Imperial College
London

# Multi-Armed Bandit: a short introduction

Alexis Laignelet

Imperial College London

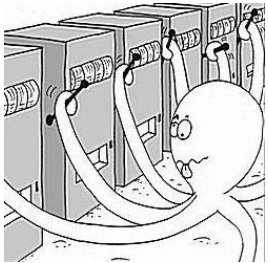
08/06/20

# Table of contents

1. Introduction
2. Stochastic bandits
3. Bayesian framework
4. Contextual bandits
5. Recent papers
6. Conclusion

# Bandits

- **Multi-armed bandits** are everywhere:
  - clinical trials,
  - A/B testing,
  - ad placement,
  - recommender system,
  - dynamic pricing,
  - ...
- There are enjoying a revival, following the latest booming in reinforcement learning.
- A simple setup, but powerful and very rich mathematically.



Drawing showing the idea of pulling multiple arms.

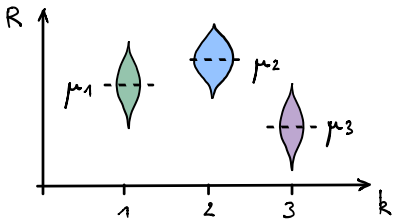
# General setup

The setup is defined by:

- a finite action set $\mathcal{A} = \{1, 2, ..., k\}$ corresponding to the arms,
- for each $a \in \mathcal{A}$, there is an **unknown** distribution $P_a$ with mean $\mu_a$.

From the learner perspective:

- activate arm $A_t \in \mathcal{A}$ and observe a reward $R_t \sim P_a$,
- maximise the **total reward** $\sum_{t=1}^{n} R_t$.




| $A_t$ | 1 | 3 | 2 | 2 | 1 | 2 |
|---|---|---|---|---|---|---|
| $R_t$ | 1.0 | 0.3 | 2.7 | 1.2 | 0.8 | 0.3 |

An example of unknown distributions and a sequence of actions/rewards.

# Learning objective

- The idea is to find the distribution with the **higher mean** $\mu^*$.
- The **optimal action** is defined as $a^* = \arg\max_a \mu_a$.
- Strategies are evaluated by a metric called the **regret** (the lower the better):

$$\mathcal{R}_n = n\mu^* - \mathbb{E}\left[\sum_{t=1}^{n} R_t\right]$$

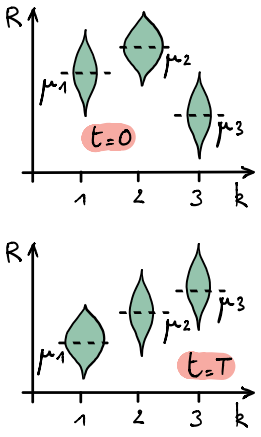
  that evaluate the cost of not knowing $\mu^*$.
- How do we choose the next action? Trade-off **exploration vs exploitation**.

# Trade-off Exploration vs Exploitation

- **Exploration**: trying different actions to make sure we do not miss the best one.
- **Exploitation**: choosing the same action over and over to maximise our reward.

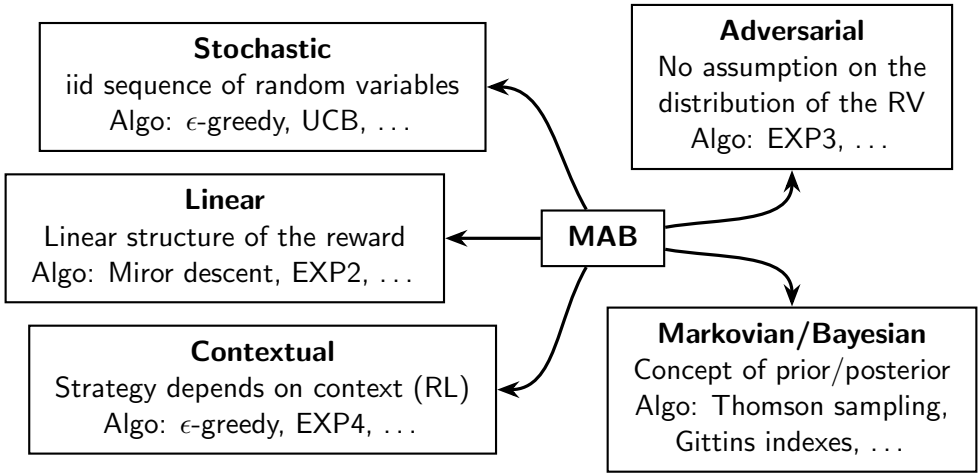
There are two main problems with **exploitation**:

- The rewards are sampled from a distribution, so even the arm with the best mean $\mu^*$ can still provide poor rewards sometimes.
- Distributions may change with time, and our **policy** has to adapt. It is the particular case of **non-stationary** bandits.




Distributions evolving with time.

# Main types of bandits



# Framework

## Stochastic assumptions

Let $k$ different **arms**, and $n \geq k$ rounds. The **probability distribution** $P_1, \ldots, P_k$ are **unknown**. The process is the following:

1. the learner chooses $a \in \{1, \ldots, k\}$,
2. the reward $R_{a,t}$ is given **independently** from the past.

Examples of possible distributions:

- $P_a \sim \mathcal{B}(\mu_a)$: Bernoulli with unknown mean $\mu_a \in [0, 1]$,
- $P_a \sim \mathcal{N}(\mu_a, 1)$: Gaussian with unit variance and unknown mean $\mu_a \in \mathbb{R}$,
- $P_a$ is sub-Gaussian (tails dominated by Gaussian),
- ...

# $\epsilon$-greedy algorithm

## $\epsilon$-greedy policy

At each round the best **greedy action** is selected (the one the largest empirical **expected reward**), but with probability $\epsilon$, a random action is chosen (excluding the best greedy one). The **action** $\boldsymbol{A_t}$ taken at round $t$ is:

$$A_t = \begin{cases} \arg\max_a \{\mathbb{E}[\mu_a]\} & \text{with probability } 1-\epsilon \\ a \text{ randomly} & \text{with probability } \epsilon \end{cases}$$

The **greater** $\boldsymbol{\epsilon}$, the greater the **exploration**. Variants exist where $\boldsymbol{\epsilon}$ **decays** with time to encourage exploration only at the beginning.

# Example (1/3)

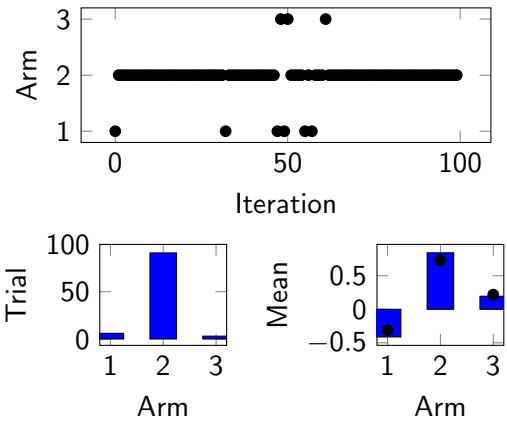
Let 10 arms with **Gaussian distribution** $\mathcal{N}(\mu_a, 1)$. The algorithm is the following:

1. choose arm according to $\epsilon$-**greedy policy**,
2. get reward $R_{a,t}$ from arm $a$ at time $t$,
3. update **expected reward** according to: $\mathbb{E}[\mu_a] \leftarrow \mathbb{E}[\mu_a] + \alpha(R_{a,t} - \mathbb{E}[\mu_a])$ with $\alpha = 1/T_a$ and $T_a$ the number of time action $a$ has been chosen.

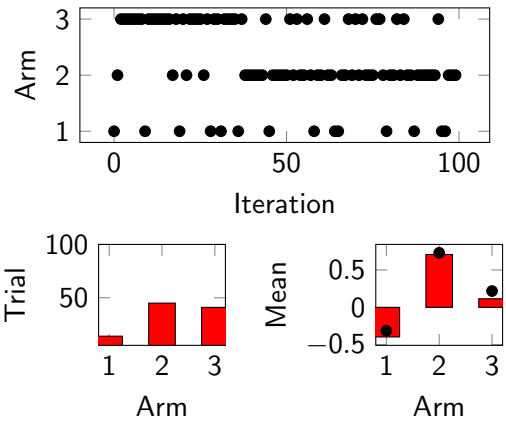
## Hyperparameters

- $\epsilon$: controls the **exploration**. High values of $\epsilon$ lead to better exploration.
- $\alpha$: controls the importance given to **present rewards** as opposed to past ones. High values of $\alpha$ mean a tendency to **forget** very quickly. Can be useful for **non-stationary** bandits.

# Example (2/3)

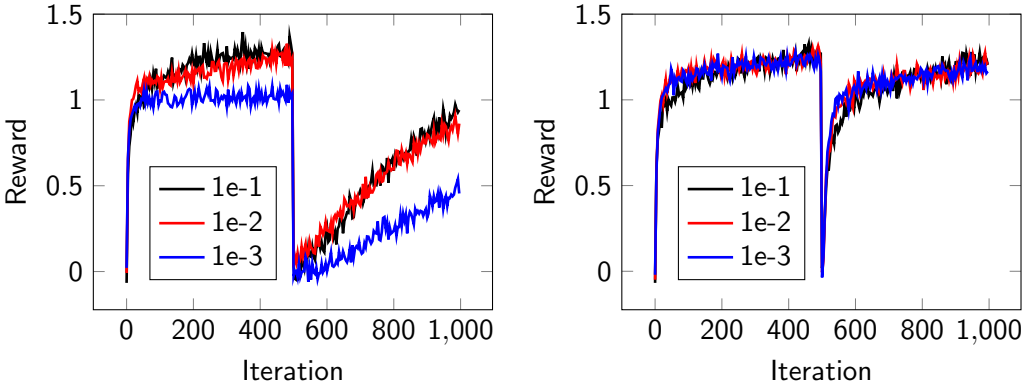



Arm chosen (on top), trials per arm (on the left), and evaluated/real (dot) mean (on the right) for $\epsilon = 0.1$.




Arm chosen (on top), trials per arm (on the left), and evaluated/real (dot) mean (on the right) for $\epsilon = 0.5$.

# Example (3/3)




Reward for $\epsilon \in \{0.1, 0.01, 0.001\}$ through 1000 iterations, averaged over 1000 experiences for a 10-armed bandit with unit variance and unknown means. On the left $\alpha$ is equals to the number of trials per arms whereas on the right $\alpha = 0.1$ which enables the learner to quickly learn the new best distribution after a random permutation between arms that occurs at iteration 500.

# Upper Confidence Bound

- Intuition: the more we sample from an arm, the more **confident** we are in our evaluation of its mean.
- Using **Hoeffding inequality**, with $u$ the confidence radius:

$$\mathbb{P}\left(\mu \geq \mathbb{E}[\mu] + u\right) \leq e^{-2Tu^2}$$

If instead, we define $\delta$ such that:

$$\mathbb{P}\left(\mu \geq \mathbb{E}[\mu] + \sqrt{\frac{\log(1/\delta)}{2T}}\right) \leq \delta$$

And then, choosing $\delta = 1/t^4$:

$$\mathbb{P}\left(\mu \geq \mathbb{E}[\mu] + \sqrt{\frac{2\log(t)}{T}}\right) \leq \frac{1}{t^4}$$

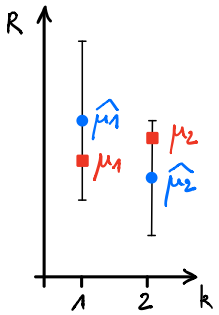
# Toward an algorithm: UCB1

## UCB1 policy

At each round the arm with the largest **Upper Confidence Bound** is selected.

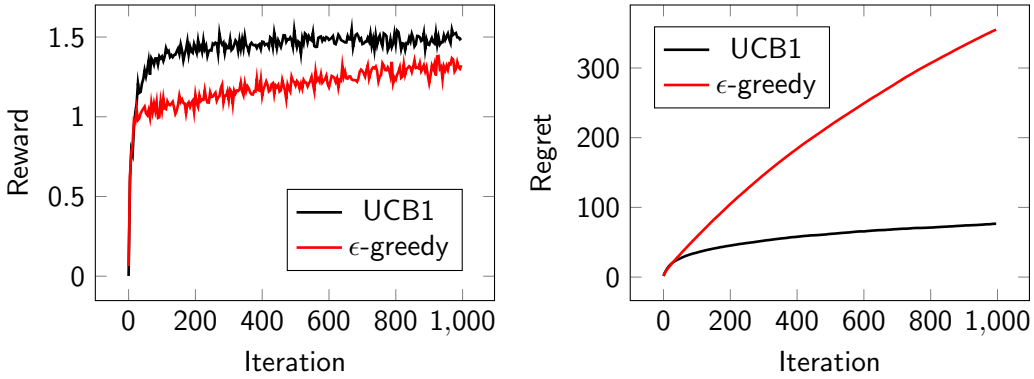
$$A_t = \arg\max_a \left\{ \mathbb{E}[\mu_a] + \sqrt{\frac{2\log(t)}{T_a}} \right\}$$

- When $T_a$ increases, our knowledge becomes more accurate and the confidence radius decreases.
- On the other hand, the confidence radius still increases with time $t$, letting an opportunity for the non top arms to be selected again later on.




Even if the second arm has a better mean, the algorithm chooses to pull the first one here, since it has not tried it a lot and could still provide the best rewards.

# Example




Reward (on the left) and regret (on the right) for UCB1 and $\epsilon$-greedy policies with $\epsilon = 0.01$ through 1000 iterations, averaged over 1000 experiences for a 10-armed bandit with Gaussian (unit variance and unknown means). Note that, contrary to $\epsilon$-greedy policy, there is no hyperparameter to tune for UCB1.

# Framework

## Bayesian assumptions

Let $k$ different **arms**, and $n \geq k$ rounds. In the Bayesian framework, the probability distribution $P_1, \ldots, P_k$ are **likelihoods** of the model, depending on a **prior**. The process is the following:

1. the learner makes an assumption on the **likelihood** and **prior distributions**,
2. the reward $R_t \sim$ likelihood provides information to update his belief on the **prior** and ultimately the probability distributions $P_1, \ldots, P_k$

- The **Bayesian regret** is defined as:

$$BR = \int \mathcal{R} dQ$$

  where $Q$ is the prior.
- We usually chose a **conjugate prior** so that the **posterior** is tractable.

# Thompson sampling

## Thompson sampling or Posterior sampling

At each round $t$, sample from the **posterior distribution**. The arm that gives the higher result is selected:

$$A_t = \arg\max_a \{\mathbb{P}(\theta_a|X_a)\}$$

where $\theta$ is our prior set of parameters.

Practically speaking, the idea is to update the **prior distribution** with the posterior distribution. This means updating our belief on the prior taking into account the lastest data:

$$\mathbb{P}(\theta|X) = \frac{\mathbb{P}(X|\theta)\mathbb{P}(\theta)}{\mathbb{P}(X)}$$

# Example (1/2)

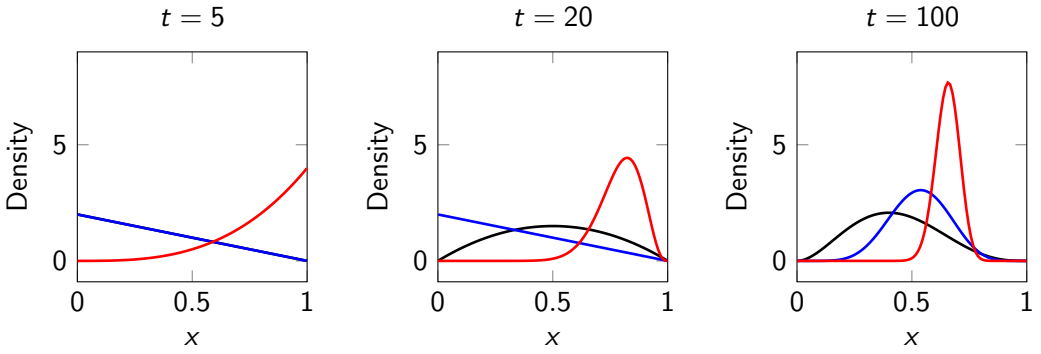
Let a **Bernoulli bandit** with three arms. This means the likelihood of the model is Bernoulli distributed. The conjugate prior is the **Beta distribution** which characterises our belief on the mean of the Bernoulli law (its unique parameter). In this case, the posterior is also a Beta distribution.

1. choose arm according to **Thompson sampling**,
2. get reward $R_{a,t} \sim \mathcal{B}(\mu_a)$ from arm $a$ at time $t$,
3. assuming the prior is $\mu_a \sim Beta(\alpha, \beta)$, update the **posterior distribution** according to:

$$\mu_a | R_{a,t} \sim \begin{cases} Beta(\alpha + 1, \beta) \text{ if } R_{a,t} = 1 \\ Beta(\alpha, \beta + 1) \text{ if } R_{a,t} = 0 \end{cases}$$

4. take the posterior as the prior of the next step.

# Example (2/2)




Density of the prior of the three arms for different value of $t$. Note that at the very beginning, the densities correspond to uniform laws $[0, 1]$ since we do not have any information. With time, your knowledge increases so the mean of the Bernoulli law we try to evaluate becomes more accurate. Note also, since we want to exploit the arm giving the best reward, your knowledge becomes more and more precise on this particular arm, leading to a very narrow distribution (in red).

# Framework

## Contextual assumptions

Let $k$ different **arms**, and $n \geq k$ rounds. The **probability distribution** $P_1, \ldots, P_k$ are **unknown** but depend on the a **context** $x$. The process is the following:

1. the learner observes a context $x_t$,
2. the learner picks an arm $a \in \{1, \ldots, k\}$,
3. the reward $R_{a,t}$ is realised.

- One simple idea is to use independent algorithm for every context.
- The expect reward is now the following:

$$\mathcal{R} = REW(\pi^*(x)) - \sum_{t=1}^{n} \mu(a_t|x_t)$$

where $REW(\pi^*(x))$ is the reward of the best policy.

# Example(1/3)

Let a **Bernoulli bandit** with two arms. In this setup, the **context** influences the probabilities of the Bernoulli law of the arms. We model this effect by defining the probability as a logistic regression of the context $x \in \mathbb{R}$, where the parameters $\theta$ are unknown for the learner and $\epsilon \sim \mathcal{N}(0, 1)$.

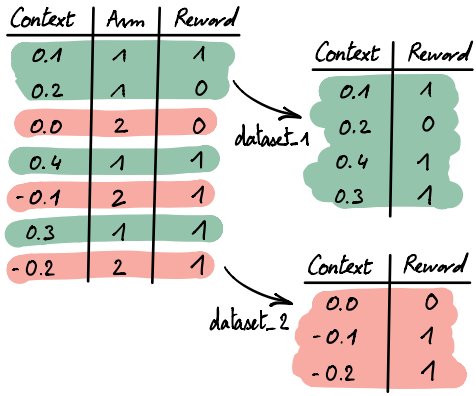
$$p_a(x) = \frac{1}{1 + e^{-\theta_a x - \epsilon}}$$

Again, for convenience, the context is only model by a unique variable $x$, and the logistic regression has only one parameter.

# Example(2/3)

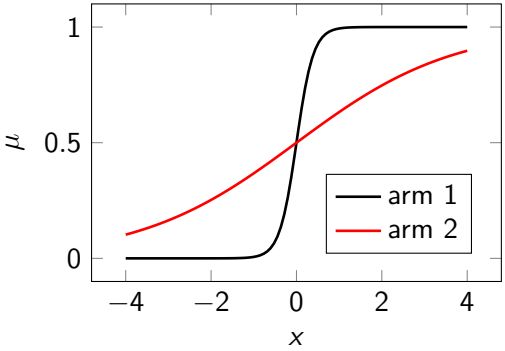
The procedure is the following:

1. Observe **rewards** for different combinations of arms and contexts.
2. Create a dataset for each arm by taking subsets of the collected data.
3. Train a model on each dataset (**supervised learning**), and predict the reward for the current context.
4. Pull the arm giving the higher **prediction**.

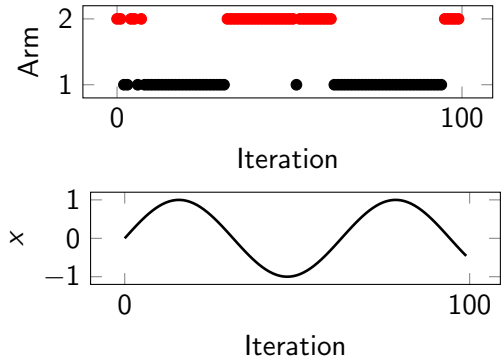



Construction of the two datasets through time.

# Example (3/3)




The mean of both arms is model as the sigmoid function of the context $x$. For $x \geq 0$, the best strategy is to pull arm 1 whereas arm 2 provides the best reward when $x < 0$.




Arm chosen (on top) depending on the context (bottom). Choosing arm 1 for $x \geq 0$ leads to a better strategy, whereas puling arm 2 when $x < 0$ is optimal.

# Restricted context

Bouneffouf, D., Rish, I., Cecchi, G. A., & Féraud, R. (2017). Context attentive bandits: Contextual bandit with restricted context. *arXiv preprint arXiv:1705.03821*.

- Main **assumptions**: $k$ arms, Bernoulli bandits, context $x \in \mathbb{R}^n$, Thompson sampling, no machine learning algorithm mapping the context to the reward.
- How to allocate a limited budget to access a **subset of features** of the context variable? Modify the Thompson sampling algorithm by assuming ***Beta* distribution**poin for each feature, and take the subset giving the largest prediction.

# Nonparametric

Guan, M. Y., & Jiang, H. (2018, April). Nonparametric stochastic contextual bandits. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

- Main **assumptions**: 2 arms, context $x \in \mathbb{R}^2$, UCB policy, reward functions $f_i(x) = 1$ if $x \in R_i$, $f_i(x) = 0.5$ otherwise.
- **Nonparametric** means no strong assumption on the form of the mapping function.
  - $(+)$ flexibility, power, performance,
  - $(-)$ more data, slower, prone to overfitting.

  eg: k-NN, Decision Trees, Support Vector Machines.

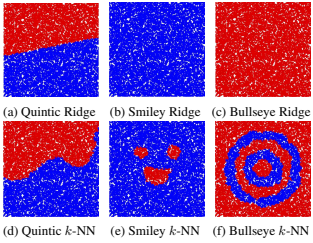



Figure from the above paper: top-arm identification using Ridge regression and 25-NN regression.

# Stochastic delays

Zhou, Z., Xu, R., & Blanchet, J. (2019). Learning in generalized linear contextual bandits with stochastic delays. In *Advances in Neural Information Processing Systems* (pp. 5198-5209).

- Main **assumptions**: $k$ arms, linear bandit, context $x \in \mathbb{R}^d$, stochastic reward $\in [0, 1]$, delayed rewards.
- How to take a decision when the reward is delayed in a stochastic way? Develop a policy based on UCB by deriving properties on the delay, view as a random variable.

# To sum up

This presentation covers:

- stochastic bandits: case of Gaussian with unknown mean $\mathcal{N}(\mu_a, 1)$,
- Bayesian framework: particular case of Bernoulli bandits,
- contextual bandits: application on Bernoulli bandits modelled by logistic regression and univariate context.

The field is way richer:

- other frameworks: linear, Lipschitz, **adversarial**, combinatorial, **non-stationary**, ect,
- other aspects: convergence, algorithms, policies, etc,
- received a lot of attention recently: many papers about multi-armed bandit.

# References


- Bubeck, S. and Cesa-Bianchi, N. (2012).
  Regret analysis of stochastic and nonstochastic multi-armed bandit problems.
  *arXiv preprint arXiv:1204.5721.*
- Lattimore, T. and Szepesvari, C. (2020).
  *Bandit Algorithms*.
  Cambridge University Press.
- Slivkins, A. (2019).
  Introduction to multi-armed bandits.
  *arXiv preprint arXiv:1904.07272.*