# Exploration and uncertainty

All assignments (questions marked with an A that yield points) must be submitted on Brightspace before the corresponding tutorial begins (see due date). Please submit all answers in one PDF file. Scans of handwritten solutions (e.g. for math answers) are permitted, but must be readable and have a file-size below 5MB. Do not submit the voluntary exercises (marked with an E), which will not earn you any points, but can help you practice the math and prepare for the exam.

You will also be asked to implement and test something in python/pytorch. We recommend that you use a Jupyter Notebook, convert your final version (including result plots) to PDF ("Download as" → "PDF via LaTeX") and attach the PDF at the end of your submission PDF. You are welcome to use other editors, but please make sure you submit the code (or the crucial code segments) and the results in an easily readable format together with your theory answers as one PDF file.

A sample solution will be published after the tutorial. Please always demonstrate how you arrived at your solution. You will receive the points when you convince us that you have seriously attempted to answer the question, even if your answer is wrong. You qualify for the exam when you have earned 75% of the total achievable points (sum over all 4 exercise sheets, not for individual sheets).

Good luck!

## A3.1: Implement and test actor-critics **(5 points)**

In this exercise you will implements the vanilla actor-critic algorithm, an off-policy extension and finally PPO. Your implementations will be based on a REINFORCE implementation from the Jupyter notebook `ac.ipynb`, that can be easily extended. Please don't forget that *all* of your answers have to be submitted in *one* PDF file, including this one.

(a) *[1 point]* Read carefully through the provided code (only few classes have changed from exercise sheet 2) and run the given REINFORCE algorithm on the `Cartpole-v1` environment for 2 million (2M) steps. You can stop episodes after 200 steps. This can take 10-20 minutes.

(b) *[1 point]* Extend the `ReinforceLearner` class in the given Jupyter Notebook with a value function as bias (see slide 6 of Lecture 6). Implement two target-definitions for the value function, selected by the `'value_targets'` parameter: `'returns'` uses the returns $R_t$ that are stored in the mini-batch, whereas `'td'` uses the TD-error. Make sure that the bias is ignored when the given parameter `'advantage_bias'` is **`False`**. Test your implementation as above on the evironment `Cartpole-v1` for 2M steps with the default parameters (using `'returns'` value targets).

*Hint:* The first heads of the model are interpreted as logits of a softmax policy, and the last head of the model is interpreted as the value function.

(c) *[1 point]* Extend the `BiasedReinforceLearner` class in your implementation with an advantage function that uses bootstrapping (replaces $R_t$ with $r_t + \gamma v_\phi(s_{t+1})$, see slide 6 of Lecture 6). Make sure the original behavior is maintained when the parameter `advantage_bootstrap` is **`False`**. Test your implementation as above on the evironment `Cartpole-v1` for 2M steps with the default parameters.

(d) *[1 point]* Run your `ActorCriticLearner` with with 80 off-policy iterations (by setting the parameter `params['offpolicy_iterations'] = 80`). Extend your implementation to the class `OffpolicyActorCriticLearner`, that uses on-policy gradients in the first and off-policy gradients in all following iterations ($\mathcal{L}_\mu$ on slide 15 of Lecture 6). Test your implementation as above on the evironment `Cartpole-v1` with the default parameters, but only for 500k steps.

*Hint:* `ReinforceLearner` has an attribute `old_pi` which is set to **`None`** at the beginning of `train()`. You can save the on-policy probabilities of the initial policy here to use them in the ratios of the off-policy loss.

(e) *[1 point]* Now extend `OffpolicyActorCriticLearner` by adding PPO clipping to the off-policy loss ($\mathcal{L}_\mu^{\text{clip}}$ on slide 18 of Lecture 6). Test your implementation as above on the evironment `Cartpole-v1` with the default parameters, but only for 500k steps.

## A3.2: Policy optimization under constraints **(3 points)**

This exercises assumes a multi-armed bandit, where the player is choosing between arms $a \in \mathcal{A} \subset \mathbb{N}$ to get a randomized reward $r \in \mathbb{R}$ drawn from $p(r|a)$. Let $Q(a) := \mathbb{E}[\, r \,|\, r \sim p(\cdot|a)]$ denote the expected reward of action $a \in \mathcal{A}$. Prove analytically that the solution to the optimization problem

$$\max_\pi \; \mathbb{E}[\, r \,|\, r \sim p(\cdot|a), a \sim \pi(a)] \qquad \text{s.t.} \qquad \mathcal{H}(\pi) \;\geq\; \epsilon\,,$$

where $\pi$ is a probability distribution over $\mathcal{A}$ and $\mathcal{H}(\pi) = -\sum_a \pi(a) \ln \pi(a)$ the corresponding entropy, is the softmax of the expected reward (with some inverse temperature $\beta$):

$$\pi(a) \quad = \quad \frac{\exp(\beta Q(a))}{\sum_{a'} \exp(\beta Q(a'))}\,, \quad \forall a \in \mathcal{A}\,.$$

*Bonus-question:* While there exists no analytical solution for the inverse temperature $\beta$, can you derive an iterative approach that improves the optimization problem for a given $\beta$ and target entropy $\epsilon$?

*Hint:* Try to formulate the above optimization problem with the method of Lagrangian multipliers, which you can read up on at *`https://en.wikipedia.org/wiki/Lagrange_multiplier`*. You may ignore the implicit constraints $\pi(a) \geq 0, \forall a \in \mathcal{A}$, at first and check whether the solution adhers to them later.

## A3.3: TD(1) is Monte-Carlo sampling **(3 points)**

Let $\{s_t, a_t, r_t\}_{t=0}^\infty$ denote an infinite trajectory sampled from some Markov chain. Prove analytically that in the limit $\lambda \to 1$ the TD($\lambda$) targets $V_\lambda^\pi(s)$ are identical to the Monte-Carlo targets $V_{\text{MC}}^\pi(s)$ .

$$V_\lambda^\pi(s_0) = (1-\lambda)\sum_{n=0}^{\infty} \lambda^n V_{n+1}^\pi(s_0), \qquad V_n^\pi(s_0) = \sum_{t=0}^{n-1} \gamma^t r_t + \gamma^n V^\pi(s_n), \qquad V_{\text{MC}}^\pi(s_0) = \lim_{n\to\infty} V_n^\pi(s_0)\,.$$

*Hint:* You can assume that $\lim_{\lambda\to 1} \lim_{m\to\infty} \lambda^m\, x_m = 0$, for any bounded sequence $-\infty < x_m < \infty, \forall m \geq 0$.

*Hint:* You can use the identity $\sum_{i=0}^{m} \sum_{j=0}^{i} A_{ij} = \sum_{j=0}^{m} \sum_{i=j}^{m} A_{ij}$, which can be proven by rearranging sumands.

## A3.4: Random exploration decays exponentially **(3 points)**

In this exercise we will show how $\epsilon$-greedy exploration yields exponentially decaying exploration, which can prevent an RL agent from learning some MDPs in expectation. For simplicity we look at an infinite state-chain $\mathcal{S} = \mathbb{Z}$, where we always start in $s_0 = 0$ and have four actions $\mathcal{A} = \{L, R, U, D\}$. The transition model is $P(s-1|s, L) = P(s|s, U) = P(s|s, D) = P(s+1|s, R) = 1$. We will explore the MDP by executing a uniform random policy $\pi(a|s) = \frac{1}{|\mathcal{A}|}, \forall a \in \mathcal{A}, \forall s \in \mathcal{S}$.

(a) *[1 point]* Derive the recursive update equation for the state distribution after $t$ time steps $\xi_t^\pi(s)$.

*Hint:* think about from which states one can transition into $s$ and define $\xi_{t+1}^\pi(s)$ in terms of $\xi_t^\pi(s)$.

(b) *[1 point]* Prove analytically that $\xi_t^\pi(s) = \frac{1}{4^t}\binom{2t}{t+s}, \forall |s| \leq t$, otherwise $\xi_t^\pi(s) = 0$.

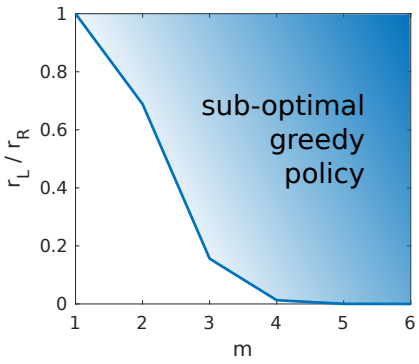
*Hint:* $\binom{n}{k} = \frac{n!}{k!(n-k!)}$ are binomial coefficients.[1] You can use the identity $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$.

(c) *[1 point]* The above $\xi_t^\pi(s)$ converges approximately[2] in the limit $t \to \infty$ to the normal distribution $\xi_\infty^\pi(s) = \mathcal{N}(s|0, 1)$. Let in the following $\bar{Q}(s, a) := \mathbb{E}_\pi[\lim_{n\to\infty} \frac{1}{n} \sum_{t=0}^{n-1} r_t | {}^{s_0=s}_{a_0=a}]$ denote the expected average reward conditioned on state and action. Reaching state $s = -1$ is rewarded by $r_L > 0$ and reaching the state $s = m$ is rewarded by $r_R > r_L$. Prove analytically that

$$\bar{Q}^\pi(0, L) \;>\; \bar{Q}^\pi(0, R) \qquad \Leftrightarrow \qquad r_L \;>\; r_R \sqrt{2\pi e}\, \frac{\sinh m}{\sinh 1}\, \xi_\infty^\pi(m)\,,$$

that is, that the greedy policy based on $\bar{Q}^\pi$ selects a suboptimal action if $m$ is too large.

*Hint:* The hyperbolic sine is defined as $\sinh(m) = \frac{1}{2}(e^m - e^{-m})$.




## A3.5: Epistemic uncertainty **(3 points)**

There is no general consensus on how epistemic uncertainty should be defined mathematically, only that it represents the "reducible" uncertainty about our prediction. In this exercise you will derive *one* possible definition. Let in the following $x \in \mathcal{X}$ denote samples drawn i.i.d. from distribution $\rho(x)$ and $y \in \mathbb{R}$ denote labels drawn i.i.d. from $\mathcal{N}(y|\mu(x), \sigma^2(x))$. $\mathbb{E}_z$ and $\mathbb{V}_z$ refer to the expectation and variance over the variable $z$, e.g. $x$ or $y$. We have estimated the function $f_\mathcal{D}(x) \approx \mathbb{E}_y[y|x]$ with some type of *unbiased regression* on the data set $\mathcal{D} := \{x_t, y_t\}_{t=1}^n \subset \mathcal{X} \times \mathbb{R}$, which implies $\mathbb{E}_\mathcal{D}[f_\mathcal{D}(x)|x] = \mu(x)$.

(a) *[1 point]* Prove that $\mathbb{E}_z\big[(g(z) - \alpha)^2\big] = \mathbb{V}_z[g(z)] + \big(\mathbb{E}_z[g(z)] - \alpha\big)^2, \forall g : \mathcal{Z} \to \mathbb{R}, \forall \alpha \in \mathbb{R}$.

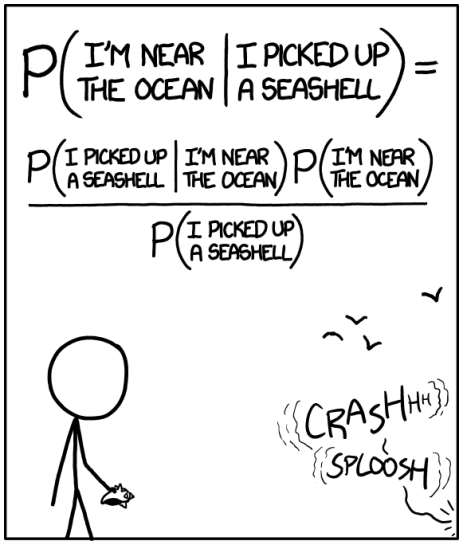
[1] see `https://en.wikipedia.org/wiki/Binomial_coefficient` for more information and identities.
[2] Proving this is *very* involved, see `http://www.math.uchicago.edu/~lawler/reu1` for a similar derivation.

(b) *[1 point]* Prove that the *expected generalization error* for a mean-squared loss of $f_{\mathcal{D}}$ is

$$\mathbb{E}_{\mathcal{D}}\Big[\underbrace{\mathbb{E}_x\big[\mathbb{E}_y\big[(y - f_{\mathcal{D}}(x))^2\big|x\big]\big]}_{\text{generalization error of } f_{\mathcal{D}}}\Big] \quad = \quad \underbrace{\mathbb{E}_x[\sigma^2(x)]}_{\text{aleatoric}} + \underbrace{\mathbb{E}_x\big[\mathbb{V}_{\mathcal{D}}[f_{\mathcal{D}}(x)|x]\big]}_{\text{epistemic uncertainty}} .$$

Note that the generalization error splits into an irreducible aleatoric uncertainty and an epistemic uncertainty over what we have inferred from the data. The latter reduces to zero if and when the variance between predictions for different $\mathcal{D}$ vanishes (which it should in the limit of infinite data).

(c) *[1 point]* Prove that the epistemic uncertainty of the *empirical mean* $f_{\mathcal{D}} = \frac{1}{n}\sum_{t=1}^{n} y_t$ for the labels of one constant input sample $x_t = x, \forall t$, is $\mathbb{V}_{\mathcal{D}}[f_{\mathcal{D}}(x)|x] = \frac{\sigma^2(x)}{n}$ .




STATISTICALLY SPEAKING, IF YOU PICK UP A SEASHELL AND *DON'T* HOLD IT TO YOUR EAR, YOU CAN PROBABLY HEAR THE OCEAN.

https://xkcd.com/1236

**Total 17 points.**