# Reinforcement Learning
# Exercise 5

October 10, 2021

In this exercise, we will implement a policy gradient algorithm for the CartPole environment and compare REINFORCE with and without a constant baseline. We will use a modified version of the CartPole environment—the standard CartPole uses discrete actions (applying force of either -10 or 10), while in this version takes actions in the form of a single real number, which represents an arbitrary force value.

## 1 Policy Gradient

Recall the policy gradient derivation from Lecture 6.

### 1.1 Policy gradient with and without baseline

**Task 1 — 25 points** Implement policy gradient for the CartPole environment with continuous action space. Use `agent.py` for implementing the reinforcement learning algorithm itself (for example, the agent and policy classes). Use these classes to implement the main training loop in the `cartpole.py` file, similarly to how it was done in Exercise 1.

Use constant variance $\sigma^2 = 25$ for the output action distribution throughout the training.

(a) basic REINFORCE without baseline **(15 points)**,

(b) REINFORCE with a constant baseline $b = 20$ **(5 points)**,

(c) REINFORCE with discounted rewards normalized to zero mean and unit variance **(5 points)**,

**Hint:** The `agent.py` file contains a basic neural network structure, together with some reasonable hyperparameter values (such as the network size and learning rate).

**Hint:** Your policy should output a probability distribution over actions. A good (and easy) choice would be to use a normal distribution (`from torch.distributions import Normal`). Log-probabilities can be calculated using the `log_prob` function of the distribution.

**Question 1.1 — 10 points** How would you choose a good value for the baseline? **Justify your answer.**

**Question 1.2 — 5 points** How does the baseline affect the training, and **why?**

## 1.2 Choosing the value of variance

**Task 2 — 10 points** Implement two cases of adjusting variance during training: exponentially decaying variance $\sigma^2 = \sigma_0^2 \cdot e^{-c \cdot k}$ (where $c = 5 \cdot 10^{-4}$ and $k$ is the number of episode) and variance learned as a parameter of the network (in both cases set the initial value $\sigma_0^2$ to 100). Use REINFORCE with normalized discounted returns for this task. The expected results for the learned variance case are shown in Figure 1.

**Hint:** To make your learned variance automatically updated by the optimizer, declare your variable inside the `__init__` function of the model using `torch.nn.Parameter(some_tensor)`.

**Question 3.1 — 5 points** Compare using a constant variance, as in Task 1, to using exponentially decaying variance and to learning variance during training. **Please explain** what the strong and weak sides of each of those approaches are.

**Question 3.2 — 5 points** In case of learned variance, what's the impact of initialization on the training performance? **Please explain**.

## 1.3 PG and experience replay

**Question 4.1 — 10 points** Could the method implemented in this exercise be **directly** used with experience replay? **Why/why not?**

**Question 4.2 — 5 points** Which steps of the algorithm would be problematic to perform with experience replay, if any? **Explain your answer.**

## 1.4 Real-world control problems

**Question 5.1 — 5 points** What could go wrong when a model with an unbounded continuous action space and a reward function like the one used here (+1 for survival) were to be used with a physical system?

**Question 5.2 — 10 points** How could the problems appearing in Question 5.1 be mitigated without putting a hard limit on the actions? **Explain your answer.**

## 1.5 Discrete action spaces

**Question 6 — 10 points** Can policy gradient methods be used with discrete action spaces? Why/why not? Which steps of the algorithm would be problematic to perform, if any? **Explain your answer.**

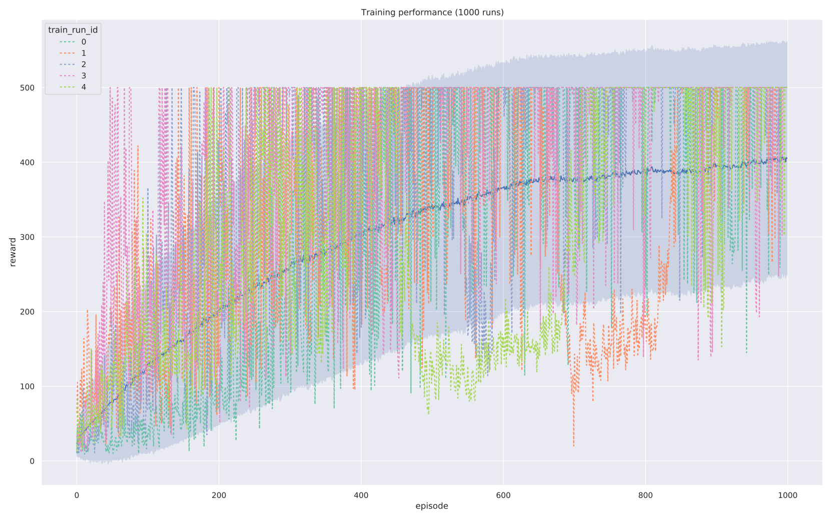

Figure 1: Expected results for REINFORCE with normalized discounted returns and learned variance.

## 2 Submission

The deadline to submit the solutions through MyCourses is on Monday, 1.11 at 12:00 (noon). Example solutions will be presented during exercise sessions the same week (1–3.11).

1. **Answers to all questions** asked in the text.
2. The **training performance plots** for each of the tasks (Task 1a, b and c, Task 2 - for different sigma values).

In addition to the report, you must submit as separate files, in the same folder as the report:

1. Python code used to solve **all task exercises**.

Please remember that not submitting a PDF report following the **Latex template** provided by us will lead to subtraction of points.

For more formatting guidelines and general tips please refer to the submission instructions file on mycourses.

If you need help or clarification solving the exercises, you are welcome to come to the exercise sessions in weeks 41/42/43 (on Mondays, Tuesdays and Wednesdays).

Good luck!