FAKULTÄT FÜR WIRTSCHAFTSINFORMATIK
UND WIRTSCHAFTSMATHEMATIK

UNIVERSITÄT MANNHEIM

Prof. Dr. Leif Döring
Daniel Schmidt, Benedikt Wille

Reinforcement Learning

7. Exercise Sheet

20.04.2026

For this assignment, knowledge from lectures 1 to 16 is assumed.

1. Convergence of Stochastic Gradient Descent

The goal of this exercise is to prove the convergence of the stochastic version of the gradient descent method. Let $F : \mathbb{R}^d \to \mathbb{R}$ be a function of the form $F(x) = \mathbb{E}[f(x, Z)]$ for some $Z \sim \mu$, whose minimum we want to find but whose gradient we cannot exactly compute. The idea is to approximate the gradient of F by $\nabla_x f(x, Z_i)$ with independent realisations $Z_i \sim \mu$ in each step, leading to the following algorithm:

Data: Realisation of initial random variable X_0, stepsizes α_k
Result: Approximation X of a stationary point of F
Set $k = 0$
while *not converged* **do**
  simulate $Z_{k+1} \sim \mu$ independently
  approximate the gradient $\nabla_x F(X_k)$ through
  $G_k = \nabla_x f(X_k, Z_{k+1})$
  set $X_{k+1} = X_k - \alpha_k G_k$
  set $k = k + 1$
end
return $X := X_k$

Algorithm 1: Stochastic Gradient Descent

Assume the following:

- Let $(\Omega, \mathcal{F}, (\mathcal{F}_k)_{k\in\mathbb{N}}, \mathbb{P})$ be a filtered probability space, where the filtration is defined by

$$\mathcal{F}_k := \sigma(X_0, Z_m, m \le k) \text{ for } Z_k \sim_{\text{i.i.d}} \mu,$$

- let $F : \mathbb{R}^d \to \mathbb{R}$, $x \mapsto \mathbb{E}[f(x, Z)]$ for $Z \sim \mu$ be an L-smooth function for some $L < 1$, i.e.

$$\|\nabla F(x) - \nabla F(y)\| \le L\|x - y\| \quad \forall x, y \in \mathbb{R}^d$$

  and let $F_* := \inf_{x\in\mathbb{R}^d} F(x) > -\infty$,
- let $\nabla_x F(x) = \mathbb{E}[\nabla_x f(x, Z)]$ and $\mathbb{E}[\|\nabla_x f(x, Z)\|^2] \le c$ for some $c > 0$ and all $x \in \mathbb{R}^d$,
- let $(\alpha_k)_{k\in\mathbb{N}}$ be a sequence of $\mathcal{F}_k$-adapted and strictly positive random variables, where

$$\sum_{k=1}^{\infty} \alpha_k = \infty \text{ and } \sum_{k=1}^{\infty} \alpha_k^2 < \infty$$

- let X_0 be such that $\mathbb{E}[F(X_0)] < \infty$, and

- let $(X_k)_{k\in\mathbb{N}}$ be the random variables generated by applying Stochastic Gradient Descent.

a) For all L-smooth functions $f : \mathbb{R}^d \to \mathbb{R}$ it holds that

$$f(x+y) \leq f(x) + y^T \nabla f(x) + \frac{L}{2}\|y\|^2 \quad \forall x, y \in \mathbb{R}^d.$$

b) Define $M_{k+1} := \nabla_x F(X_k) - \nabla_x f(X_k, Z_{k+1})$ and show that

$$\mathbb{E}[M_{k+1}|\mathcal{F}_k] = 0 \text{ and } \mathbb{E}[\|M_{k+1}\|^2|\mathcal{F}_k] \leq c - \|\nabla_x F(X_k)\|^2 \quad \forall k \in \mathbb{N}.$$

c) Schow that $\lim_{k\to\infty} F(X_k) = F_\infty$ almost surely for some almost surely finite random variable.

d) Show that $\lim_{k\to\infty} \|\nabla_x F(X_k)\|^2 = 0$ almost surely.

2. *Programming task: SARSA

The aim of this exercise is to implement the SARSA algorithm (Algorithm 20), which we will later use to demonstrate weaknesses of the Q-learning algorithm when it is applied to robust Reinforcement Learning. In addition to the behaviour policy mode from the last exercise sheet, implement ϵ-greedy exploration with both a constant ϵ and ϵ decreasing with a rate function. *Hint: You may use the same helper function that determines the stepsizes for implementing ϵ-greedy exploration.*

3. *Programming task: Actor-Critic revisited

The aim of this exercise is to create a general actor-critic method (cf. Algorithm 11). Implement an algorithm that combines a policy evaluation scheme with a policy improvement scheme (so far, we have only used the greedy policy derived from the policy evaluation). Compare this algorithm with different choices of critics to the other stochastic control algorithms we have implemented, analyzing their performance and behavior in various environments.

4. *Programming task: Exploration matters

The aim of this exercise is to observe how different exploration schemes affect the performance of Q-learning. Test at least a uniformly random behavior policy and ϵ-greedy exploration with a fixed high rate, a fixed low rate, and a rate of $\frac{1}{n}$ on the following environments: multi-step bandits with 10 branches, 2 sequential steps each, and normally distributed arms. The first environment should resemble a regular multi-armed bandit with one possible action at each step, while the second should have 10 possible actions at each step. In both environments, one arm should have a mean of 0.1 with variance 1, and all other arms should have a mean of -0.1 with variance 1. Analyze how these effects relate to the concepts of exploration and exploitation discussed in the bandit context.

5. *Programming task: Robust RL

The aim of this exercise is to demonstrate an effect of SARSA compared to Q-learning in the context of robust RL, where exploration can be costly. For instance, imagine a robot learning to drive a car, where every crash requires purchasing a new car. In such scenarios, algorithms like SARSA, which are designed to be more cautious, can be advantageous. To illustrate this, visually compare the estimated policies of Q-learning and SARSA during training on a windy GridWorld. The grid is 12×4, with the start state in the top left and the goal state (reward 100) in the top right. Along the top line are cliff states (reward -100), and the default reward is -1. Experiment with different wind strengths (probabilities) for being pushed upwards towards the cliff and variations of the reward structure. Analyze why different policies may emerge during optimization based on the algorithm update rules. Formulate a hypothesis on the general behavior and comparability of the algorithms in environments with very low rewards, and explain the differences through the algorithmic distinctions between Q-learning and SARSA.

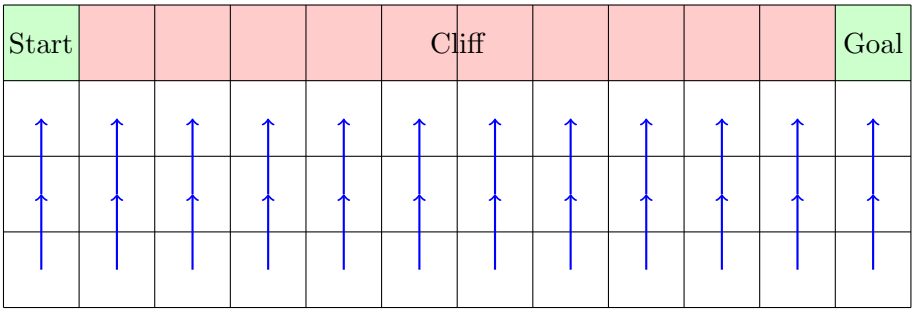

Abbildung 1: Visualization of the windy cliff walk game.

6. *Programming task: The Bias Problem

The aim of this exercise is to observe a well-known effect of Q-learning and determine whether the alternative versions implemented in Exercise 2 can counteract it. The issue arises from the fact that $\mathbb{E}[\max_{1\leq i\leq n} X_i] \geq \max_{1\leq i\leq n} \mathbb{E}[X_i]$, leading to a positive bias in the estimator for the Bellman operator used in Q-learning. Implement metrics to measure the summed total bias, the summed squared total bias, and the summed bias and summed squared bias at selected state-action pairs. Compare the overestimation effect in Q-learning to its alternative versions and provide a theoretical explanation for why these algorithms exhibit less overestimation bias. While overestimation bias would not be problematic if it were uniform across all state-action pairs, identify factors influencing the magnitude of overestimation bias in Q-learning and the extent to which the alternative algorithms reduce it. Use this knowledge to construct examples of multi-step bandits and Grid Worlds where Q-learning performs poorly, the alternative algorithms perform poorly, and where both types of algorithms fail.

The solution to the theoretical exercises will be discussed in the exercise class in B2 on April 20, 2026, at Mathelounge in B6 B301.