation of the environment.

Abbildung 2: In Frozen Lake, the goal is to cross a frozen lake from start(S) to goal(G) without falling into a hole(H) by walking over the frozen (F) lake. Due to the slipperiness of the frozen lake, the agent may not always move in the intended direction. [1]

Familiarize yourself with the OpenAI gym environment Frozen Lake (Figure 2) (Link). Instantiate the environment as follows: `gym.make('FrozenLake-v1', desc=None, map_name="4x4", is_slippery=True)`.

(a) **Monte Carlo Prediction $v \approx v_\pi$ (State Values)**

- *Reflection Question*: Select an initial policy: What could this look like (aside from a purely random policy)? Why is a purely random policy not always suitable? What characteristics must an initial policy have?
- *Algorithm*: Implement the Monte Carlo prediction algorithm to estimate the state values for the policy you defined earlier (Policy Evaluation).
- *Visualization*: Show the estimated values in the Frozen Lake environment as a heatmap.
- *Analysis*: Explain the limitations of Monte Carlo methods using state-value estimations in unknown environment dynamics and why these estimations alone are not sufficient for creating an optimal policy. Use your generated heatmap to clarify this fact.

(b) **On-Policy Monte Carlo Control $\pi \approx \pi_*$ (Action-Value)**

- *Algorithm*: Use the MC-Control Algorithm to learn the optimal policy using action values (Policy Evaluation and Control). Extend the previously implemented algorithm for MC Policy Prediction to estimate action values. Then adjust your policy iteratively, exploiting the Action-Value Function according to the idea of General Policy Iteration.
- *Visualization*: Illustrate the performance of each adjusted policy over the iterations and visualize your strongest policy.
- *Analysis*: Examine the action values for a specific state, e.g., state (3,2), and explain what these mean regarding the influences of the environment dynamics. How does the environment dynamics affect the evaluation of action values? To what extent can the calculated action values be relied upon in a non-deterministic environment?

**Aufgabe 3.3 Temporal Difference Learning (TD-Learning):** (3 points) = 2 + 1

TD learning is a method that acquires state/action values through experiential learning. Instead of relying on samples, as in the Monte Carlo algorithm, estimated values for future states are used to update current estimates. This technique is known as *bootstrapping*. Use the Frozen Lake environment (Figure 2) as before for the following subtasks.

(a) **TD(0) Prediction $V \approx V_\pi$ (State-Value)**

- *Algorithm*: Implement the TD(0) algorithm for the estimation of state values (Policy Evaluation). Use a random (uniformly distributed) policy.
- *Conceptualization*: Discuss the conceptual differences between Monte Carlo and TD learning, especially regarding incrementality and bootstrapping. What are the advantages of TD learning compared to Monte Carlo sampling?
- *Visualization*: Use a heatmap to graphically represent the estimated values for each state in the Frozen Lake environment.
- *Analysis*: Discuss the advantages of TD learning compared to Monte Carlo methods in terms of sample efficiency and adaptability.

(b) **SARSA (on-policy TD Control) $Q \approx Q_*$ (Action-Value)**

- *Algorithm*: Implement the SARSA algorithm (State-Action-Reward-State-Action) to learn the optimal policy using action values (Policy Evaluation and Control). Use an appropriate approach to balance exploration and exploitation.
- *Visualization*: Illustrate the average cumulative reward over the episodes. Compare scenarios with different exploration parameters to observe how the degree of exploration affects learning speed and achieved reward.

# Literatur

[1] Ariel Kwiatkowski, Mark Towers, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulao, Andreas Kallinteris, Markus Krimmel, Arjun KG, Rodrigo Perez-Vicente, Andrea Pierre, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments.