Prof. Dr. Leif Döring
Benedikt Wille

Reinforcement Learning
06.05.2025

10. Exercise Sheet

For this assignment, knowledge from lectures 1 to 22 is assumed.

1. (μ-strong) convexity

a) Prove that all local minima of convex functions must be global minima. In particular, this means that all global minima have same height.

b) Show that a μ-strongly convex function has a unique global minimum

c) Let $f : \mathbb{R}^d \to \mathbb{R}$ with $f(x) = x^T Ax$ for a matrix $A \in \mathbb{R}^{d\times d}$. Show that f is L-smooth and μ-strongly convex, where L is twice the largest and μ twice the smallest singular value.

2. PL-condition

a) Prove that μ-strong convexity implies the PL-condition (4.5), i.e.

$$\|\nabla f(x)\|^2 \geq 2r(f(x) - f_*) \quad \forall x \in \mathbb{R}^d \tag{1}$$

for $r = \mu$ and $f_* = \min_{x\in\mathbb{R}^d} f(x) > -\infty$.

b) Show that $f(x) = x^2 + 3\sin^2(x)$ satisfies the PL-condition and prove that f is not convex. Plot the function to see why gradient descent converges.
Hint: The plot can also help to find the parameter r of the PL-condition.

3. Stochastic gradient descent

In the lecture we proved convergence of SGD to stationary points if the function is L-smooth and bounded from below and we use stepsizes fulfilling the Robbins-Monro conditions. Consider the setting from the theorem of the lecture and additionally assume μ-strong convexity. Prove that $\|X_n - x_*\| \to 0$ almost surely.

4. Fully deterministic game with non-convex value function

Imagine a game where you have two states s_1 and s_2 and two actions a_1 and a_2 in each of the states. The game is fully deterministic and after each turn switches, regardless of the action, between s_1 s_2, giving, regardless of the state, a reward of 1 in a_1 and -1 in a_2 and $\gamma \in (0, 1)$. Assume a "learner", that decides based on a one-dimensional linear softmax with features $\phi(s, a) \in \mathbb{R}$. Show that there exist (non-trivial) values of the features such that the value function of this game is non-convex in the parameter θ of the softmax family.

5. Non-convex, L-smooth value function

Imagine a game with a starting state, two states s_0 and s_1 as well as a terminal state T. A player in the starting state may decide to transfer to either state and receive rewards R_0 and R_1 accordingly, after which he is transferred to the terminal state. Consider the (differentiable parameterised) family of stationary policies $\{\pi^\theta\}_{\theta\in[0,1]}$ with

$$\pi_\theta = \theta^2\delta_{\to s_0} + (1-\theta^2)\delta_{\to s_1}$$

that transfers to s_0 with probability θ^2 and to s_1 with probability $1-\theta^2$.

a) Assume that R_0 and R_1 are chosen uniformly and independently according to a continuous probability distribution. Show that the value function is non-convex with a probability of at least $\frac{1}{2}$.

b) Show that regardless of the values of R_0 and R_1 the value function is L-smooth, so that the gradient descent with diminishing stepsizes from the lecture converges

c) Show that, regardless of the values of R_0 and R_1, if you restrict the parameter θ to a subset $[\theta_0, \theta_1] \subset [0,1]$ it fulfills the PL-condition and find a constant stepsize, such that gradient descent with constant stepsize converges to a global minimum.

6. *Programming task: REINFORCE

The aim of this exercise is to implement Algorithms 31 and 32: REINFORCE and Mini-batch REINFORCE. To get started, study the A2C algorithm (e.g. here)—it's already implemented in Stable Baselines3. Use the existing A2C implementation as a template: copy it into a separate folder within the Stable Baselines3 Zoo and modify it where necessary. Don't forget to create a corresponding hyperparameter file.

7. *Programming task: Experiment size and metrics

The aim of this exercise is to help you understand the difference in experiment scale between tabular environments and classic control tasks. Train your custom algorithms on the Gymnasium environments Acrobot and CartPole, using as many seeds as feasible within a reasonable timeframe. Monitor and plot the logged metrics, and compare them to what you've seen in the tabular case. Additionally, explore the rliable python package and use it to visualize further aggregated evaluation metrics across the two environments. Reflect on why rliable and its metrics have become widely used in the RL community and what problem they solve.
Hint: It can be interesting to see how trained agents behave in the environments. Try using the enjoy function to visualize their performance.

The solution to the theoretical exercises will be discussed in the exercise class in B5 on May 13, 2025, at Mathelounge in B6 B301.