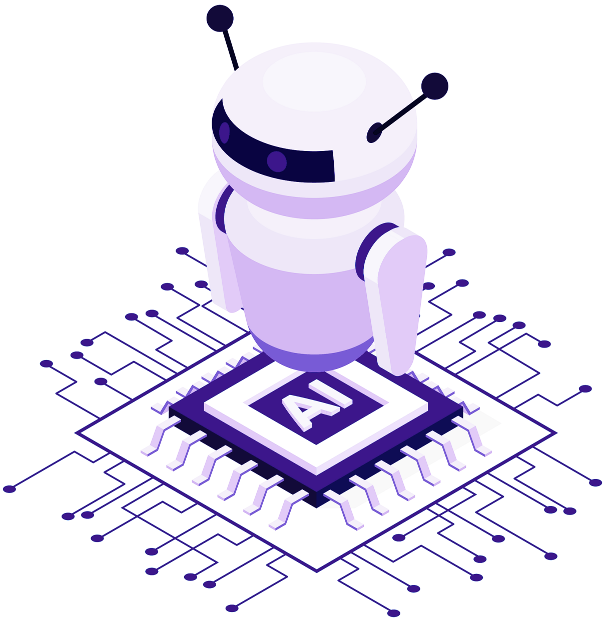

# Deep Reinforcement Learning

Professor Mohammad Hossein Rohban

Homework 9:

## Advanced Theory

Designed By:

Abdollah Zohrabi
abdollahzz1381@gmail.com

Soroush Vafaie Tabar
abdollahzz1381@gmail.com

Spring 2025

# Preface

Welcome to the homework!

In this homework, we aim to extend our theoretical knowledge of bandits to more complex scenarios. Unlike the simple version that was described in the class, real-world bandits assume a more general distribution for arm values instead of limiting them to 0 to 1 intervals. We first introduce the sub-Gaussian variables as an often-used light tail distribution and prove their bounds. Then, We extend the UCB bounds to use 1-sub-Gaussian variables. Afterward, we finish this homework by exploring a practical variation of bandits in online learning. These concepts form a fundamental basis for better understanding online learning and its applications in real-world problems such as recommendation systems, targeted advertising, and sequential optimization.

## Grading

The grading will be based on the following criteria, with a total of 100 points:

| **Task** | **Points** |
|---|---|
| **1 Light-tailed Distributions** | |
| —Hoeffding's | 10 |
| —Sub-Gaussian | 15 |
| **2 UCB** | |
| —UCB | 40 |
| **3 Online Learning** | |
| —Randomized Weighted Majority Algorithm | 35 |
| Bonus (Writing your report in Latex) | 5 |
| Bonus (UCB Power-of-2 Variation) | 35 |
| Bonus (Hedge algorithm) | 15 |

Only one of the optional parts will be graded.

## Submission

The deadline for this homework is 1404/03/16 (June 6th 2025) at 11:59 PM.

Please submit your work by following the instructions below:

- Place your solution alongside the Jupyter notebook(s).
  - Your written solution must be a single PDF file named `HW9_Solution.pdf`.
- Zip all the files together with the following naming format:

  `DRL_HW9_[StudentNumber]_[FullName].zip`

  - Replace `[FullName]` and `[StudentNumber]` with your full name and student number, respectively. Your `[FullName]` must be in CamelCase with no spaces.
- Submit the zip file through Quera in the appropriate section.
- We provided this LaTeX template for writing your homework solution. There is a 5-point bonus for writing your solution in LaTeX using this template and including your LaTeX source code in your submission, named `HW9_Solution.zip`.
- If you have any questions about this homework, please ask them in the Homework section of our Telegram Group.
- If you are using any references to write your answers, consulting anyone, or using AI, please mention them in the appropriate section. In general, you must adhere to all the rules mentioned here and here by registering for this course.

Keep up the great work and best of luck with your submission!

# Contents

1 Light-tailed Distributions 1

1.1 Hoeffding's Inequality 1

1.2 Sub-Gaussian 1

2 UCB 2

2.1 The Upper Confidence Bound Algorithm 2

2.2 Power of 2 version of UCB Algorithm$^*(Bonus)$ 4

3 Online Learning 4

3.1 Randomized Weighted Majority Algorithm 4

3.2 Hedge Algorithm$^*(Bonus)$ 5

# 1 Light-tailed Distributions

## 1.1 Hoeffding's Inequality

(a) For a random variable $X$ with $\mathbb{E}X = 0$ and $a \leq X \leq b$ then for $s > 0$

$$\mathbb{E}e^{sX} \leq e^{s^2(b-a)^2/8}$$

(b) Let $Z_1, \ldots, Z_n$ be independent bounded random variables with $Z_i \in [a, b]$ for all $i$, where $-\infty < a \leq b < \infty$. Then

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \geq t\right) \leq \exp\left(\frac{-2nt^2}{(b-a)^2}\right)$$

and

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^{n}(Z_i - \mathbb{E}[Z_i]) \leq -t\right) \leq \exp\left(\frac{-2nt^2}{(b-a)^2}\right)$$

for all $t \geq 0$.

## 1.2 Sub-Gaussian

**Definition:** A random variable $X$ with mean $\mu = \mathbb{E}(X)$ is sub-Gaussian if there exists a positive number $\sigma$ such that

$$\mathbb{E}\left[e^{\lambda(X-\mu)}\right] \leq e^{\lambda^2\sigma^2/2} \quad \forall \lambda \in \mathbb{R}.$$

$\sigma$ is known as the sub-Gaussian parameter.

**Importance:** Sub-Gaussian random variables are important in probability theory, statistics, and machine learning because they offer strong concentration properties and tail bounds, making them ideal for analyzing the behavior of random processes. These variables exhibit light tails, meaning the probability of large deviations from the mean decays at least as fast as the Gaussian distribution — exponentially fast — which ensures that extreme values are highly unlikely. This light-tailed nature is critical for deriving powerful inequalities (like Hoeffding's and Bernstein's) that bound the probability of large errors in estimators or learning algorithms. As a result, sub-Gaussian variables provide a robust mathematical foundation for understanding and controlling uncertainty in high-dimensional settings, randomized algorithms, and statistical learning theory which we will use to our best in computing regrets.

(a) If $X$ is a sub-Gaussian random variable with parameter $\sigma$, prove that for any $t > 0$,

(1) $\Pr[X > \mathbb{E}[X] + t] \leq e^{-t^2/2\sigma^2}$.

(2) $\Pr[X < \mathbb{E}[X] - t] \leq e^{-t^2/2\sigma^2}$.

(3) $\Pr[|X - \mathbb{E}[X]| \geq t] \leq 2e^{-t^2/2\sigma^2}$.

.

(b) [Hoeffding's Inequality] Suppose that the random variable $\{X_i\}_{i=1}^n$ are independent and $\mathbb{E}[X_i] = \mu_i$ and $X_i$ is sub-Gaussian with parameter $\sigma_i$. Then for all $t \geq 0$, we have

$$\Pr\left[\left|\sum_{i=1}^n (X_i - \mu_i)\right| \geq t\right] \leq 2\exp\left(-\frac{t^2}{2\sum_{i=1}^n \sigma_i^2}\right).$$

(c) Let $X_1, \ldots, X_n \sim X$ be i.i.d. sub-Gaussian random variables with variance proxy $\sigma^2$. Then, for any $\epsilon \geq 0$ and $\delta \in [0,1]$ we have

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n X_i - \mathbb{E}X \geq \epsilon\right) \leq e^{-n\epsilon^2/(2\sigma^2)}$$

and

$$\mathbb{P}\left(\frac{1}{n}\sum_{i=1}^n X_i - \mathbb{E}X < \sqrt{\frac{2\sigma^2 \log(1/\delta)}{n}}\right) \geq 1 - \delta.$$

# 2 UCB

## 2.1 The Upper Confidence Bound Algorithm

In the multi-armed bandit problem with independent 1-sub-Gaussian values, we aim to minimize the regret. We define the UCB index as follows:

$$\text{UCB}_i(t-1,\delta) = \begin{cases} \infty & \text{if } T_i(t-1) = 0 \\ \hat{\mu}_i(t-1) + \sqrt{\frac{2\log(1/\delta)}{T_i(t-1)}} & \text{otherwise.} \end{cases}$$

where $\hat{\mu}_i(t)$ is the empirical mean of values sampled from arm $i$ and $T_i(t)$ is the number of times which arm $i$ is selected until time $t$.

**Algorithm 1** UCB Algorithm

```
Input: Number of arms $K$, confidence parameter $\delta$
Pull each arm once to initialize
for $\ell = 1, 2, \ldots$ do
    Let $t$ be the current time step
    Compute $A_\ell = \arg\max_{i \in [K]} \text{UCB}_i(t-1,\delta)$
    Pull arm $A_\ell$
end for
```

(a) [Regret decomposition lemma] Prove that for any policy $\pi$ and stochastic bandit environment $\nu$ with $\mathcal{A}$ finite or countable and horizon $n \in \mathbb{N}$, the regret $R_n$ of policy $\pi$ in $\nu$ satisfies

$$R_n = \sum_{a \in \mathcal{A}} \Delta_a \mathbb{E}[T_a(n)]$$

where $\Delta_i = R_{\text{max}} - R_i$ is the regret for arm $i$.

(b) Assuming $\delta$ is a fixed constant $C$, try to find a bad event in which results in linear regret after a large step $K$. Induce how we should choose $\delta$ to avoid this scenario.

**Definition of the Good Event $G_i$:**
Without loss of generalization assume that the first arm is optimal. Let $G_i$ be the event for suboptimal arm $i$ in the UCB algorithm, defined as:

$$G_i = \left\{ \mu_1 < \min_{t\in[n]} \text{UCB}_1(t,\delta) \right\} \cap \left\{ \hat{\mu}_{i,u_i} + \sqrt{\frac{2\log(1/\delta)}{u_i}} < \mu_1 \right\}$$

where $\mu_1$ is the true mean of the optimal arm, $u_i$ is a large enough time (a constant which is chosen later) and $\hat{\mu}_{iu_i}$ is the empirical mean of arm $i$ based on the first $u_i$ observations $(\frac{1}{u_i}\Sigma_{j=1}^{u_i} X_j)$. Event $G_i$ occurs when the UCB for the optimal arm is never underestimated and the UCB for arm $i$ is lower than the optimal arm's payoff after $u_i$ observations, making arm $i$ unlikely to be selected as optimal.

The event $G_i$ in the UCB algorithm is considered good because it ensures that the optimal arm (with mean $\mu_1$) is likely to be selected. The first condition ($\text{UCB}_1(t,\delta) > \mu_1$) prioritizes the optimal arm in selections. The second condition ($\text{UCB}_i < \mu_1$ after $u_i$ observations) limits the selection of the suboptimal arm $i$. In case $G_i$ does not occur the probability is insignificant and does not dominate the regret value.

(c) **Lemma:** Prove that if $G_i$ occurs, then arm $i$ will be played at most $u_i$ times: $T_i(n) \le u_i$.)

(d) Prove that:

$$\mathbb{E}[T_i(n)] \le u_i + P(G_i^c)n$$

(e) Assume that $u_i$ is chosen large enough that

$$\Delta_i - \sqrt{\frac{2\log(1/\delta)}{u_i}} \ge c\Delta_i$$

for some $c \in (0,1)$ to be chosen later.Show that:

$$\mathbb{P}\left( \hat{\mu}_{iu_i} + \sqrt{\frac{2\log(1/\delta)}{u_i}} \ge \mu_1 \right) \le \exp\left( -\frac{u_i c^2 \Delta_i^2}{2} \right).$$

(f) Also show:

$$\mathbb{P}(G_i^c) \le n\delta + \exp\left( -\frac{u_i c^2 \Delta_i^2}{2} \right)$$

(g) By choosing appropriate values for $u_i$ and also $c \in (0,1)$ show that:

$$\mathbb{E}[T_i(n)] \le 3 + \frac{16\log(n)}{\Delta_i^2},$$

.

(h) Show that if $\delta = \frac{1}{n^2}$ then

$$R_n \le 3\sum_{i=1}^{k} \Delta_i + \sum_{i:\Delta_i>0} \frac{16\log(n)}{\Delta_i},$$

(i) Show that:

$$R_n \leq 8\sqrt{nk\log(n)} + 3\sum_{i=1}^{k} \Delta_i.$$

(**Hint:** Break the sum in the previous part with respect to a term $\Delta$.)

## 2.2 Power of 2 version of UCB Algorithm$^*(Bonus)$

**Attention:** Only one of the bonuses is graded. **Attention2:** This is a hard problem, we will assign points to your try and errors even if you don't find a good bound.

Fix a 1-sub-gaussian $K$-armed bandit environment and a horizon $n$ consider the version of UCB that works in phases of exponentially increasing length of 1,2,4,. . . . in each phase, the algorithm uses the action that would have been chosen by UCB at the beginning of the phase. This method is useful in scenarios where rewards may be observed with delay or there is a communication cost for observing results.

**Algorithm 2** UCB Power-of-2 Variant

```
Input: Number of arms $K$, confidence parameter $\delta$
Pull each arm once to initialize
for $\ell = 1, 2, \ldots$ do
    Let $t$ be the current time step
    Compute $A_\ell = \arg\max_{i\in[K]} \text{UCB}_i(t-1, \delta)$
    Pull arm $A_\ell$ exactly $2^\ell$ times
end for
```

Find a regret bound for this version of UCB.

# 3 Online Learning

Consider this scenario; you have $n$ experts helping you make a decision. e.g. $n$ models deciding whether or not to show an ad to a user. In a simple case, you can consider this a multi-armed bandit problem. But a limiting factor in this case is that we do not consider the changes to arms during the time. e.g. suppose that each model gets updated after making a mistake. So we explore solutions to show the effectiveness of our model in more complex scenarios. Pay attention that we may need to define a new sense of regret when we face the problem.

## 3.1 Randomized Weighted Majority Algorithm

The RWM algorithm is a method for making predictions based on a set of experts. The experts' predictions are weighted based on their past performance, and in each round, an expert is chosen to make a prediction based on these weights. The weight of each expert is updated after each round based on whether the expert was correct or incorrect.

The steps for the RWM algorithm are:

1. **Assigning Weights**: Assign a weight $w_i(t)$, where $i \in [N]$, to each expert at each round. Pick expert $i$ with probability proportional to their weight $w_i(t)$. Also $w_i(0) = 1$ for each expert.

2. **Prediction**: Let $i_t$ denote the chosen expert at round $t$. the experts make a decision which is wrong or correct.

3. **Update Weights**: Upon observing the outcome $o_t$, the weights are updated as follows:

$$w_i(t+1) = \begin{cases} w_i(t) & \text{if expert } i_t \text{ is correct} \\ w_i(t) \cdot (1-\epsilon) & \text{if expert } i_t \text{ is wrong} \end{cases}$$

(a) Total weight is updated as follows:

$$\mathbb{E}[S_{t+1}] = \mathbb{E}[S_t] \cdot (1 - \epsilon \cdot P(\tilde{m}_t = 1))$$

where $S_t = \sum_{i=1}^{N} w_i(t)$, and $\tilde{m}_t$ indicates whether a mistake occurs at iteration $t$.

(b) After $T$ rounds, total weight satisfies:

$$\mathbb{E}[S_{T+1}] \leq N \cdot e^{-\epsilon \sum_{t=1}^{T} P(\tilde{m}_t=1)}$$

(c) Expected total mistakes are bounded by:

$$E[M] \leq (1+\epsilon)M_i + \frac{\ln N}{\epsilon}, \quad \forall i \in [N].$$

where $M_i$ indicates the number of mistakes expert $i$ makes and $M$ is the total number of errors (use $w_i(t) < S_t$)

(d) Final bound on expected total mistakes is:

$$E[M] \leq \min_{i \in [N]} M_i + 2\sqrt{T \ln N}$$

. where $T$ is current time step (use $M < T$). Explain the relation, and whether it is a good regret bound or not.

## 3.2 Hedge Algorithm$^*$$(Bonus)$

**Attention:** Only one of the bonuses is graded.

The Hedge Algorithm is designed for the **dot-product game**, a general framework for online decision-making with expert advice. This setting extends binary outcome prediction by allowing flexible loss values, accommodating a wide range of applications.

### Problem Setting

+ **Learner**: At each round $t = 1, \ldots, T$, the learner selects a probability vector $\mathbf{p}_t = (p_{t1}, p_{t2}, \ldots, p_{tN})$, where $p_{ti} \geq 0$ and $\sum_{i=1}^{N} p_{ti} = 1$, representing a distribution over $N$ experts.

+ **Adversary**: Simultaneously, the adversary chooses a loss vector $\ell_t = (\ell_{t1}, \ell_{t2}, \ldots, \ell_{tN})$, where $\ell_{ti} \in \mathbb{R}$ is the loss associated with expert $i$ and also, $\ell_{ti} \in [-1, 1]$.

+ **Loss**: The learner incurs a loss given by the dot product:

$$\ell_t \cdot \mathbf{p}_t = \sum_{i=1}^{N} \ell_{ti} p_{ti}.$$

This loss represents the expected loss over the chosen distribution of experts.

## Hedge Algorithm

The Hedge Algorithm maintains weights for each expert, updating them based on observed losses to favor experts with lower cumulative losses. The steps are as follows:

1. **Initialization**: Set initial weights for all experts $i = 1, \dots, N$:

$$w_i(1) = 1.$$

2. **For each round** $t = 1, \dots, T$:

   + **Choose distribution**: Compute the probability distribution over experts:

$$p_t(i) = \frac{w_t(i)}{\sum_{j=1}^{N} w_t(j)}.$$

   + **Observe loss**: Receive the loss vector $\ell_t$ from the adversary.

   + **Update weights**: Adjust the weight of each expert $i$ using the exponential update rule:

$$w_{t+1}(i) = w_t(i) \cdot e^{-\epsilon \ell_{ti}},$$

   where $\epsilon > 0$ is the *learning rate*, controlling the sensitivity to losses.

(a) The total weight at round $t+1$ is:

$$S_{t+1} = \sum_i w_{t+1}(i)$$

then show that

$$S_{t+1} \leq S_t \left(1 - \epsilon \sum_i p_t(i)\ell_t(i) + \epsilon^2 \sum_i p_t(i)\ell_t^2(i)\right)$$

(b) The regret of the Hedge algorithm is the difference between the cumulative loss of the learner and the cumulative loss of the best fixed strategy $p$ over all rounds:

$$\text{Regret} = \sum_{t=1}^{T} \ell_t \cdot p_t - \min_{p \in \Delta^N} \sum_{t=1}^{T} \ell_t \cdot p$$

Using the weight update rule and the loss bounds, the regret can be written as:

$$\text{Regret} \leq \frac{\log n}{\epsilon} + \epsilon \sum_{t=1}^{T} \ell_t^2 \cdot p_t$$

Compare this bound with RWM algorithm.

(c) Finally:

$$\text{Regret} \leq \frac{\log n}{\epsilon} + \epsilon T \leq 2\sqrt{T \log n}$$