2. Multi-armed bandits

Possible sequential choices of a bee: land on a white or a yellow flower

Possible sequential choices of a bee: land on a white or a yellow flower

The choice is based on previous experience $H_{t\text{-}1} = \{a_1, r_1, \ldots, a_{t\text{-}1}, r_{t\text{-}1}\}$ with the aim of maximizing expected rewards (in drops of nectar).

Rewards are typically random and unknown in advance:

$$f(r_w|a = \text{white}) \qquad f(r_y|a = \text{yellow})$$

Online decision-making involves a trade-off between:

- Exploitation, take most rewarding action, given the current information
- Exploration, gather more information to take better decisions in the future

through a number of trials t (earning vs. learning, control vs. estimation)

Getting to know the best long-term strategy may involve short-term sacrifices.

Settings

Stochastic environment: rewards are sampled from an unknown product distribution. Rewards are i.i.d.

Adversarial environment: rewards are chosen deterministically by an adversary which, at time t, possibly knows all the past, but not the currently selected arm.

Examples

- Medical treatment

Exploitation	Use the known treatment

Exploration	Try an experimental one

- On line advertising

Exploitation	Show the most succesful advert

Exploration	Show a different advert

- Oil drilling

Exploitation	Drill at the best known location

Exploration	Drill a new location

- Game playing

Exploitation	Play the move you believe is best

Exploration	Play an experimental move

- Sales planning

Exploitation	Use last season strategy

Exploration	Try new offers

Which ones are stochastic and which ones adversarial?

The stochastic m-armed bandit problem

Let us formalize the concept…

A stochastic multi-armed bandit is defined by a tuple $\langle \mathcal{A}, \mathcal{R} \rangle$

- $\mathcal{A}$ is a set of $m=|\mathcal{A}|$ actions (or "arms")
- At each step t
  - agent selects one action $a_t \in \mathcal{A}$
  - environment generates a reward r_t
- The reward follows an unknown probability density $r \sim f(r|a)$
- Actions do not affect future rewards
- The objective is to take appropriate decisions so as to maximize the cumulative reward

$$\sum_{\tau=1}^{t} r_\tau$$

- The action-value is the mean reward for action a: $Q(a) = E\{r \mid A = a\}$

- The optimum value is $V^* = Q(a^*) = \max_{a \in \mathcal{A}} Q(a)$

- The regret is the opportunity loss for one step: $l_t = E\{V^* - r_t\}$

- The total regret is the total opportunity loss

Number of pulls for arm a up to time t

$$L_t = E\left\{\sum_{\tau=1}^{t} V^* - r_\tau\right\} = \sum_{a \in \mathcal{A}} E\{N_t(a)\}\left(V^* - Q(a)\right)$$
$$= \sum_{a \in \mathcal{A}} E\{N_t(a)\}\,\Delta_a$$

Minimizing the total regret is equivalent to maximizing the cumulative reward.

A good algorithm ensures small counts for large gaps Δ_a.
But gaps are unknown!

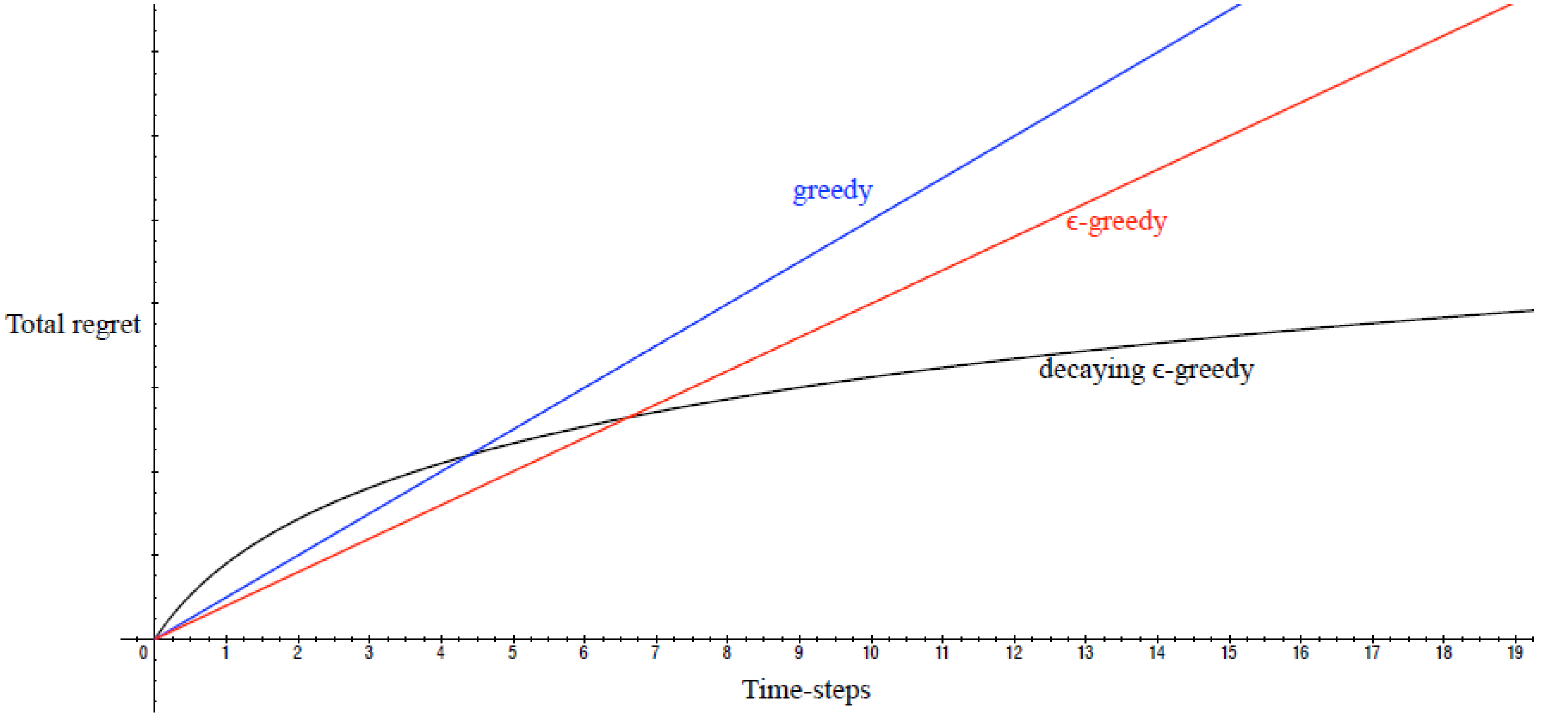
The problem is that we do not know the value of each action. We need to guess it from experience.

At any time step, there are actions that are better than the rest: the *greedy* actions. If we adopt them, we **exploit**. If we choose other, we **explore**.

Both cannot be simultaneous: exploring is worse in terms of reward in the short term, but pays off in the long term.

Let us see how to balance between them, four approaches:

1. Random exploration
2. Upper-confidence-bound action selection
3. Soft-max strategy and gradient bandits
4. Probability matching



From David Silver, Course on RL, 2015

Is it possible to achieve sub-linear total regret? Yes!

Action-value estimation

How to estimate values of actions? By averaging the observed rewards:

$$\hat{Q}_t(a) = \frac{1}{N_t(a)} \sum_{\tau=1}^{t} r_\tau^a$$

Number of pulls for arm *a* up to time *t*

Greedy action selection: $a_t = \arg\max_a \hat{Q}_t(a)$

The agent can get stuck in a non-optimal action for ever.

The regret grows linearly in time.

Approach 1. Random exploration

Random exploration technique:

ε-greedy action selection:
$$a_t = \begin{cases} \arg\max_a \hat{Q}_t(a) & \text{with probability } 1-\varepsilon \\ \mathit{random\ action} & \text{with probability } \varepsilon \end{cases}$$

We will be making errors and regret will be linear (but every possible action will be sampled an infinite number of times as $t \to \infty$), unless…

…taking a decaying schedule for ε:

$$\varepsilon_t = \min\left\{1, \frac{\delta|\mathcal{A}|}{t}\right\} \qquad \delta > 0$$

we obtain a logarithmic asymptotic total regret provided that

$$\Delta_a = V^* - Q(a) \qquad\qquad \delta \geq \frac{1}{\min_{a \neq a^*} \Delta_a^2}$$

Best performance if δ is close to this bound [Auer 2002]

Simple ε-greedy bandit algorithm

Initialize

for $a = 1{:}m$

$Q(a) = \text{rand}$

$N(a) = 0$

$t = 0$

Loop forever

$t \leftarrow t + 1$

Update ε

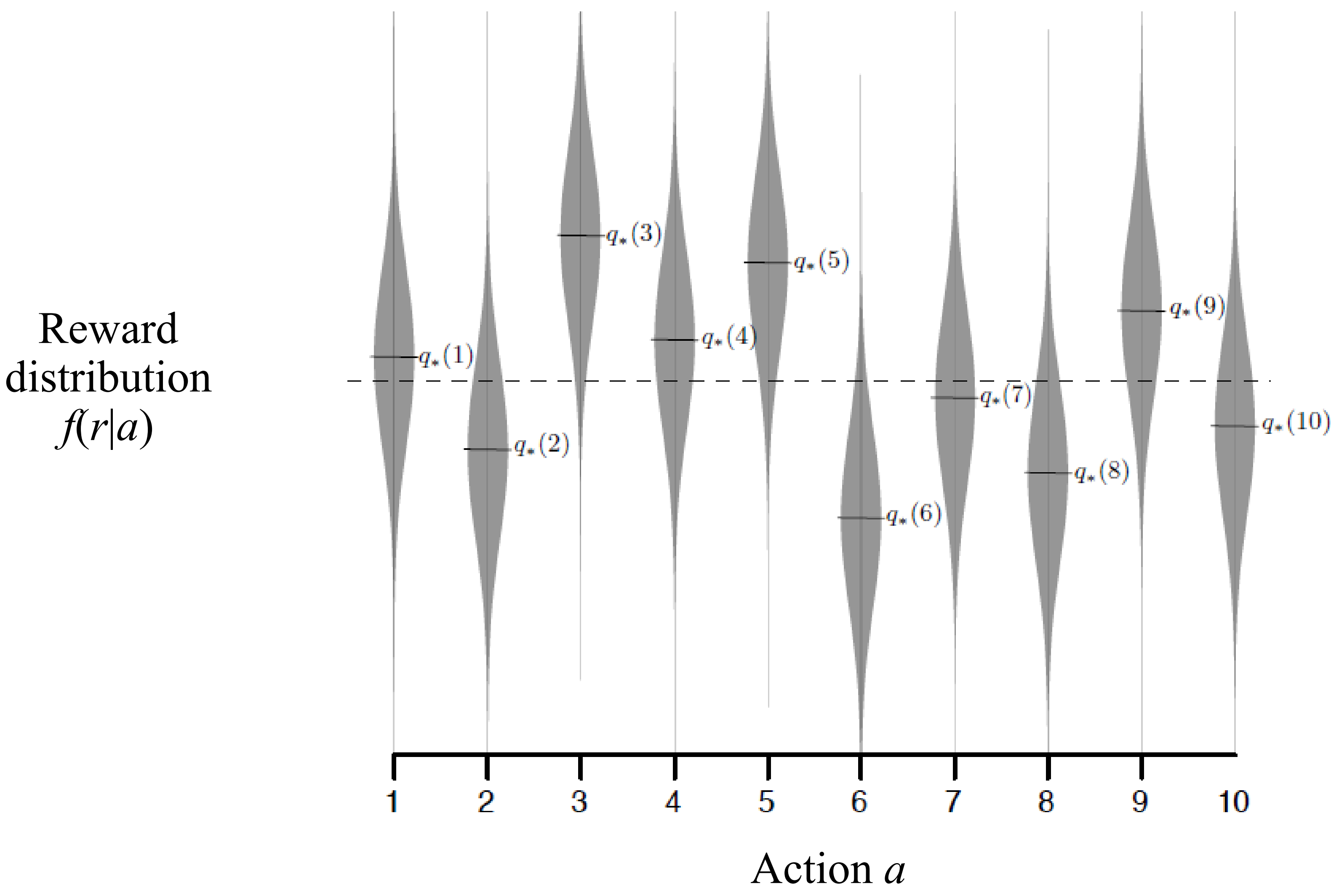
$$A \leftarrow \begin{cases} \arg\max_a Q(a) & \text{with probability } 1-\varepsilon \\ \textit{random action} & \text{with probability } \varepsilon \end{cases}$$

$r \leftarrow bandit(A)$

$N(A) \leftarrow N(A) + 1$

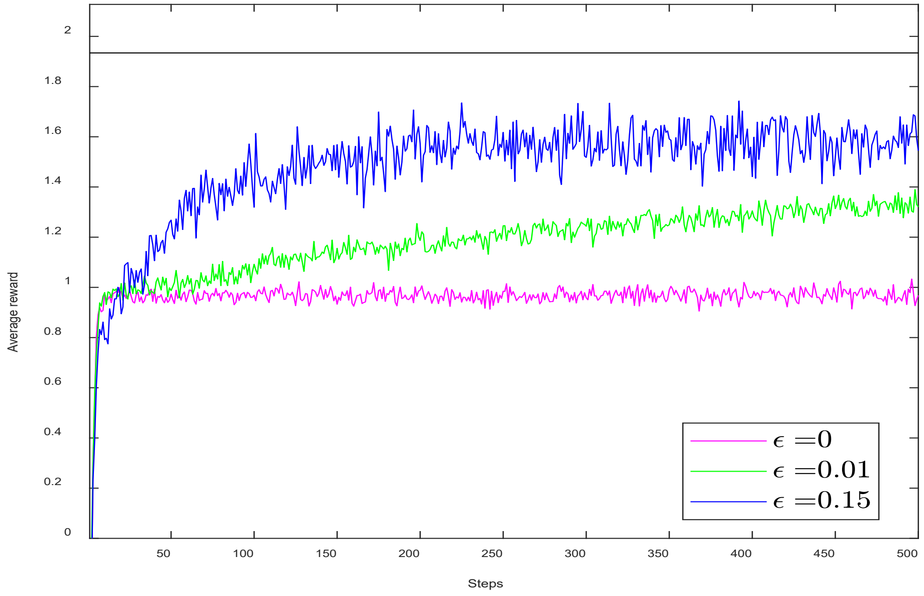
$$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\big(r - Q(A)\big)$$

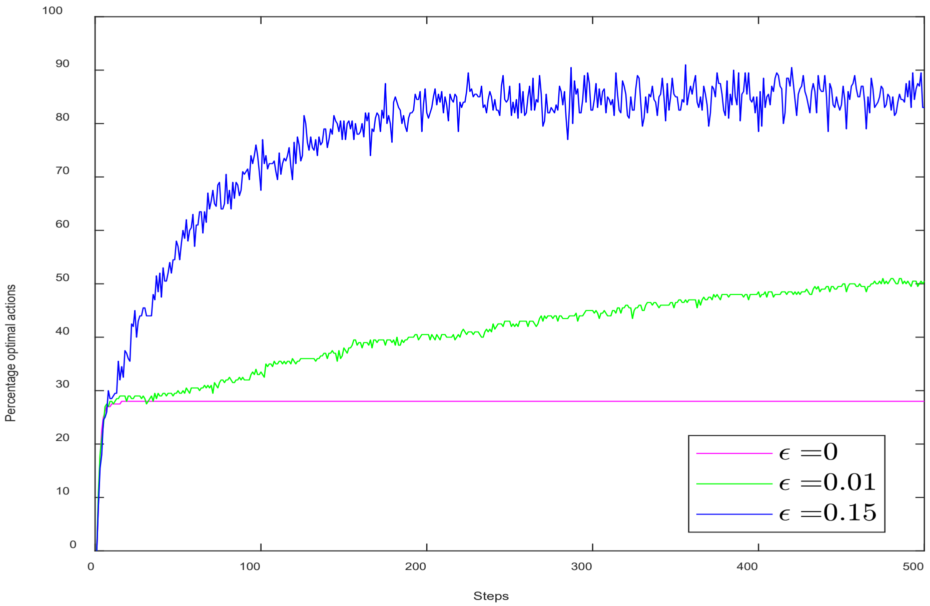
Example 2.1. A 10-armed bandit with unit-variance Gaussian rewards



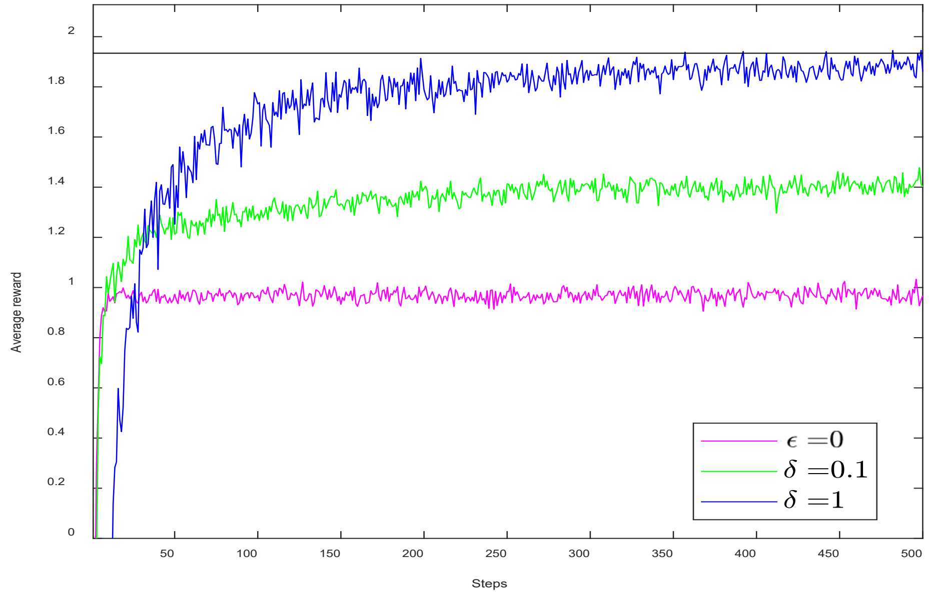
From Sutton & Barto, Reinforcement Learning: An Introduction, 2018

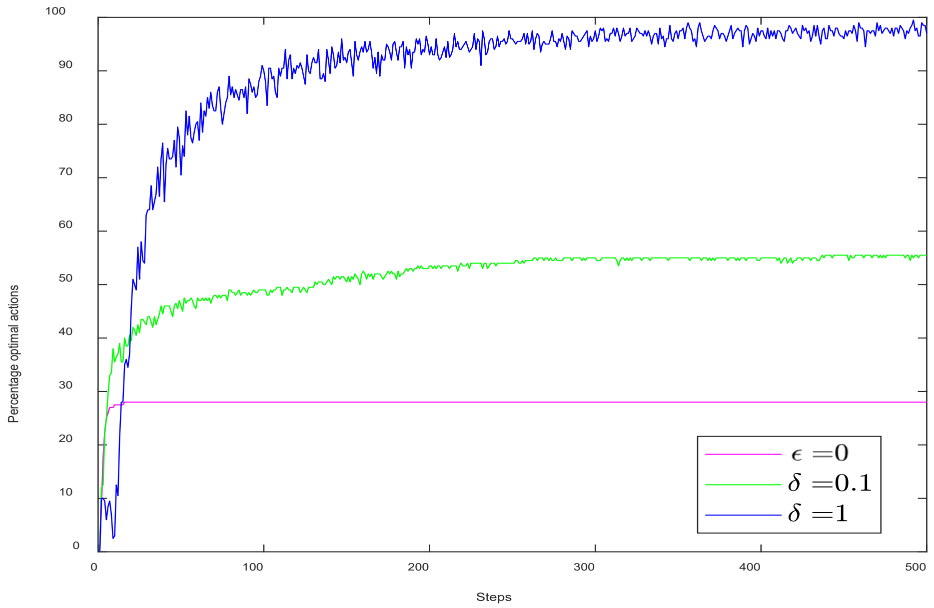
Example 2.1. Performance averaged over 200 runs with **constant ε**





Example 2.1. Performance averaged over 200 runs with **decaying ε**



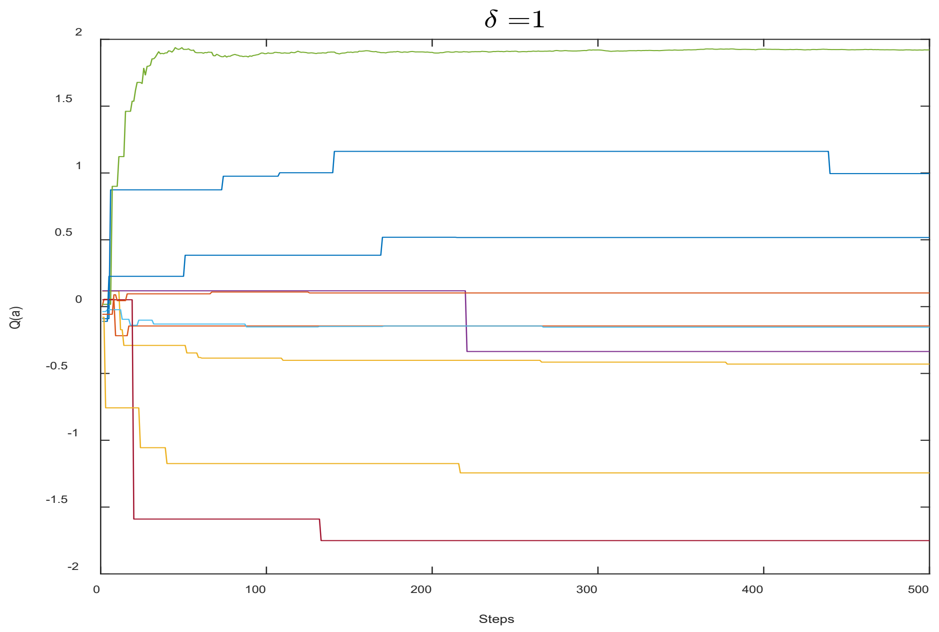


What would the performance be if variances were higher?

What if they were lower?

Example 2.1. Performance averaged over 200 runs with **decaying ε**

Evolution of the estimated $Q(a)$ on a single run



True means $E\{r|a\}$: 0.8782 -0.1503 -0.4655 -0.8348 1.9340
-0.2624 -2.5056 0.7514 0.1667 -1.4844

Check the bound on δ

Lower performance bound

The performance of any decision algorithm depends on the similarity between the pdf of optimal arm $f(r|a^*)$ and other arms: hard problems have similarly looking arms with different means.

Theorem (Lai and Robbins)

Asymptotical total regret is at least logarithmic in the number of steps

$$\lim_{t\to\infty} L_t \geq \log t \sum_{a\neq a^*} \frac{\Delta_a}{D\left(f(r\,|\,a)\,\|\,f(r\,|\,a^*)\right)} \approx O\left(\sum_{a\neq a^*}\frac{1}{\Delta_a}\right)$$

where D is the Kullback-Leibler divergence between two density functions, and

$$\Delta_a = V^* - Q(a)$$

Incremental implementation and tracking

Even if rewards were deterministic at any given time, some exploration is needed if average rewards change over time.

An implementation of estimated $Q(a)$ that requires constant memory and computation effort is:

$$\hat{Q}_{t+1}(a) = \frac{1}{N_t(a)} \sum_{i=1}^{N_t(a)} r_i = \frac{1}{N_t(a)} \left(r_t + \sum_{i=1}^{N_t(a)-1} r_i \right)$$

Number of times action a has been selected up to time t

$$= \frac{1}{N_t(a)} \left(r_t + \left(N_t(a)-1\right) \frac{1}{\left(N_t(a)-1\right)} \sum_{i=1}^{N_t(a)-1} r_i \right)$$

$$= \frac{1}{N_t(a)} \left(r_t + \left(N_t(a)-1\right) \hat{Q}_t(a) \right) = \frac{1}{N_t(a)} \left(r_t + N_t(a)\hat{Q}_t(a) - \hat{Q}_t(a) \right)$$

$$= \hat{Q}_t(a) + \frac{1}{N_t(a)} \underbrace{\left(r_t - \hat{Q}_t(a) \right)}_{\text{Error in the estimate}}$$

where r_i are the rewards obtained whenever a is the selected action.

If rewards change over time we can exploit this implementation giving more weight to recent rewards than to long-past rewards:

$$\hat{Q}_{t+1}(a) = \hat{Q}_t(a) + \alpha\left(r_t - \hat{Q}_t(a)\right)$$

where the step size $\alpha \in (0,1]$ is constant, and may be different for every action.

We can even change the value of α over the iterations. To ensure convergence to $Q(a)$ with probability 1:

$$\sum_{t=1}^{\infty} \alpha_t = \infty \qquad \sum_{t=1}^{\infty} \alpha_t^2 < \infty$$

These conditions are met for $\alpha_t = 1/t$.

For a constant step size the condition is not met indicating that the estimates never converge completely but continue to vary in response to the recently received rewards.

For a constant value of α, how does the estimate depend on the initial value?

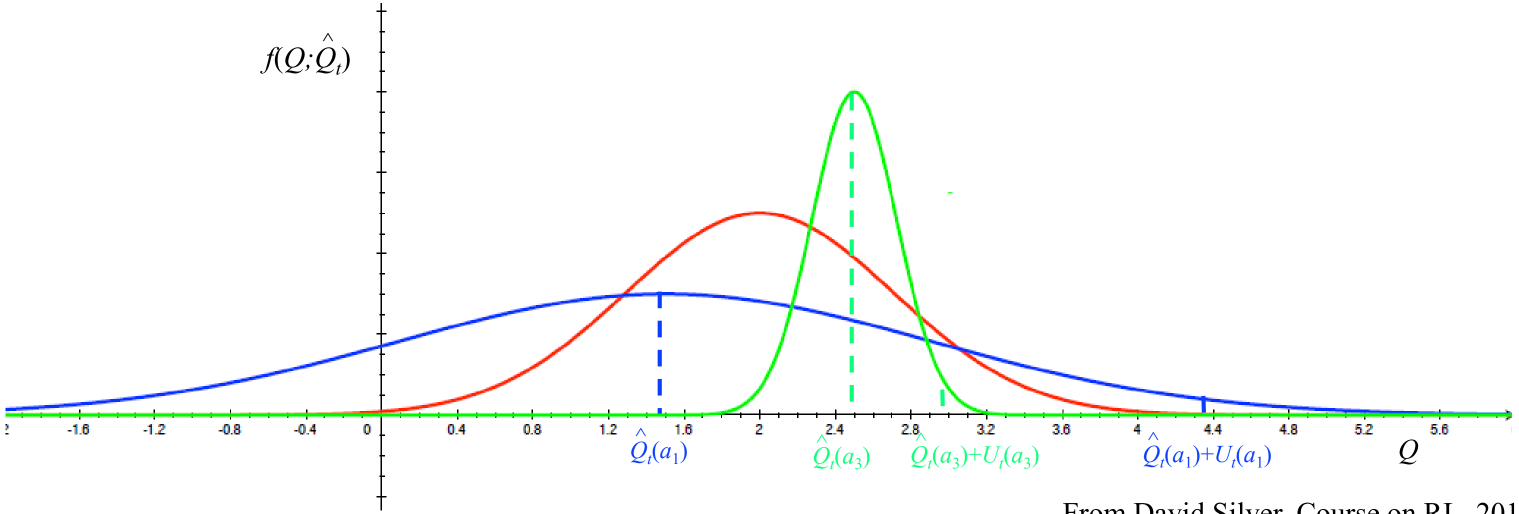
$$
\begin{aligned}
\hat{Q}_{t+1}(a) &= \hat{Q}_t(a) + \alpha\left(r_t - \hat{Q}_t(a)\right) \\
&= \alpha r_t + (1-\alpha)\hat{Q}_t(a) \\
&= \alpha r_t + (1-\alpha)\left(\alpha r_{t-1} + (1-\alpha)\hat{Q}_{t-1}(a)\right) \\
&= \alpha r_t + (1-\alpha)\alpha r_{t-1} + (1-\alpha)^2 \hat{Q}_{t-1}(a) \\
&= \alpha r_t + (1-\alpha)\alpha r_{t-1} + (1-\alpha)^2 \alpha r_{t-2} + \ldots + (1-\alpha)^{t-1}\alpha r_1 + (1-\alpha)^t \hat{Q}_1(a) \\
&= (1-\alpha)^t \hat{Q}_1(a) + \sum_{\tau=1}^{t} \alpha (1-\alpha)^{t-\tau} r_\tau
\end{aligned}
$$

The initial value contributes with a bias that fades to zero over time. Check that $\lim_{t\to\infty} \hat{Q}_{t+1}(a) = E\{r \mid a\}$

Utterly large optimistic initial values is a good strategy for exploration when $\varepsilon = 0$. **Why?**

Approach 2. Upper-confidence-bound action selection

The ε-greedy strategy treats all possible non-greedy actions equally during exploration. It would be wiser to select those having more potential for optimality, taking into account how close the estimates are to true ones.

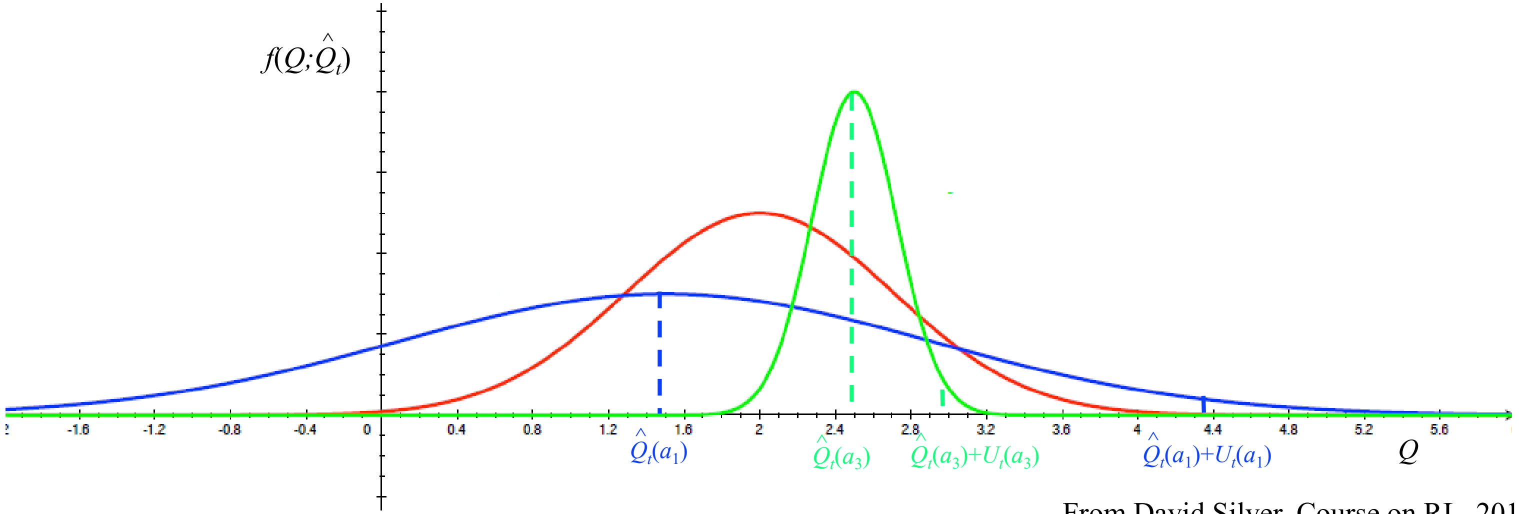


From David Silver, Course on RL, 2015

Which action should we pick?

Approach 2. Upper-confidence-bound action selection

Note that a_1 has the potential of being more rewarding than a_3...



From David Silver, Course on RL, 2015

The more uncertain we are about an action-value the more important is to explore it. It could be the best action! Optimism in the face of uncertainty.

Approach 2. Upper-confidence-bound action selection

Let us change the rule for picking an action:

- Estimate an upper confidence $U_t(a)$ for each action value, such that

$$Q(a) \le \hat{Q}_t(a) + U_t(a)$$

  with high probability. This will depend on the number of times $N(a)$ that a has been selected.

- Select actions maximizing the Upper Confidence Bound (UCB) algorithm:

$$a_t \doteq \arg\max_a \left[\hat{Q}_t(a) + U_t(a) \right]$$

  which naturally balances exploration and explotation.

Check Annex 1 for a derivation of the confidence bound $U_t(a)$.

Approach 2. Upper-confidence-bound action selection

$c > 0$ controls the level of exploration

$$a_t \doteq \arg\max_a \left[\hat{Q}_t(a) + c \underbrace{\sqrt{\frac{2 \ln t}{N_t(a)}}}_{\text{Related to the uncertainty in the estimate of } Q(a)} \right]$$

where no Gaussianity assumption is made on the estimated value-function.

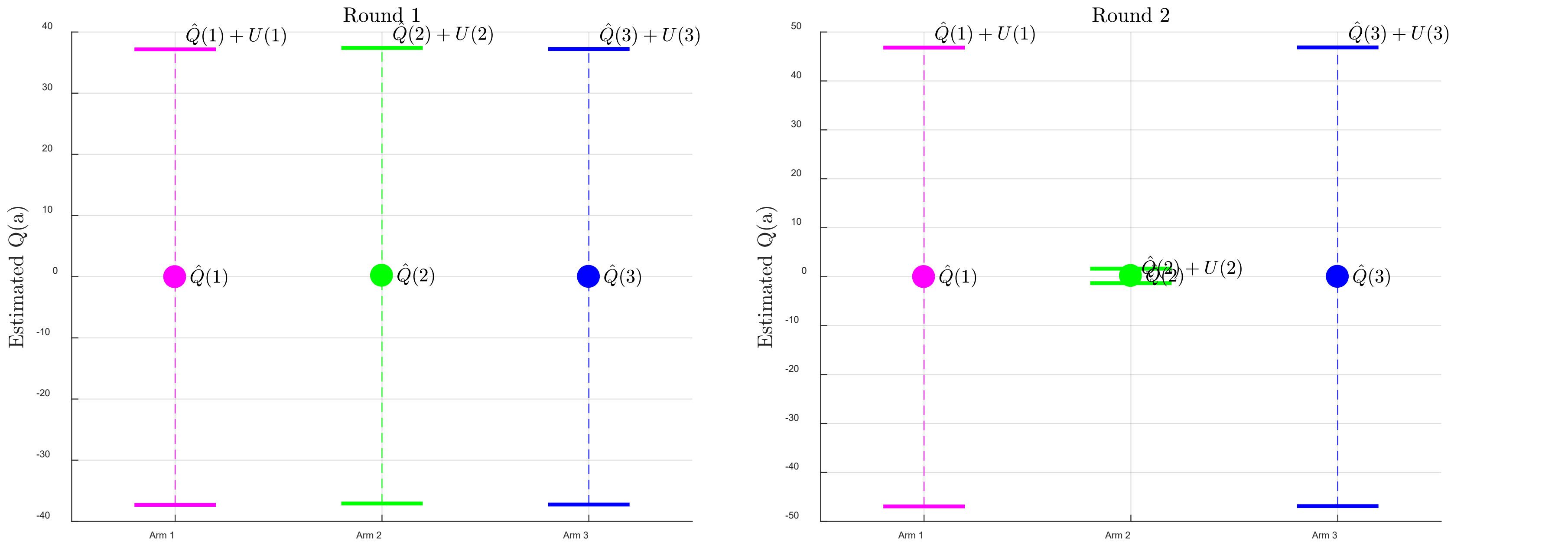
Note that uncertainty decreases with $N_t(a)$, the number of times that action a has been selected prior to time t.

UCB achieves logarithmic asymptotic regret: $L_t \leq \sum_{a \neq a^*} \min\left(\frac{10}{\Delta_a} \log t, t\Delta_a \right)$

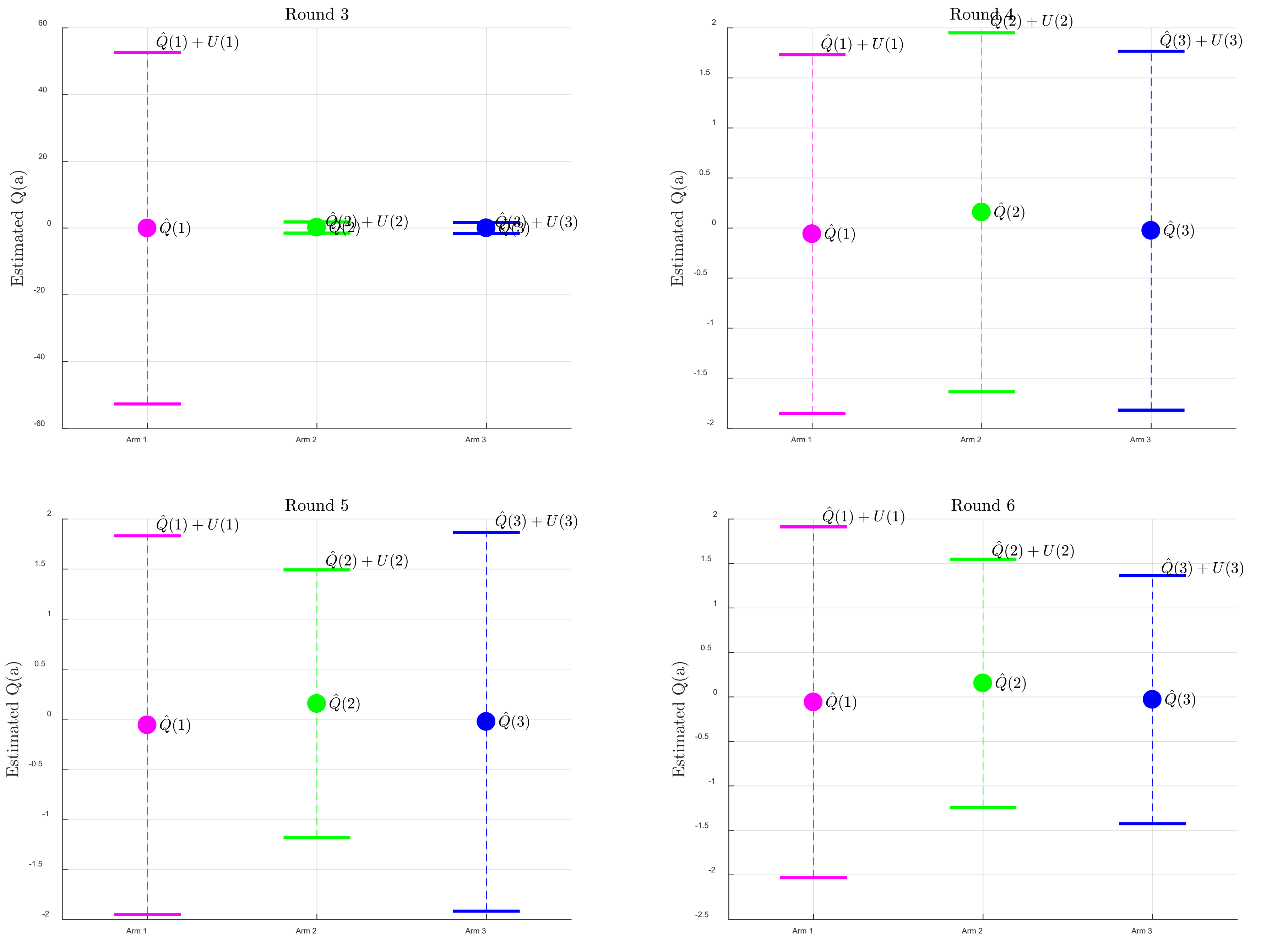
However, exploration mostly occurs at the beginning of the learning process. How to make it work in a non-stationary scenario?

Why does it work?

The evolution of the estimated $Q(a)$ and the uncertainty in a 3 arms case with Gaussian rewards.

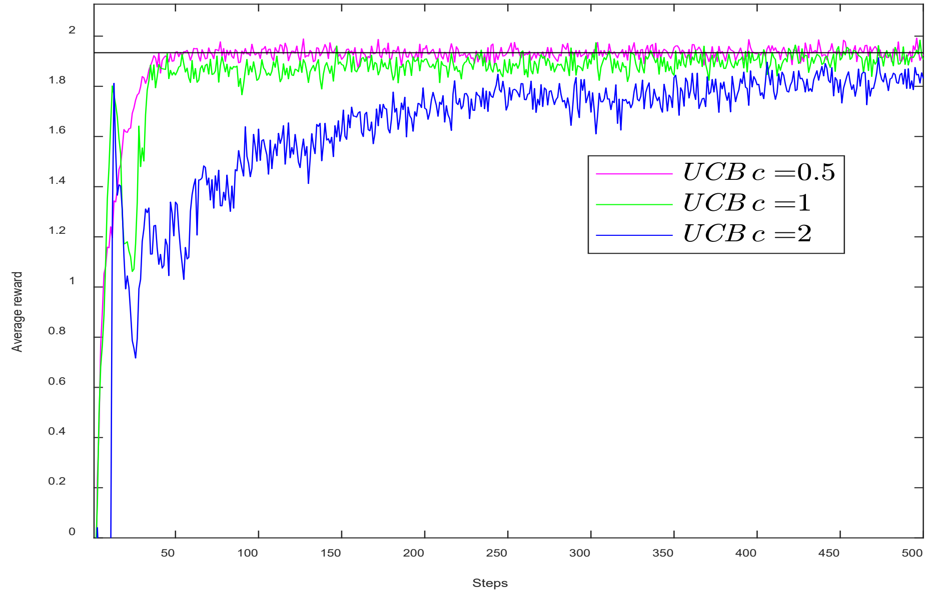


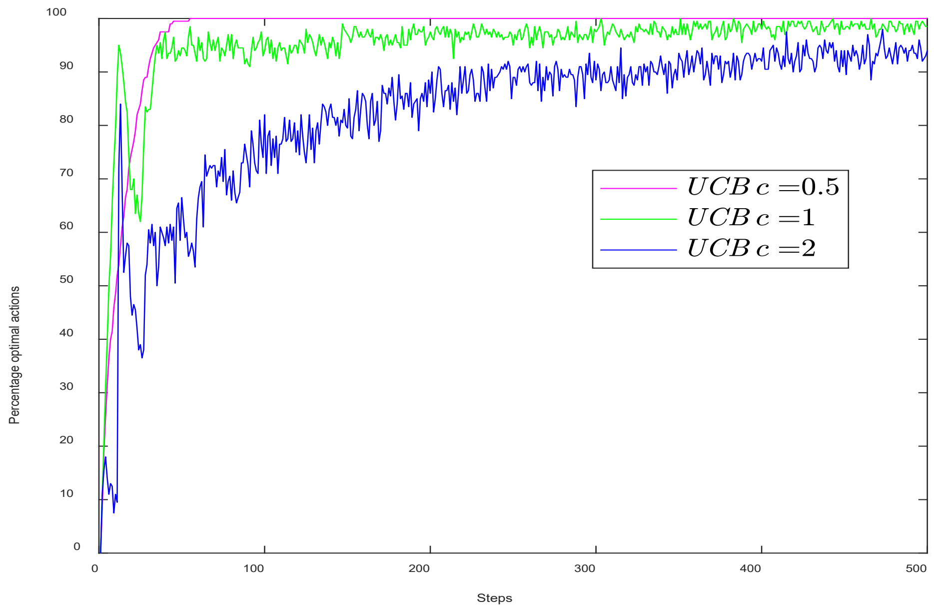
Guess which arm is selected upon each time step.



Round 3
Round 4
Round 5
Round 6
Estimated Q(a)
Arm 1
Arm 2
Arm 3
$\hat{Q}(1)+U(1)$
$\hat{Q}(2)+U(2)$
$\hat{Q}(3)+U(3)$
$\hat{Q}(1)$
$\hat{Q}(2)$
$\hat{Q}(3)$

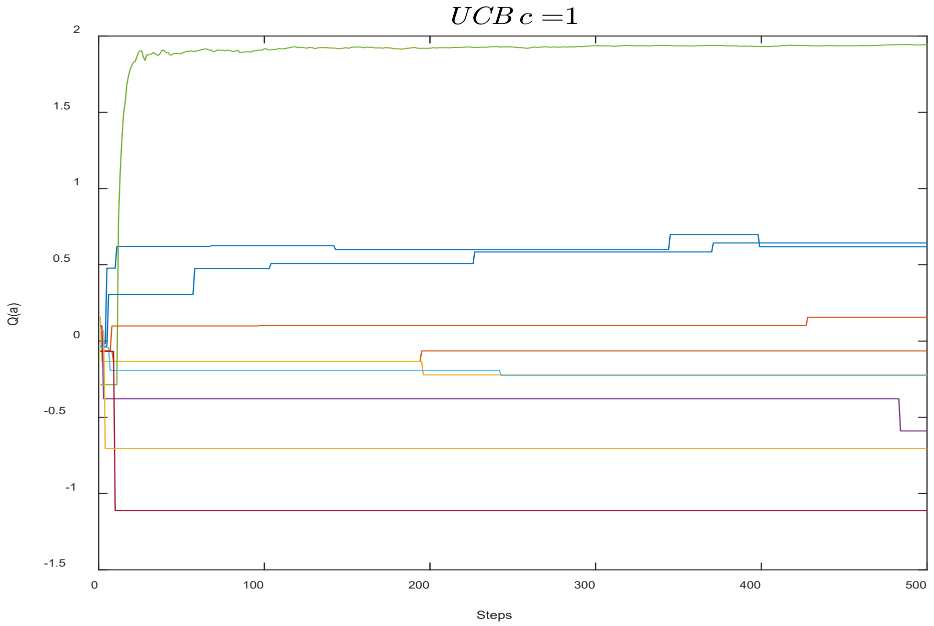
Example 2.1. UCB convergence averaged over 200 runs





Example 2.1. UCB convergence

Evolution of the estimated $Q(a)$ on a single run



True means $E\{r|a\}$: 0.8782 -0.1503 -0.4655 -0.8348 1.9340
-0.2624 -2.5056 0.7514 0.1667 -1.4844

Why only the largest ones are correctly estimated?

Approach 3. Soft-max strategy

An alternative way of addressing the drawbacks of ε-greedy approach is to use a soft-max strategy adopting a Boltzmann distribution, where actions are selected according to the following probability distribution:

$$\pi_t(a) = \frac{\exp\left(\frac{\hat{Q}_t(a)}{\gamma}\right)}{\sum_{y \in \mathcal{A}} \exp\left(\frac{\hat{Q}_t(y)}{\gamma}\right)}$$

The value of $\gamma > 0$ denotes the *temperature* of the system:

- $\gamma \rightarrow 0$ represents no exploration at all
- $\gamma \rightarrow \infty$ reflects a choice of action values with almost equal probability

Beware when computing exponentials!

… moving towards gradient bandits

Let us define $H_t(a)$ a preference for action a and introduce it in the expression above:

$$\pi_t(a) = \frac{\exp\left(H_t(a)\right)}{\sum_{y\in\mathcal{A}} \exp\left(H_t(y)\right)}$$

At each time step, action A_t is randomly selected according to $\pi_t(a)$ and a reward r_t is received. Then, if maximizing $E\{r_t\}$ using a gradient approach, the action preferences are updated as:

$$H_{t+1}(A_t) = H_t(A_t) + \alpha\left(r_t - \bar{R}_t\right)\left(1 - \pi_t(A_t)\right)$$

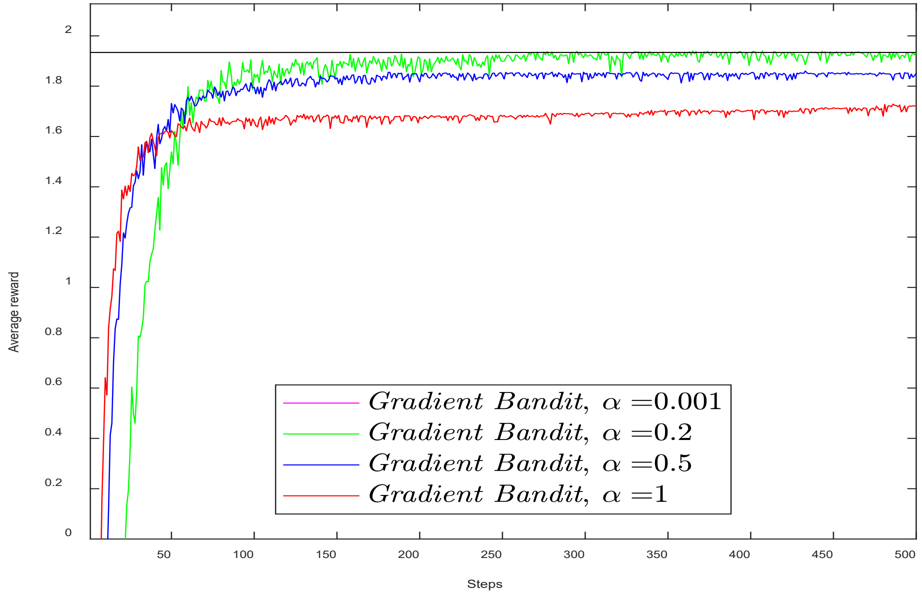
$$H_{t+1}(a) = H_t(a) - \alpha\left(r_t - \bar{R}_t\right)\pi_t(a) \qquad \text{for all } a \neq A_t$$

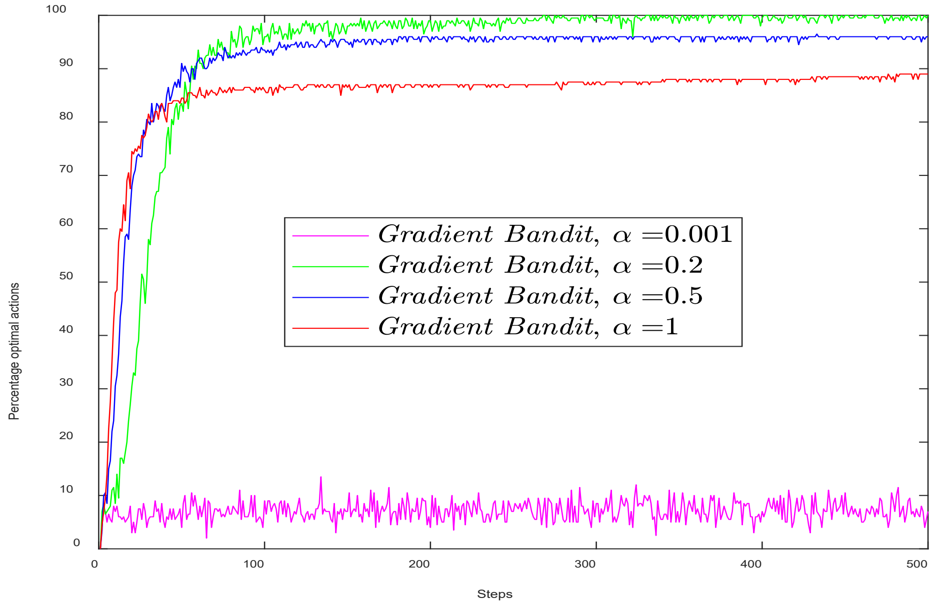
$\bar{R}_t$ is a baseline value

Interpretation: if the reward is higher than the baseline $\bar{R}_t$, the probability of taking A_t in the future is increased, and viceversa. The non-selected actions move in the opposite direction.

See Annex 2 for a proof.

Example 2.1. Gradient bandit convergence averaged over 200 runs





$$\overline{R}_t = \frac{1}{|\mathcal{A}|}\sum_{a\in\mathcal{A}} \hat{Q}(a)$$

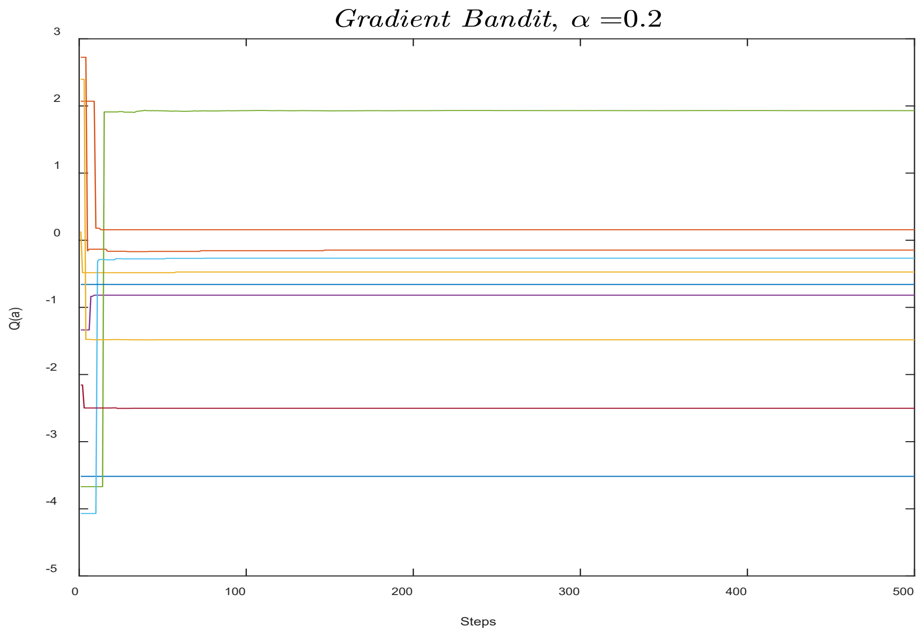
Convergence is faster if we take

$$\overline{R}_t = \frac{1}{|\mathcal{A}|}\sum_{a\in\mathcal{A}} E\{r\,|\,a\}$$

but requires prior knowledge

Example 2.1. Gradient bandit convergence

Evolution of the estimated $Q(a)$ on a single run



True means $E\{r|a\}$: 0.8782 -0.1503 -0.4655 -0.8348 1.9340
-0.2624 -2.5056 0.7514 0.1667 -1.4844

Approach 4. Probability matching

If we had some prior knowledge about $Q(a)$, how can we introduce it?
Assume we have observed a list of rewards associated to action a:

$$r_{t-1}^a, r_{t-2}^a, \ldots, r_1^a$$

The posterior of the average reward $Q(a)$ can be written using Bayes rule:

$$f\left(Q(a) \middle| r_t^a, r_{t-1}^a, r_{t-2}^a, \ldots, r_1^a\right) \propto f\left(r_t^a \middle| Q(a), r_{t-1}^a, r_{t-2}^a, \ldots, r_1^a\right) f\left(Q(a) \middle| r_{t-1}^a, r_{t-2}^a, \ldots, r_1^a\right)$$

rewards are independent over time

$$= f\left(r_t^a \middle| Q(a)\right) f\left(Q(a) \middle| r_{t-1}^a, r_{t-2}^a, \ldots, r_1^a\right)$$

If we have a model for the densities $f(.)$, we can sample K independent values and obtain an unbiased estimate as:

$$\hat{Q}(a) = \frac{1}{K} \sum_{k=1}^{K} Q_k(a)$$

Thomson sampling bandit algorithm

Initialize

$t = 0$

for $a = 1{:}m$

$\quad Q(a) = \text{rand}$

Loop forever

$$t \leftarrow t+1$$

$$A \leftarrow \arg\max_a \hat{Q}(a)$$

$$r_t^A \leftarrow bandit(A)$$

for $i = 1 : m$

$\quad$ for $k = 1 : K$

$$Q_k(i) \sim f\left(Q(i) \middle| r_t^i, r_{t-1}^i, \ldots, r_1^i\right)$$

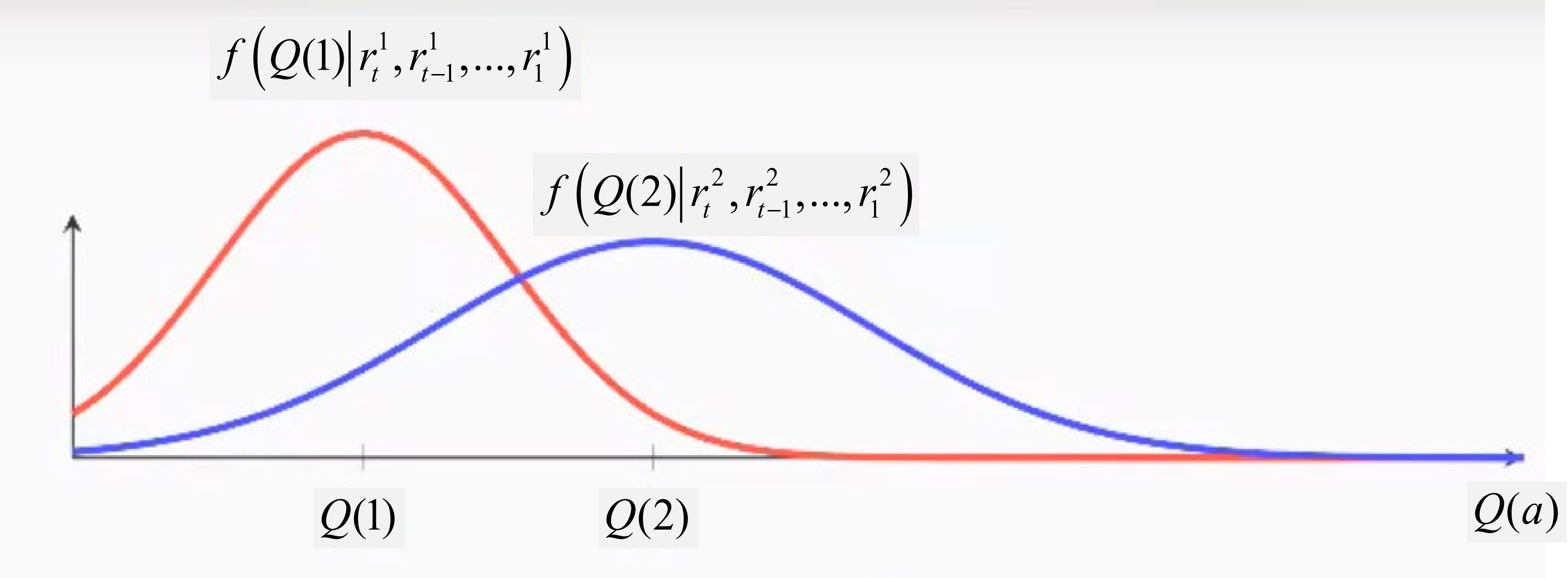
$$\hat{Q}(i) \leftarrow \frac{1}{K}\sum_{k=1}^{K} Q_k(i)$$

Generate K random samples of $Q(i)$ for each action

Rewards observed for action i up to time t

Smaller values of K entail more exploration. In general it works better than UCB (more aggressive in exploring)

Why does it work?



As arm #2 has been selected less times the variance of estimated $Q(2)$ is larger, so it is likely we draw samples with more extreme value → exploration (as in UCB)

Example 2.3. An m-armed bandit with **binary rewards**, with Beta priors

Let us see how to compute the posterior in a particular example. Assume the reward r associated to each action a has a Bernoulli density:

$$f_a\left(r\middle|\theta_a\right)=\theta_a^r\left(1-\theta_a\right)^{1-r} \qquad a=1,\ldots,m$$

It turns out that the mean of r^a (rewards for action a) is:

$$Q(a)=E\left\{r^a\right\}=1\times\theta_a+0\times\left(1-\theta_a\right)=\theta_a$$

Let us assume that the prior for the unknown $Q(a)$ is a Beta distribution:

$$f\left(\theta_a\middle|r_{t-1}^a,r_{t-2}^a,\ldots,r_1^a\right)=\text{Beta}\left(\theta_a;\alpha_a,\beta_a\right)\propto\theta_a^{\alpha_a-1}\left(1-\theta_a\right)^{\beta_a-1}$$

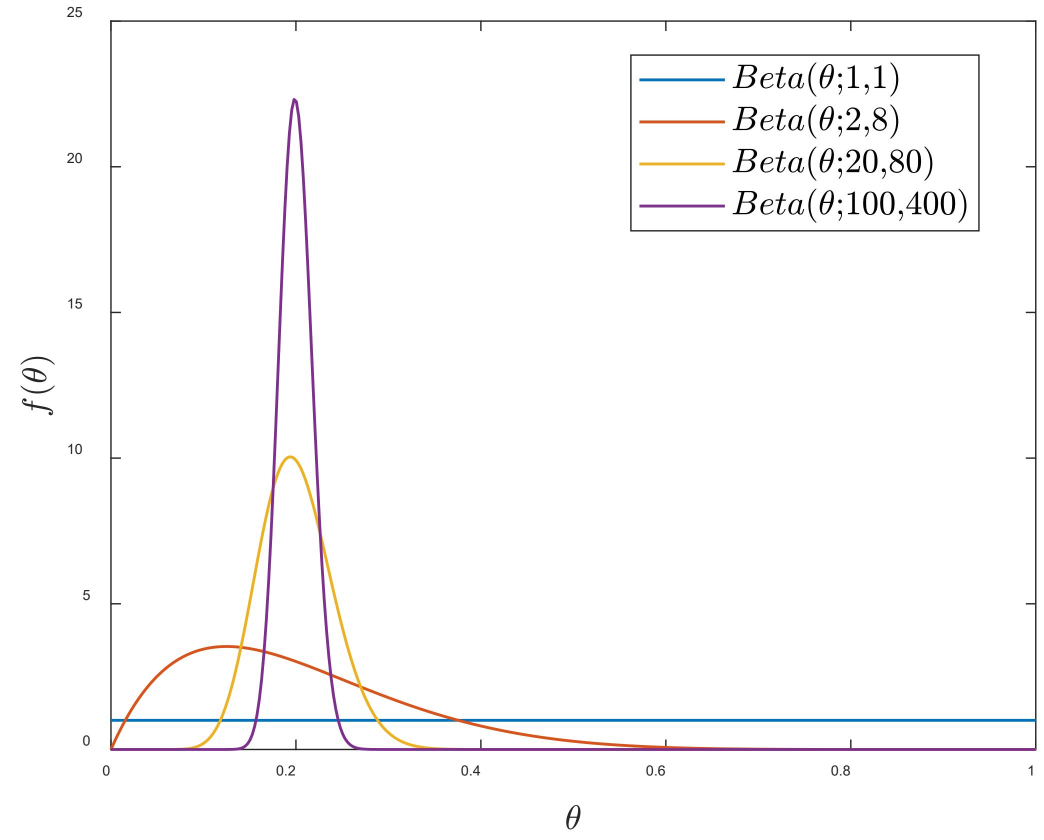
$$E\left\{\theta_a\right\}=\alpha_a/\left(\alpha_a+\beta_a\right)$$

where

$$\alpha_a-1:\ \text{\# of 1 observed in } r_{t-1}^a,r_{t-2}^a,\ldots,r_1^a$$

$$\beta_a-1:\ \text{\# of 0 observed in } r_{t-1}^a,r_{t-2}^a,\ldots,r_1^a$$

Note that we have different α and β for each possible action a.



25
20
15
10
5
0
0
0.2
0.4
0.6
0.8
1
$f(\theta)$
θ
$Beta(\theta;1,1)$
$Beta(\theta;2,8)$
$Beta(\theta;20,80)$
$Beta(\theta;100,400)$

Example 2.3. An *m*-armed bandit with **binary rewards**, with Beta priors

Let us combine the Bernoulli and the Beta densities to compute the posterior after arm *a* has been selected and we observed the reward:

- If reward is 1

$$f\left(\theta_a \middle| r_t^a = 1, r_{t-1}^a, \ldots, r_1^a\right) \propto f\left(r^a = 1 \middle| \theta_a\right) f\left(\theta_a \mid r_{t-1}^a, \ldots, r_1^a\right)$$

$$\propto \theta_a \theta_a^{\alpha_a - 1}\left(1-\theta_a\right)^{\beta_a - 1} = \mathrm{Beta}\left(\theta_a; \alpha_a + 1, \beta_a\right)$$

- If reward is 0

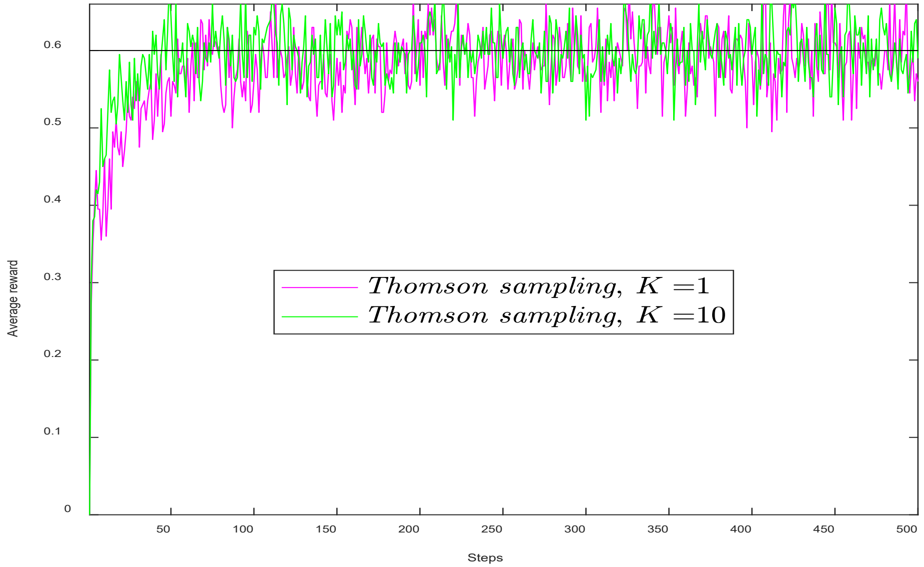
$$f\left(\theta_a \middle| r_t^a = 0, r_{t-1}^a, \ldots, r_1^a\right) \propto f\left(r^a = 0 \middle| \theta_a\right) f\left(\theta_a \mid r_{t-1}^a, \ldots, r_1^a\right)$$

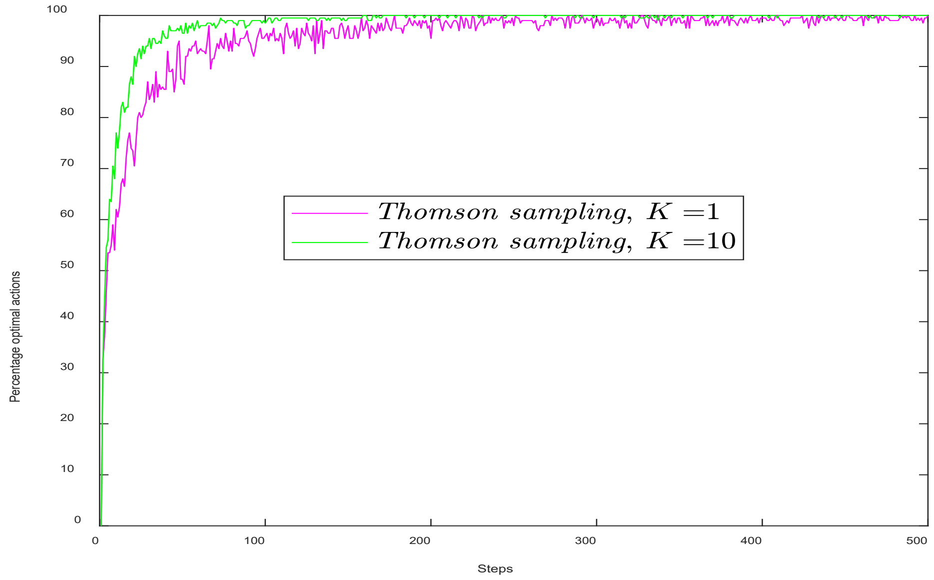
$$\propto \left(1-\theta_a\right) \theta_a^{\alpha_a - 1}\left(1-\theta_a\right)^{\beta_a - 1} = \mathrm{Beta}\left(\theta_a; \alpha_a, \beta_a + 1\right)$$

Then, we obtain Beta distributions for each arm. How to generate random samples of $Q(a)$ following a Beta distribution? In Python:

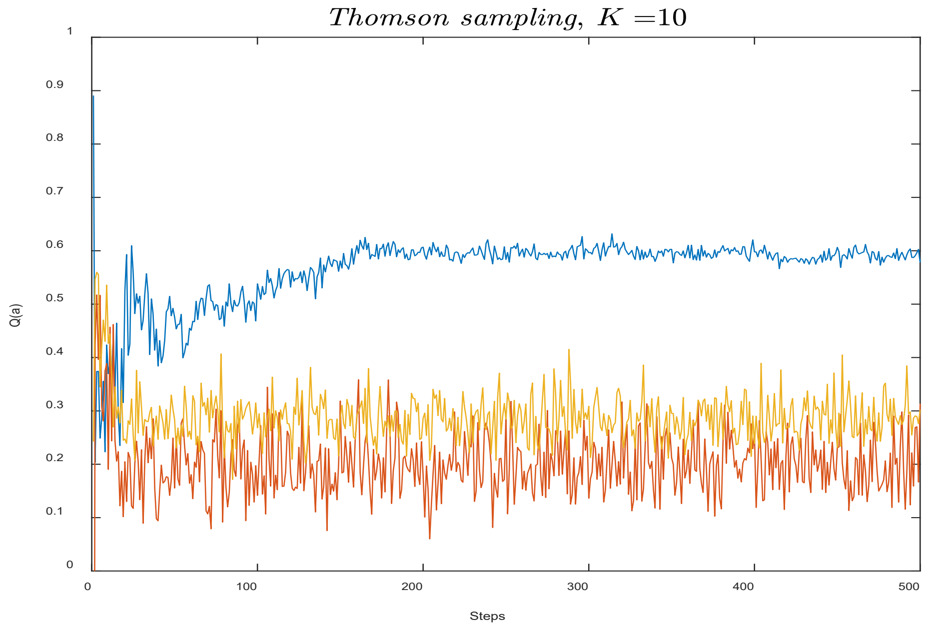
```
Q = numpy.random.beta(alpha,beta)
```

Example 2.3. Thomson sampling convergence averaged over 200 runs, for $\{0,1\}$ random reward on each of the 3 arms. $Q(a) = \theta_a = E\{r^a\}$ are $[0.6 \quad 0.2 \quad 0.1]$

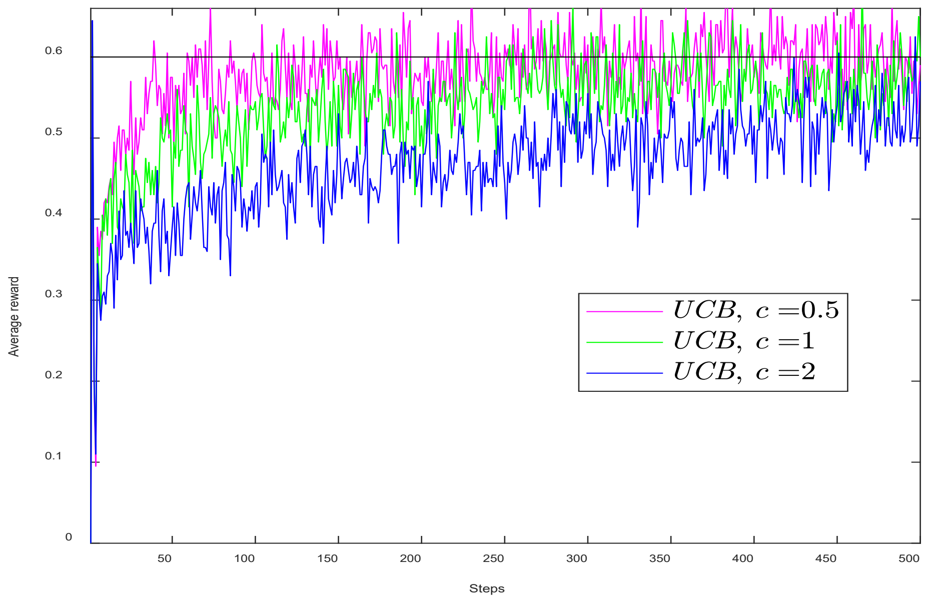




Example 2.3. Thomson sampling convergence averaged over 200 runs, for {0,1} random reward on each of the 3 arms. $Q(a) = \theta_a = E\{r^a\}$ are [0.6   0.2   0.1]

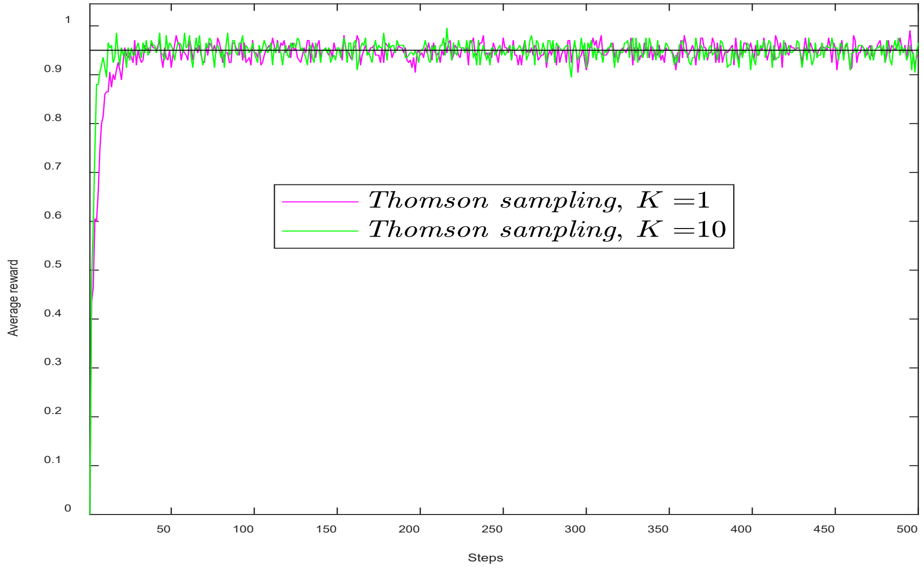


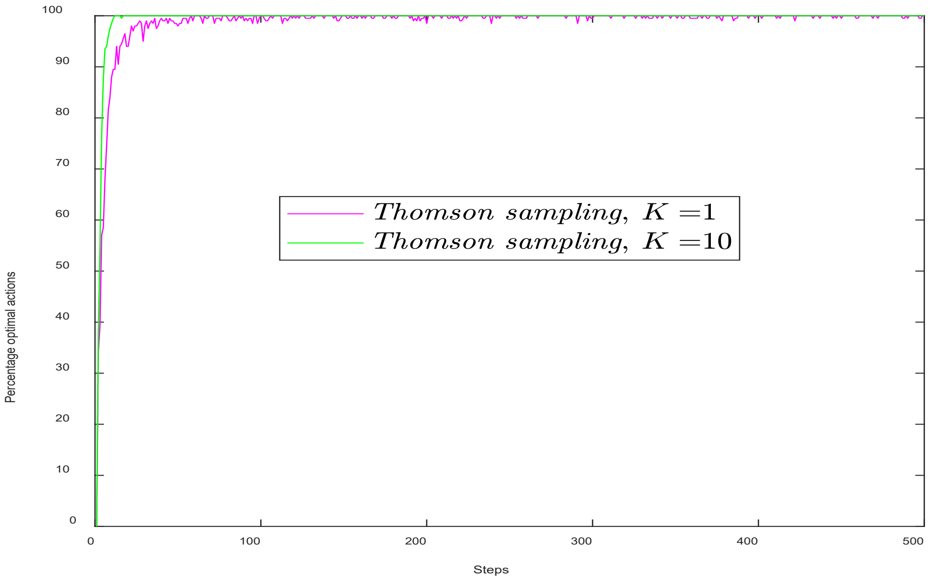
Example 2.3. UCB convergence averaged over 200 runs, for a $\{0,1\}$ random reward for each of the 3 arms. $Q(a) = \theta_a = E\{r^a\}$ are $[0.6 \quad 0.2 \quad 0.1]$



Note that the Bayesian approach provides faster convergence and better stability than UCB.

Example 2.3. Convergence gets better as means are more different: $Q(a) = \theta_a = E\{r^a\}$ are now $[0.95 \quad 0.2 \quad 0.1]$





No Exploration! → **Random Exploration** → **Smart Exploration**

Greedy algorithm

Є-Greedy algorithm

Upper confidence bounds (UCB)
Thompson sampling

Contextual bandits

Now the best possible action is tied to a context that is observed by the agent.

This is industrially exploited to generate personal user interfaces, personal advertising, personalised web contents,…

Contextual bandits

We need to learn a mapping from situations to actions, let us introduce states:

- A contextual bandit is defined by a tuple $\langle \mathcal{A}, \mathcal{S}, \mathcal{R} \rangle$
- $\mathcal{A}$ is a set of m actions (or “arms”)
- $\mathcal{S} = \Pr\{s\}$ is an unknown distribution over states (or “contexts”)
- At each step t
  - Environment generates state $s_t \sim \mathcal{S}$
  - Agent selects one action $a_t \in \mathcal{A}$
  - Environment generates a reward r_t
- The reward follows an unknown probability density $r \sim f(r|s, a)$, without memory
- States at each step are independent
- The objective is to find a policy that maximizes the cumulative reward

- The action-value function is now the mean reward for action and state:

$$Q(s,a) = E\{r \mid S = s, A = a\}$$

- Several approaches are possible:
  - Define P states using some kind of clustering, and solve P parallel m-armed bandits
  - Parametrize function $Q(s, a)$, e.g. linear function, neural network [1]

[1] M. Collier, H. Urdiales Llorens, "Deep Contextual Multi-armed Bandits", https://arxiv.org/abs/1807.09809

Linear regression for contextual bandits

- Model the action-value function with a linear approximator:

$$Q(s,a) = \boldsymbol{\phi}(s,a)^T \boldsymbol{\theta}$$

Context vector (user demographics, history of web browsing, possible actions,… and any other relevant stuff!)

- We gather a number of observations and rewards:

E.g., number of clicks in a web page

$$r_\tau = \boldsymbol{\phi}(s_\tau, a_\tau)^T \boldsymbol{\theta} + w_\tau \qquad \tau = 1,\dots,t$$

being w_τ a zero mean disturbance.

- Then, we can solve the parameters of the model by using least squares:

$$\mathbf{A}_t = \begin{bmatrix} \boldsymbol{\phi}(s_1,a_1)^T \\ \vdots \\ \boldsymbol{\phi}(s_t,a_t)^T \end{bmatrix} \qquad \mathbf{r}_t = \begin{bmatrix} r_1 \\ \vdots \\ r_t \end{bmatrix} \qquad \boldsymbol{\theta}_t = \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \mathbf{A}_t^T \mathbf{r}_t$$

for which it is possible to derive recursive-in-time low-cost solutions.

- The least squares approach allows computing the variance of the action-value function, i.e. the uncertainty in its estimation. Therefore, we can define a UCB for the linear model:

$$a_{t+1} \doteq \arg\max_a \left[\hat{Q}_{t+1}(s,a) + c \underbrace{\sqrt{2 \ln t \, \boldsymbol{\phi}(s_{t+1},a)^T \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \boldsymbol{\phi}(s_{t+1},a)}} \right]$$

$c > 0$ controls the level of exploration

Confidence bound $U_t(a)$: it is related to the uncertainty in the estimate of $Q(s, a)$

Some further degree of exploration can be obtained using parameter c.

- The cost of matrix inversion can be reduced by noting that

$$\mathbf{A}_t = \begin{bmatrix} \mathbf{A}_{t-1} \\ \boldsymbol{\phi}(s_t, a_t)^T \end{bmatrix}$$

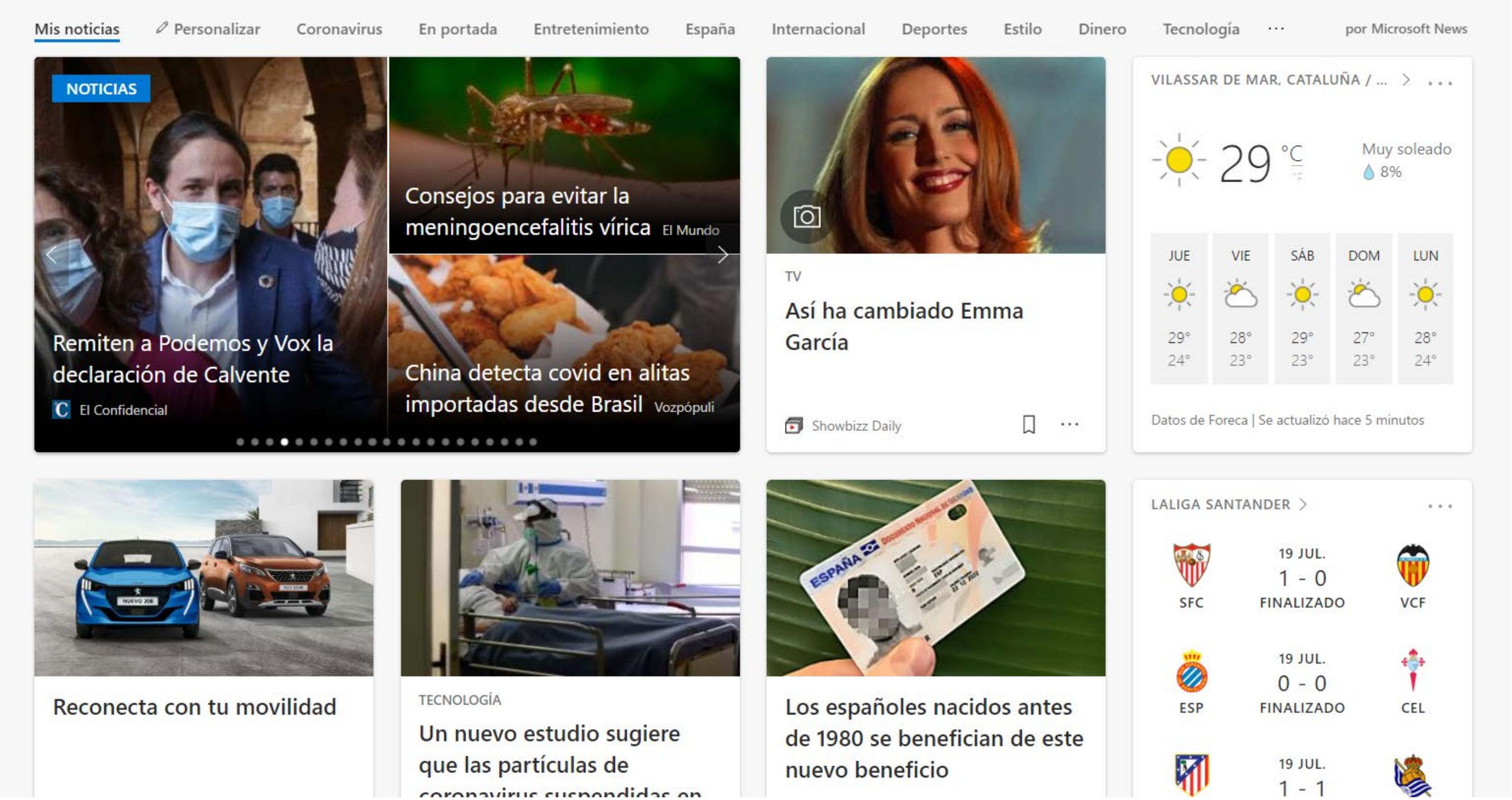
and using the recursion provided by the Sherman-Morrison formula.

- Diagonal loading $\left(\mathbf{A}_t^T \mathbf{A}_t + \lambda \mathbf{I}\right)^{-1}$ is needed to guarantee the existence of the inverse for small t.

Check Annex 3 for a derivation of the confidence bound $U_t(a)$.

Practical use case

MSN news: 26% improvement in click through rate after adopting the contextual bandits approach.



https://www.microsoft.com/en-us/research/blog/real-world-interactive-learning-cusp-enabling-new-class-applications

The adversarial m-armed bandit problem

Let us formalize the concept…

An adversarial multi-armed bandit is defined by $\langle \mathcal{A}, \mathcal{R} \rangle$

- $\mathcal{A}$ is a set of $m=|\mathcal{A}|$ actions (or "arms")
- At each step t

  - an oblivious adversary defines rewards $\mathcal{R} = \{r_{1,t}, \ldots, r_{m,t}\} \in [0,1]^m$

  - agent selects one action $a_t \in \mathcal{A}$

  - agent receives a reward $r_{a_t,t}$ from the set $\mathcal{R}$

- The objective is to take appropriate decisions so as to minimize the pseudo-regret

$$\bar{L}_t = \max_{i=1,\ldots,m} E\left\{\sum_{\tau=1}^{t} r_{i,\tau} - \sum_{\tau=1}^{t} r_{a_\tau,\tau}\right\}$$

Exp3 algorithm

Initialize

$t=0,\ \text{for } i=1:m \quad p_1(i)=1/m \quad S_0(i)=0 \quad \eta\in(0,1]$

Loop forever

$t \leftarrow t+1$

exploration & exploitation

$A \leftarrow$ random sample from distribution $p_t(1),\dots,p_t(m)$

$r_t^A \leftarrow bandit(A)$

for $i=1:m$

Kronecker delta

$$\hat{s}_t = \frac{r_t^A}{p_t(A)}\delta_{i-A}$$

unbiased estimate of reward

$$S_t(i) \leftarrow S_{t-1}(i)+\hat{s}_t$$

$$p_{t+1}(i) = (1-\eta)\frac{\exp\left(\eta_t S_t(i)\right)}{\sum_{k=1}^{m}\exp\left(\eta_t S_t(k)\right)}+\frac{\eta}{m}$$

If the algorithm is run with $\eta_t=\sqrt{\dfrac{\ln m}{tm}}$, the pseudo-regret is bounded: $\overline{L}_t \le 2\sqrt{tm\ln m}$

http://cseweb.ucsd.edu/~yfreund/papers/bandits.pdf

Quiz #1

Annex 1. Proof for UCB

Let us apply a fundamental statistical result:

Hoeffding's inequality

Let $X_1, \ldots, X_t$ be iid bounded random variables and let the sample mean be:

$$\bar{X}_t = \frac{1}{t} \sum_{\tau=1}^{t} X_\tau$$

Then,

$$\Pr\left\{ E\{X\} > \bar{X}_t + u \right\} \le \exp\left(-2tu^2\right)$$

Let us apply Hoeffding's inequality to rewards of the bandit on each selecting action:

$$\Pr\left\{ Q(a) > \hat{Q}_t(a) + U_t(a) \right\} \le \exp\left(-2N_t(a)U_t(a)^2\right)$$

Define a confidence bound on the value of $Q(a)$ using this inequality, and solve for $U_t(a)$:

$$\exp\left(-2N_t(a)U_t(a)^2\right) = p$$

$$U_t(a) = \sqrt{\frac{-\ln p}{2N_t(a)}}$$

We can reduce p as we observe more rewards, so that we can be more confident as time passes, e.g. $p = t^{-4}$:

$$U_t(a) = \sqrt{\frac{2\ln t}{N_t(a)}}$$

Annex 2. Gradient bandits

Gradient ascend principle – the preference $H(a)$ is incremented proportionally to the increment in performance:

$$H_{t+1}(a) = H_t(a) + \alpha \frac{\partial E\{r_t\}}{\partial H_t(a)}$$

where the performance measure is $E\{r_t\} = \sum_{x \in \mathcal{A}} \pi_t(x) Q(x)$

Let us compute the gradient:

$$\frac{\partial E\{r_t\}}{\partial H_t(a)} = \frac{\partial}{\partial H_t(a)} \left[\sum_{x \in \mathcal{A}} \pi_t(x) Q(x) \right] = \sum_{x \in \mathcal{A}} Q(x) \frac{\partial \pi_t(x)}{\partial H_t(a)} = \sum_{x \in \mathcal{A}} \left(Q(x) - B_t \right) \frac{\partial \pi_t(x)}{\partial H_t(a)}$$

As the gradient sums to zero over all the actions, so it is licit to introduce a baseline B_t that does not depend on x. We will choose arbitrarily $B_t = \bar{R}_t$.

Now we have to manipulate the expression…

$$
\begin{aligned}
\frac{\partial E\{r_t\}}{\partial H_t(a)} &= \sum_{x\in\mathcal{A}} \pi_t(x)\big(Q(x)-B_t\big)\frac{\partial \pi_t(x)}{\partial H_t(a)}\frac{1}{\pi_t(x)} \\
&= E\left\{\big(Q(A_t)-B_t\big)\frac{\partial \pi_t(A_t)}{\partial H_t(a)}\frac{1}{\pi_t(A_t)}\right\} \\
&= E\left\{\big(r_t-\bar{R}_t\big)\frac{\partial \pi_t(A_t)}{\partial H_t(a)}\frac{1}{\pi_t(A_t)}\right\} && \text{since } E\{r_t \mid A_t\} = Q(A_t) \\
&= E\left\{\big(r_t-\bar{R}_t\big)\pi_t(A_t)\big(\mathbb{1}_{a=x}-\pi_t(a)\big)\frac{1}{\pi_t(A_t)}\right\} && \text{to be proven in next slide} \\
&= E\left\{\big(r_t-\bar{R}_t\big)\big(\mathbb{1}_{a=x}-\pi_t(a)\big)\right\}
\end{aligned}
$$

Now substitute the expectation by the argument to obtain the stochastic-like algorithm proposed:

$$
H_{t+1}(A_t) = H_t(A_t) + \alpha\big(r_t-\bar{R}_t\big)\big(\mathbb{1}_{a=A_t}-\pi_t(a)\big) \qquad \text{for all } a
$$

Let us finally prove the expression for the gradient:

$$\frac{\partial \pi_t(x)}{\partial H_t(a)} = \frac{\partial}{\partial H_t(a)}\left[\frac{\exp\left(H_t(x)\right)}{\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right)}\right]$$

$$= \frac{\frac{\partial \exp\left(H_t(x)\right)}{\partial H_t(a)}\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right) - \exp\left(H_t(x)\right)\frac{\partial \sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right)}{\partial H_t(a)}}{\left(\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right)\right)^2}$$

$$= \frac{\mathbb{1}_{a=x}\exp\left(H_t(x)\right)\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right) - \exp\left(H_t(x)\right)\exp\left(H_t(a)\right)}{\left(\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right)\right)^2}$$

$$= \frac{\mathbb{1}_{a=x}\exp\left(H_t(x)\right)}{\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right)} - \frac{\exp\left(H_t(x)\right)\exp\left(H_t(a)\right)}{\left(\sum_{y\in\mathcal{A}}\exp\left(H_t(y)\right)\right)^2}$$

$$= \mathbb{1}_{a=x}\pi_t(x) - \pi_t(x)\pi_t(a)$$

$$= \pi_t(x)\left(\mathbb{1}_{a=x} - \pi_t(a)\right)$$

Annex 3. UCB for linear contextual bandits

Again, we want to compute the bound on the estimation of $Q(s,a)$:

$$\Pr\left\{Q(s,a) > \hat{Q}_{t+1}(s,a) + U_{t+1}\right\} \leq p$$

assuming now that the mean reward fits a linear model: $Q(s,a) = \boldsymbol{\phi}(s,a)^T \boldsymbol{\theta}$

while the observed rewards are $r_\tau = \boldsymbol{\phi}(s_\tau, a_\tau)^T \boldsymbol{\theta} + w_\tau$

Using the t observed rewards we can determine $\boldsymbol{\theta}$ using LS:

$$\boldsymbol{\theta}_t = \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \mathbf{A}_t^T \mathbf{r}_t$$

Therefore, the estimated mean reward at time $t+1$ is given by:

$$\hat{Q}_{t+1}(s,a) = \boldsymbol{\phi}(s_{t+1}, a)^T \boldsymbol{\theta}_t$$

Let us now statistically characterize $\hat{Q}_t(s,a)$ in terms of bias and variance.

The estimate is unbiased according to our model:

$$E\left\{\hat{Q}_{t+1}(s,a)\right\} = E\left\{\boldsymbol{\phi}(s_{t+1},a)^T \boldsymbol{\theta}_t\right\} = \boldsymbol{\phi}(s_{t+1},a)^T E\left\{\boldsymbol{\theta}_t\right\} = \boldsymbol{\phi}(s_{t+1},a)^T \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \mathbf{A}_t^T E\left\{\mathbf{r}_t\right\}$$

$$= \boldsymbol{\phi}(s_{t+1},a)^T \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \mathbf{A}_t^T \mathbf{A}_t \boldsymbol{\theta} = \boldsymbol{\phi}(s_{t+1},a)^T \boldsymbol{\theta}$$

The variance depends on the covariance matrix of the LS estimate:

$$\operatorname{var}\left\{\hat{Q}_{t+1}(s,a)\right\} = E\left\{\left|\hat{Q}_{t+1}(s,a) - \boldsymbol{\phi}(s_{t+1},a)^T \boldsymbol{\theta}\right|^2\right\} = E\left\{\left|\boldsymbol{\phi}(s_{t+1},a)^T \boldsymbol{\theta}_t - \boldsymbol{\phi}(s_{t+1},a)^T \boldsymbol{\theta}\right|^2\right\}$$

$$= \boldsymbol{\phi}(s_{t+1},a)^T E\left\{\left|\boldsymbol{\theta}_t - \boldsymbol{\theta}\right|^2\right\} \boldsymbol{\phi}(s_{t+1},a) = \boldsymbol{\phi}(s_{t+1},a)^T \operatorname{cov}\left\{\boldsymbol{\theta}_t\right\} \boldsymbol{\phi}(s_{t+1},a_t)$$

$$\operatorname{cov}\left\{\boldsymbol{\theta}_t\right\} = \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \mathbf{A}_t^T E\left\{\mathbf{w}\mathbf{w}^T\right\} \mathbf{A}_t \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} = \sigma_w^2 \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1}$$

since rewards are independent among observations.

Now let us assume that the estimated mean reward is Gaussian:

$$\hat{Q}_{t+1}(s,a) \sim \mathcal{N}\left(\boldsymbol{\phi}(s_{t+1},a)^T \boldsymbol{\theta}; \sigma_w^2 \boldsymbol{\phi}(s_{t+1},a)^T \left(\mathbf{A}_t^T \mathbf{A}_t\right)^{-1} \boldsymbol{\phi}(s_{t+1},a)\right)$$

We can now rewrite the bound:

$$\Pr\left\{Q(s,a)-\hat{Q}_{t+1}(s,a)>U_{t+1}\right\}=\int_{U_{t+1}}^{\infty}\frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{x^2}{2\sigma^2}\right)dx$$

where $\sigma^2=\sigma_w^2\boldsymbol{\phi}(s_{t+1},a)^T\left(\mathbf{A}_t^T\mathbf{A}_t\right)^{-1}\boldsymbol{\phi}(s_{t+1},a)$.

Changing variables we can use the Gaussian integral function $\mathcal{Q}$ and upper bound it by the exponential (Chernoff bound):

$$\Pr\left\{Q(s,a)-\hat{Q}_{t+1}(s,a)>U_{t+1}\right\}=\mathcal{Q}\left(U_{t+1}\left(\sigma_w^2\boldsymbol{\phi}(s_{t+1},a)^T\left(\mathbf{A}_t^T\mathbf{A}_t\right)^{-1}\boldsymbol{\phi}(s_{t+1},a)\right)^{-1/2}\right)$$

$$\leq\exp\left(-U_{t+1}^2\left(\sigma_w^2\boldsymbol{\phi}(s_{t+1},a)^T\left(\mathbf{A}_t^T\mathbf{A}_t\right)^{-1}\boldsymbol{\phi}(s_{t+1},a)\right)^{-1}\right)\leq p$$

From here: $U_{t+1}=\left(-\ln p\,\sigma_w^2\,\boldsymbol{\phi}(s_{t+1},a)^T\left(\mathbf{A}_t^T\mathbf{A}_t\right)^{-1}\boldsymbol{\phi}(s_{t+1},a)\right)^{1/2}$

We can reduce the value of p as we get more rewards, to signal that we are more confident on the value of $Q(s,a)$ as time passes, e.g. $p=t^{-4}$.

Some further references…

- Practical applications for multi-armed bandits: https://arxiv.org/pdf/1904.10040.pdf
- https://banditalgs.com/
- Exploration and exploitation (by David Silver): https://www.youtube.com/watch?v=sGuiWX07sKw&list=PLqYmG7hTraZBiG_XpjnPrSNw-1XQaM_gB&index=9
- Contextual bandits (by Pascal Poupart): https://www.youtube.com/watch?v=jlcbEZTgisQ
- Bayesian multi-armed bandits (by Nando de Freitas): https://www.youtube.com/watch?v=vz3D36VXefI
- Regret analysis of stochastic and non-stochastic multi-armed bandits https://arxiv.org/pdf/1204.5721v2.pdf