

Deep Reinforcement Learning and Control

Multi-armed bandits

Spring 2020, CMU 10-403

Katerina Fragkiadaki



This lecture - Motivation

Learning to act in a non-sequential setup:

- Each action results in an **immediate** reward.
- We want to choose actions that maximize our **expected** immediate reward.
- Q: why expected?
- A: because rewards of actions are not deterministic, e.g., displaying an ad can generate different click rates in different days.

This lecture - Motivation

Learning to act in a non-sequential setup:

- Each action returns an immediate reward.
- We want to choose actions that maximize our cumulative expected reward or final reward.

Examples: choosing which advertisement to display to a user

This lecture - Motivation

Learning to act in a non-sequential setup:

- Each action returns an immediate reward.
- We want to choose actions that maximize our cumulative expected reward or final reward.

Examples: choosing which advertisement to display to a user

Actions: the advertisements to be displayed, Rewards: the user click-through rate

This lecture - Motivation

Learning to act in a non-sequential setup:

- Each action returns an immediate reward.
- We want to choose actions that maximize our expected (immediate) reward.

Examples: drug design

Actions: the chemical consistencies to mix (**continuous action space!** NEXT lecture), Rewards: drug effectiveness as measured in mice

Exploration vs Exploitation Dilemma

- Online decision-making involves a fundamental choice:
 - **Exploitation:** Make the best decision given current information
 - **Exploration:** Gather more information
- The best long-term strategy may involve **short-term sacrifices**
- Gather enough information to make the best overall decisions

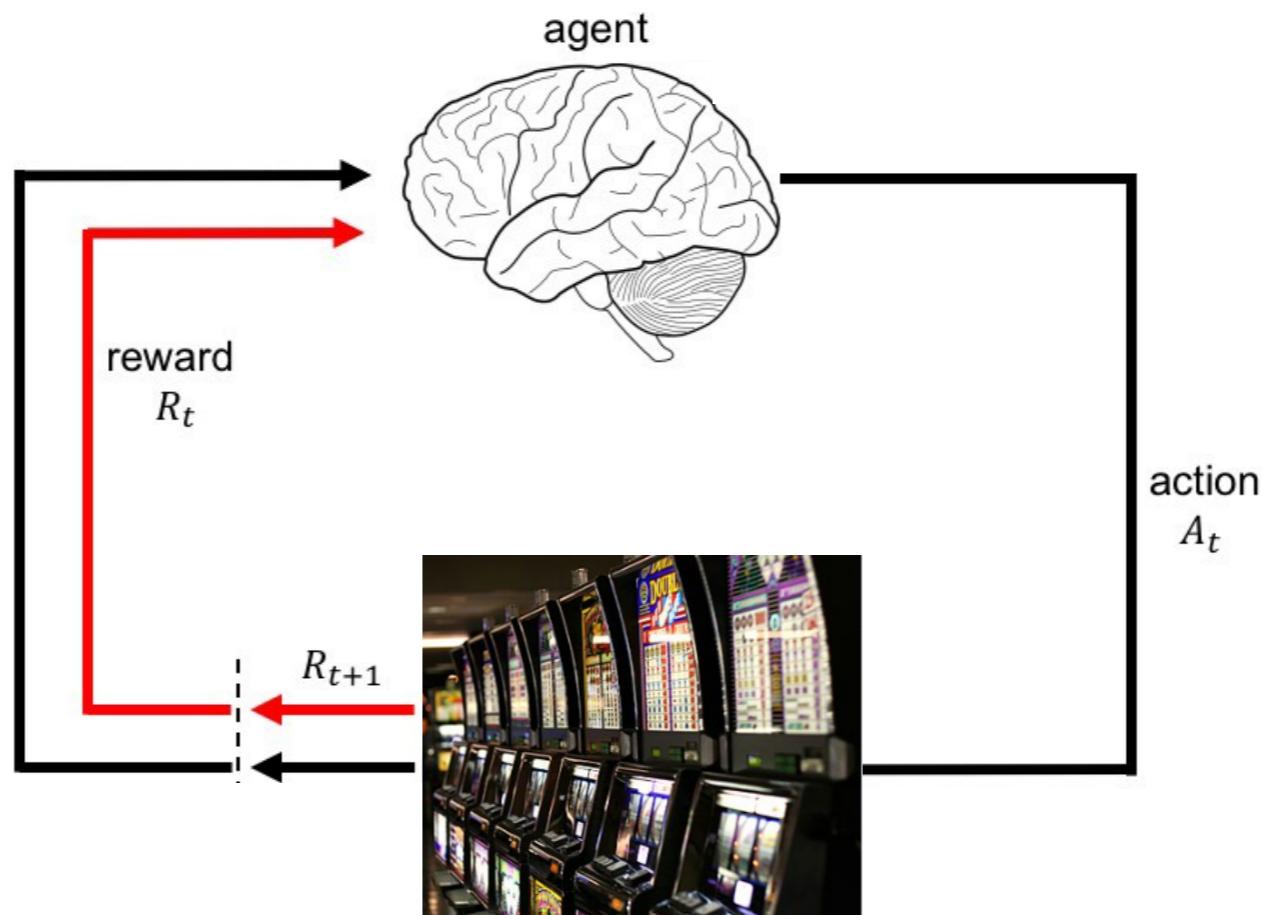
Exploration vs Exploitation Dilemma

- Online decision-making involves a fundamental choice:
 - **Exploitation:** Make the best decision given current information
 - **Exploration:** Gather more information
- The best long-term strategy may involve **short-term sacrifices**
- Gather enough information to make the best overall decisions
- The exploration/exploitation dilemma is not a problem encountered in computational RL or deep RL: It is a fundamental problem in decision making of any intelligent agent, e.g., a human being.

Exploration vs. Exploitation Dilemma

- Restaurant Selection
 - **Exploitation:** Go to your favorite restaurant
 - **Exploration:** Try a new restaurant
- Oil Drilling
 - **Exploitation:** Drill at the best known location
 - **Exploration:** Drill at a new location
- Game Playing
 - **Exploitation:** Play the move you believe is best
 - **Exploration:** Play an experimental move

Multi-Armed Bandits



$$A_t, R_{t+1}, A_{t+1}, R_{t+2}, A_{t+2}, A_{t+3}, R_{t+3}, \dots$$

The state does not change! (a.k.a. stateless)

Multi-Armed Bandits

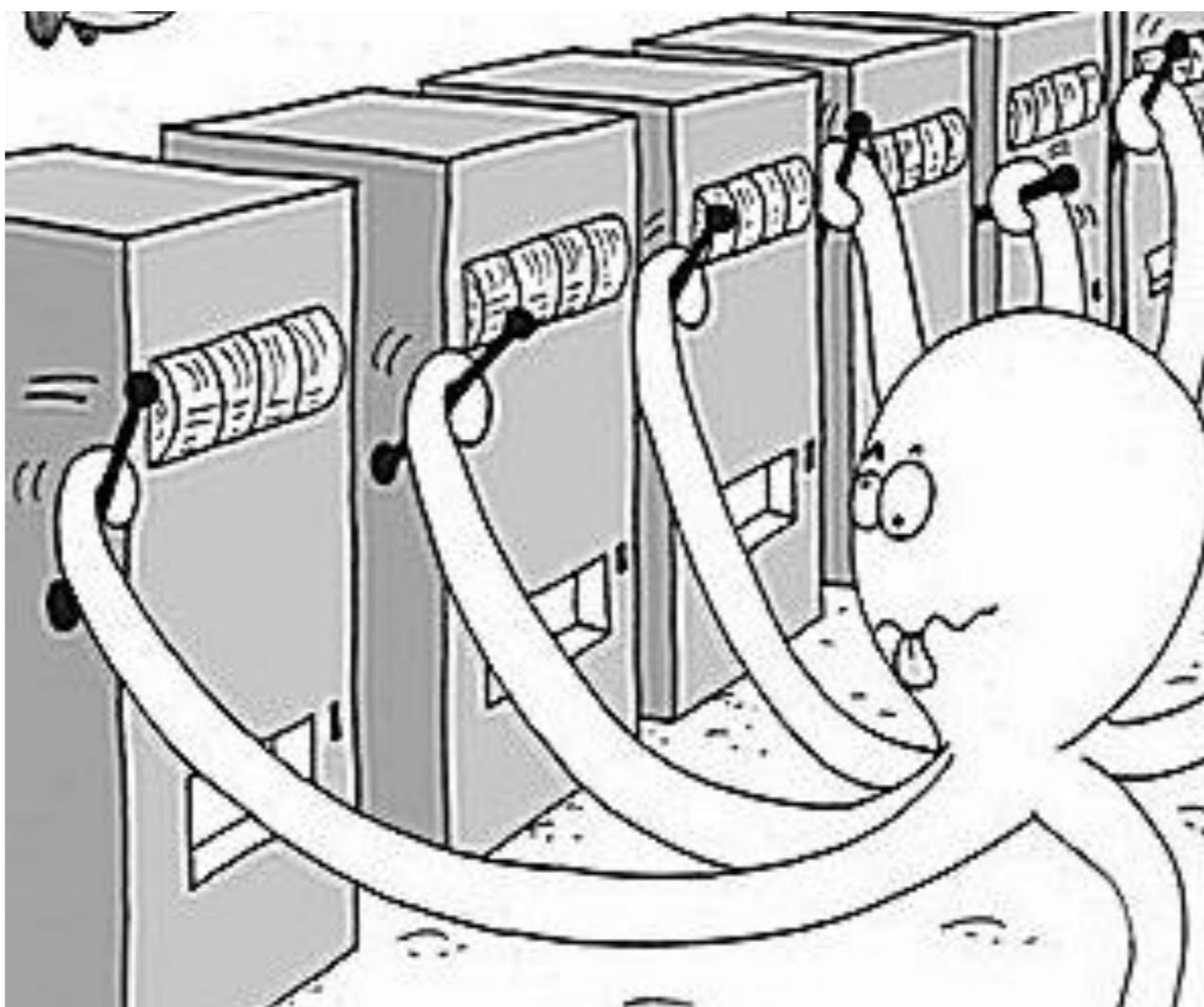
One-armed bandit= Slot machine (English slang)



source: infoslotmachine.com

Multi-Armed Bandits

Multi-Armed bandit = Multiple Slot Machine



source: Microsoft Research

Multi-Armed Bandit Problem

At each timestep t the agent chooses one of the K arms and plays it.

The i th arm produces reward $r_{i,t}$ when played at timestep t .

The rewards $r_{i,t}$ are drawn from a probability distribution \mathcal{P}_i with mean μ_i .

need to be learned!!

The agent does not know neither the full arm reward distributions neither their means.



source: Pandey et al.'s slide

Agent's Objective: Maximize cumulative rewards.

This is equivalent to finding the arm with the highest mean reward.

Example: Bernoulli Bandits

Recall: The Bernoulli distribution is the discrete probability distribution of a random variable which takes the value 1 with probability p and the value 0 with probability $q=1-p$, that is, the probability distribution of any single experiment that asks a yes–no question.

- Each action (arm when played) results in success or failure, **rewards are binary**.
- Mean reward for each arm represents the probability of success.
- Action (arm) $k \in \{1\dots K\}$ produces a success with probability $\theta_k \in [0,1]$.



$$\theta_1$$

win 0.6
of time



$$\theta_2$$

win 0.4
of time



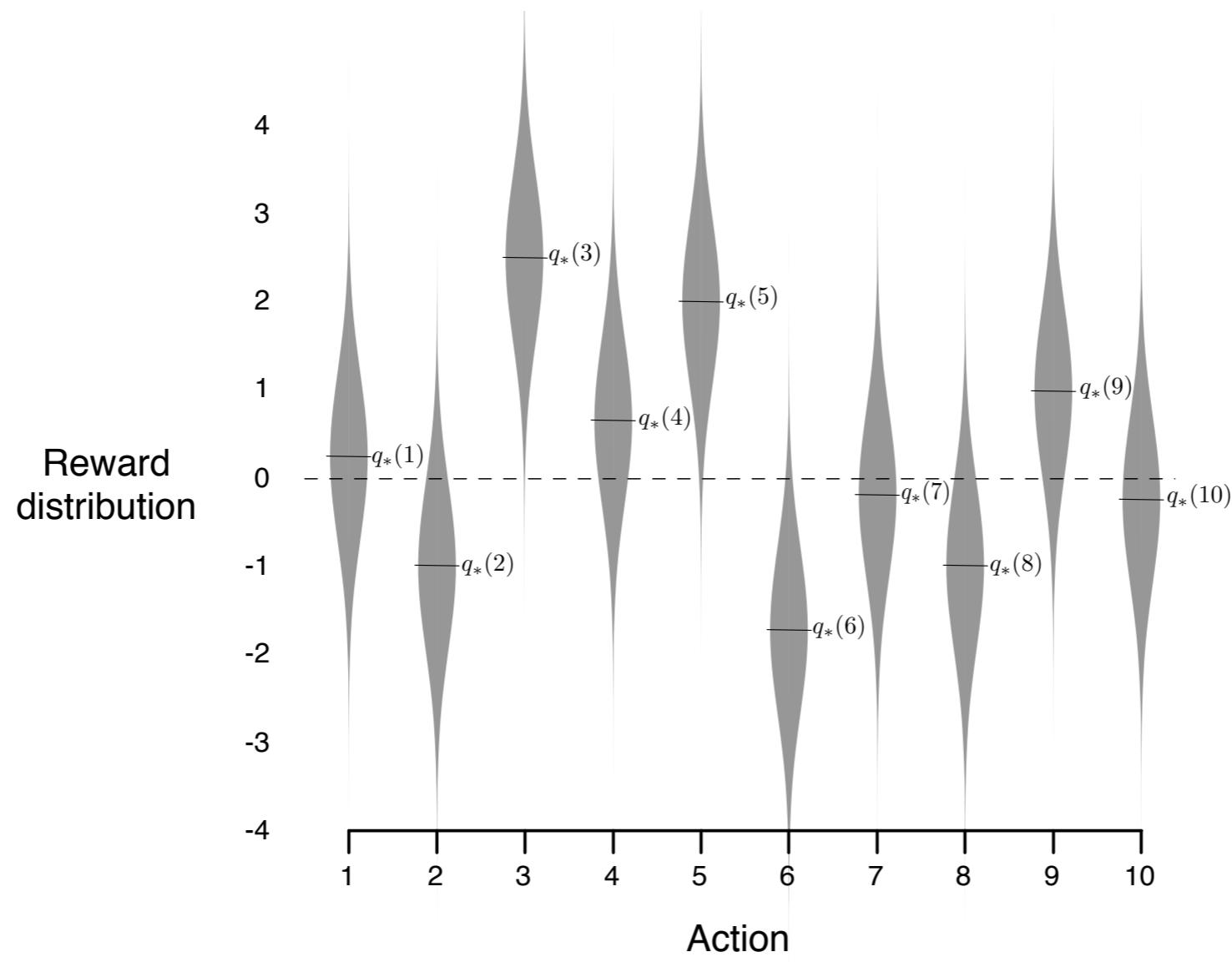
$$\theta_3$$

win 0.45
of time

Example: Gaussian Bandits

$$R_t \sim \mathcal{N}(q_*(a), 1)$$

- Each action (arm when played) results in a **real number**.
- Action (arm) $k \in \{1 \dots K\}$ produces on average reward equal to the mean of its Gaussian distribution.



Real world motivation: content presentation

We have two variations of content of a webpage, A and B, and we want to decide which one to display to engage more users.

- Two arm bandits: each arm corresponds to an content variation shown to users (not necessarily the same user).
- Mean rewards: the total percentage of users that would click on each invitation

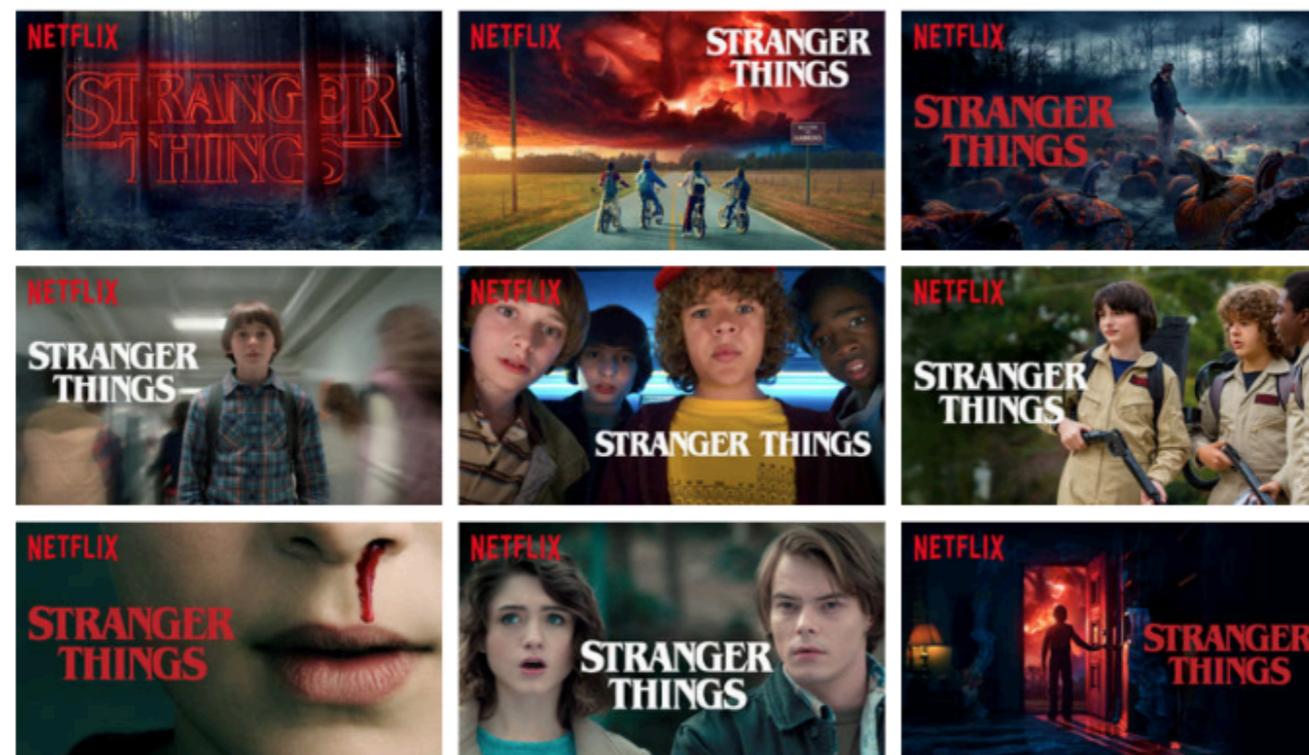
The screenshot shows a promotional page for a dinner with President Barack Obama. The top navigation bar features the "OBAMA BIDEN" logo. Below it, the main headline reads "DINNER WITH BARACK" and "Your chance to meet the President". There are two large, light blue rectangular buttons and a red "GET STARTED" button. The central image is a photograph of President Obama smiling at a table. Overlaid on the photo is the text "DINNER WITH BARACK" in a stylized font, along with "YOU'RE INVITED." and "WE'LL COVER YOUR AIRFARE." At the bottom of the page, there is a detailed legal disclaimer about the promotion's rules and restrictions, followed by the "OBAMA BIDEN" logo and links to "Privacy Policy" and "Terms of Service". A small note at the very bottom states that contributions are not tax deductible.

The screenshot shows the same promotional page as variation A, but with different visual elements. The main image features a group of people, including First Lady Michelle Obama and President Obama, sitting around a table. The text overlay "DINNER WITH BARACK" and "You're invited. We'll cover your airfare." is positioned on the right side of the image. The legal disclaimer at the bottom is identical to variation A. The "OBAMA BIDEN" logo and "GET STARTED" button are also present.

Real world motivation: NETFLIX artwork

For a particular movie, we want to decide what image to show (to all the NETFLIX users)

- Actions: uploading one of the K images to a user's home screen
- Ground-truth mean rewards (unknown): the % of NETFLIX users that will click on the title and watch the movie
- Estimated mean rewards: the average click rate observed (quality engagement, not clickbait)



The Exploration/Exploitation Dilemma

- Suppose you form estimates

$$Q_t(a) \approx q_*(a), \quad \forall a \quad \text{action-value estimates}$$

The Exploration/Exploitation Dilemma

- Suppose you form estimates

$$Q_t(a) \approx q_*(a), \quad \forall a \quad \text{action-value estimates}$$

- Define the greedy action at time t as

$$A_t^* \doteq \arg \max_a Q_t(a)$$

The Exploration/Exploitation Dilemma

- Suppose you form estimates

$$Q_t(a) \approx q_*(a), \quad \forall a \quad \text{action-value estimates}$$

- Define the greedy action at time t as

$$A_t^* \doteq \arg \max_a Q_t(a)$$

- If $A_t = A_t^*$ then you are exploiting

If $A_t \neq A_t^*$ then you are exploring

The Exploration/Exploitation Dilemma

- Suppose you form estimates

$$Q_t(a) \approx q_*(a), \quad \forall a \quad \text{action-value estimates}$$

- Define the greedy action at time t as

$$A_t^* \doteq \arg \max_a Q_t(a)$$

- If $A_t = A_t^*$ then you are exploiting
- If $A_t \neq A_t^*$ then you are exploring
- You can't do both, but you need to do both

The Exploration/Exploitation Dilemma

- Suppose you form estimates

$$Q_t(a) \approx q_*(a), \quad \forall a \quad \text{action-value estimates}$$

- Define the greedy action at time t as

$$A_t^* \doteq \arg \max_a Q_t(a)$$

- If $A_t = A_t^*$ then you are exploiting
- If $A_t \neq A_t^*$ then you are exploring
- You can't do both, but you need to do both
- You can never stop exploring, but maybe you should explore less with time.

Regret

- The **action-value** is the mean reward for action a ,

$$q_*(a) \doteq \mathbb{E}[R_t | A_t = a], \quad \forall a \in \{1, \dots, k\} \quad (\text{expected return})$$

- The **optimal value** is

$$v_* = q(a^*) = \max_{a \in \mathcal{A}} q_*(a)$$

- The **regret** is the opportunity loss for one step

$$I_t = \mathbb{E}[v_* - q_*(a_t)] \quad \text{reward} = -\text{regret}$$

- The **total regret** is the total opportunity loss

$$L_t = \mathbb{E} \left[\sum_{t=1}^T v_* - q_*(a_t) \right]$$

- Maximize cumulative reward = minimize total regret

Regret

- The **count** $N_t(a)$: the number of times that action a has been selected prior to time t
- The **gap** Δ_a is the difference in value between action a and optimal action a_* : $\Delta_a = v_* - q_*(a)$
- Regret is a function of gaps and the counts

$$\begin{aligned} L_t &= \mathbb{E} \left[\sum_{t=1}^T v_* - q_*(a_t) \right] \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)](v_* - q_*(a)) \quad \text{how come??} \\ &= \sum_{a \in \mathcal{A}} \mathbb{E}[N_t(a)]\Delta_a \end{aligned}$$

Forming Action-Value Estimates

- To simplify notation, let us focus on one action
 - We consider only its rewards, and its estimate after $n+1$ rewards:

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n - 1}$$

- How can we do this incrementally (without storing all the rewards)?
- Could store a running sum and count (and divide), or equivalently:

$$Q_{n+1} = Q_n + \frac{1}{n} [R_n - Q_n]$$

- This is a standard form for learning/update rules:

$$\text{NewEstimate} \leftarrow \text{OldEstimate} + \text{StepSize} [\text{Target} - \text{OldEstimate}]$$

Forming Action-Value Estimates

- To simplify notation, let us focus on one action
 - We consider only its rewards, and its estimate after $n+1$ rewards:

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n - 1}$$

- How can we do this incrementally (without storing all the rewards)?
- Could store a running sum and count (and divide), or equivalently:

$$Q_{n+1} = Q_n + \frac{1}{n} [R_n - Q_n]$$

- This is a standard form for learning/update rules: error

NewEstimate \leftarrow OldEstimate + StepSize $\left[\text{Target} - \text{OldEstimate} \right]$

Derivation of incremental update

$$Q_n \doteq \frac{R_1 + R_2 + \cdots + R_{n-1}}{n - 1}$$

$$\begin{aligned} Q_{n+1} &= \frac{1}{n} \sum_{i=1}^n R_i \\ &= \frac{1}{n} \left(R_n + \sum_{i=1}^{n-1} R_i \right) \\ &= \frac{1}{n} \left(R_n + (n-1) \frac{1}{n-1} \sum_{i=1}^{n-1} R_i \right) \\ &= \frac{1}{n} \left(R_n + (n-1)Q_n \right) \\ &= \frac{1}{n} \left(R_n + nQ_n - Q_n \right) \\ &= Q_n + \frac{1}{n} [R_n - Q_n], \end{aligned}$$

Non-stationary bandits

- Suppose the true action values change slowly over time
 - then we say that the problem is nonstationary
- In this case, sample averages are not a good idea
 - Why?

Non-stationary bandits

- Suppose the true action values change slowly over time
 - then we say that the problem is nonstationary
- In this case, sample averages are not a good idea
- Better is an “exponential, recency-weighted average”:

Give larger weights for the most recent reward

$$\begin{aligned} Q_{n+1} &\doteq Q_n + \alpha [R_n - Q_n] \\ &= (1 - \alpha)^n Q_1 + \sum_{i=1}^n \alpha(1 - \alpha)^{n-i} R_i, \end{aligned}$$

where $\alpha \in (0,1]$ and constant

The smaller the i , the smaller the multiplier-> forgetting earlier rewards

Action selection in multi-armed bandits

Fixed exploration period + Greedy

1. Allocate a fixed time period to exploration when you try bandits **uniformly at random**
2. Estimate mean rewards for all actions:
$$Q_t(a) = \frac{1}{N_t(a)} \sum_{i=1}^{t-1} r_i \mathbf{1}(A_i = a)$$
3. Select the action that is optimal for the estimated mean rewards, breaking ties at random: $a_t = \operatorname{argmax}_{a \in \mathcal{A}} Q_t(a)$
4. GOTO 2

Fixed exploration period + Greedy

- After the fixed exploration period we have formed the following reward estimates



$$Q_t(a_1) = 0.3$$



$$Q_t(a_2) = 0.5$$



$$Q_t(a_3) = 0.1$$

Q: Will the greedy method always pick the second action?

- Greedy can lock onto a suboptimal action forever. Why?
- ⇒ Greedy has linear total regret

ϵ -Greedy Action Selection

- In greedy action selection, you always exploit
- In ϵ -greedy, you are usually greedy, but with probability ϵ you instead pick an action at random (possibly the greedy action again)
- This is perhaps the simplest way to balance exploration and exploitation

ε -Greedy Action Selection

A simple bandit algorithm

Initialize, for $a = 1$ to k :

$$\begin{aligned} Q(a) &\leftarrow 0 \\ N(a) &\leftarrow 0 \end{aligned}$$

Repeat forever:

$$\begin{aligned} A &\leftarrow \begin{cases} \arg \max_a Q(a) & \text{with probability } 1 - \varepsilon \\ \text{a random action} & \text{with probability } \varepsilon \end{cases} \quad (\text{breaking ties randomly}) \\ R &\leftarrow \text{bandit}(A) \\ N(A) &\leftarrow N(A) + 1 \\ Q(A) &\leftarrow Q(A) + \frac{1}{N(A)} [R - Q(A)] \end{aligned}$$

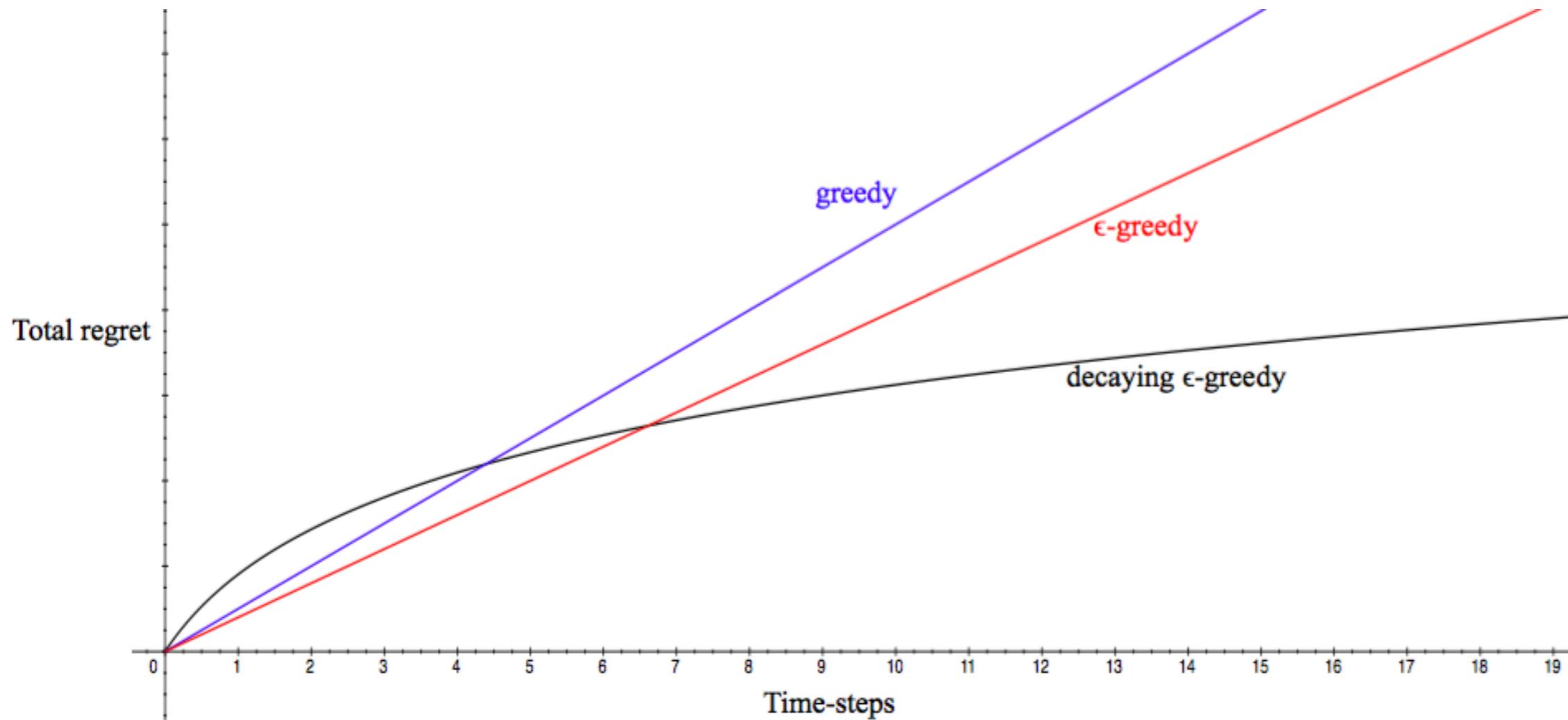
ϵ -Greedy Algorithm

- The ϵ -greedy algorithm continues to explore forever
 - With probability $1 - \epsilon$ select $a_t = \operatorname{argmax}_{a \in \mathcal{A}} Q_t(a)$
 - With probability ϵ select a random action
- Constant ϵ ensures minimum regret

$$I_t \geq \frac{\epsilon}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \Delta_a$$

- $\Rightarrow \epsilon$ -greedy has linear total regret

Counting Regret



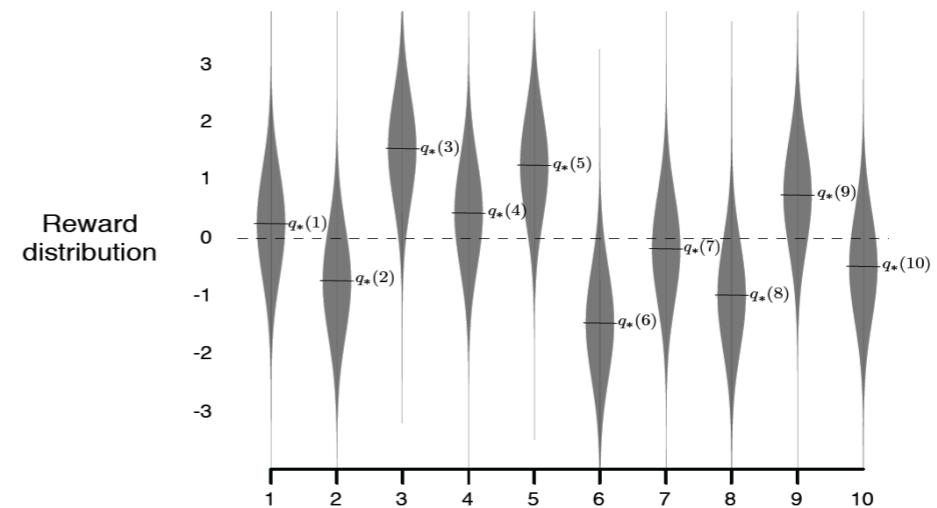
- If an algorithm forever explores it will have linear total regret
means it will never converge???
- If an algorithm never explores it will have linear total regret

Average reward for three algorithms

We sample 10 arm bandits instantiations:

action-value $q^*(a)$: expected reward

$$q_*(a) \sim \mathcal{N}(0, 1)$$
$$R_t \sim \mathcal{N}(q_*(a), 1)$$

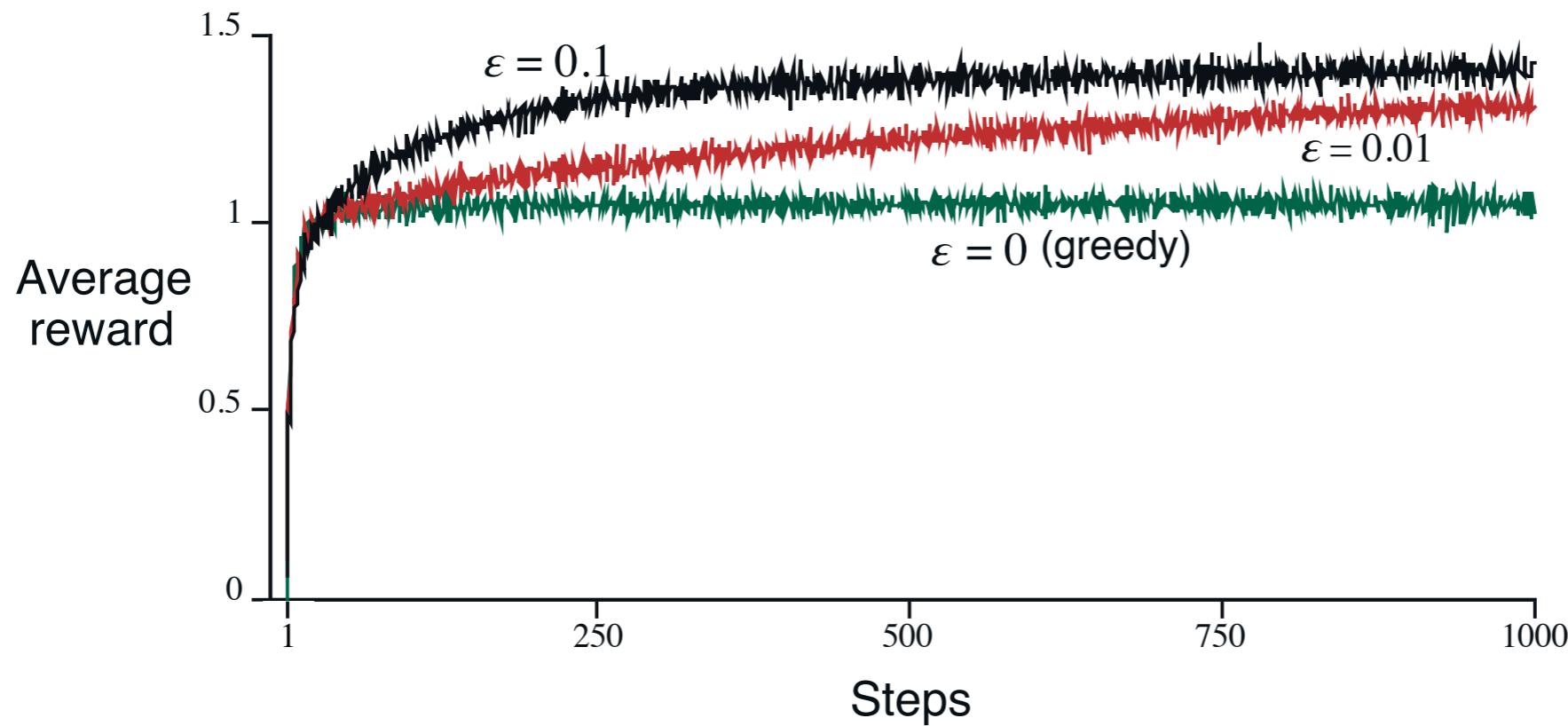
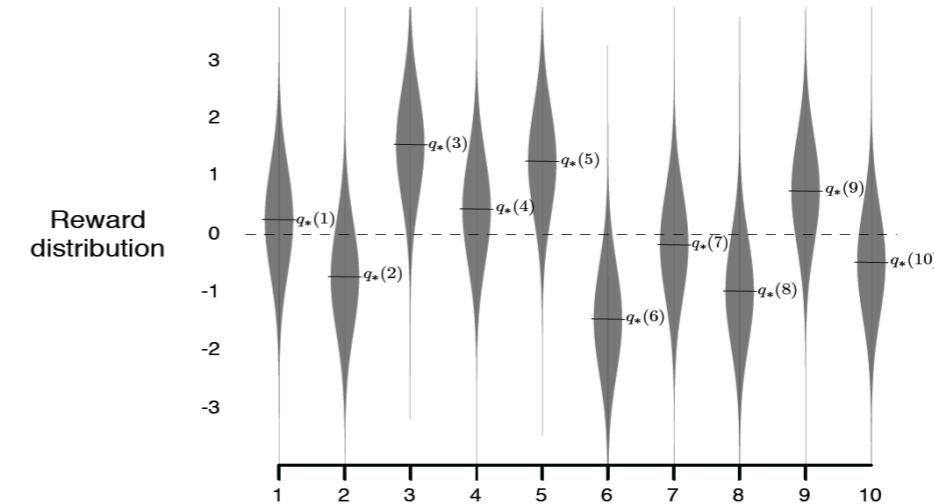


Average reward for three algorithms

We sample 10 arm bandits instantiations:

$$q_*(a) \sim \mathcal{N}(0, 1)$$

$$R_t \sim \mathcal{N}(q_*(a), 1)$$

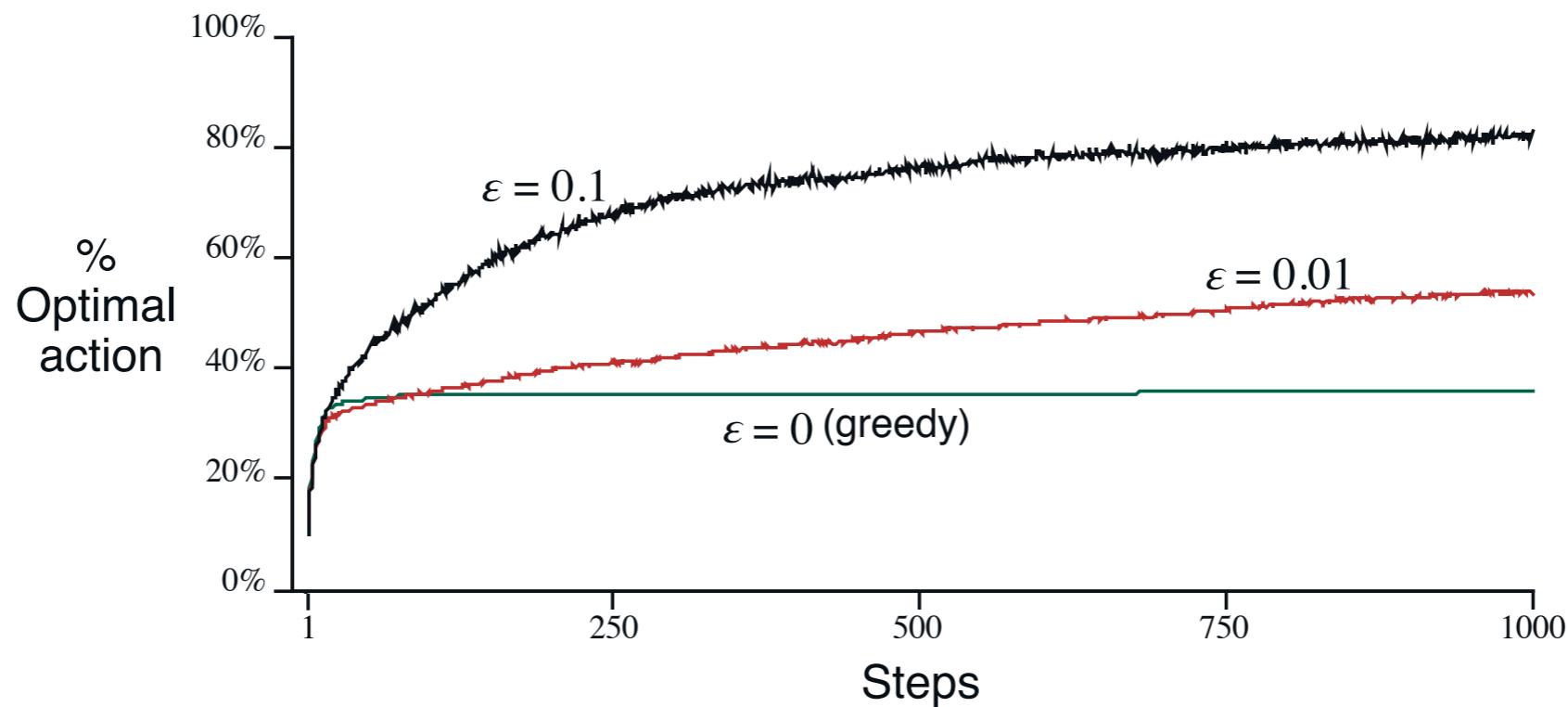
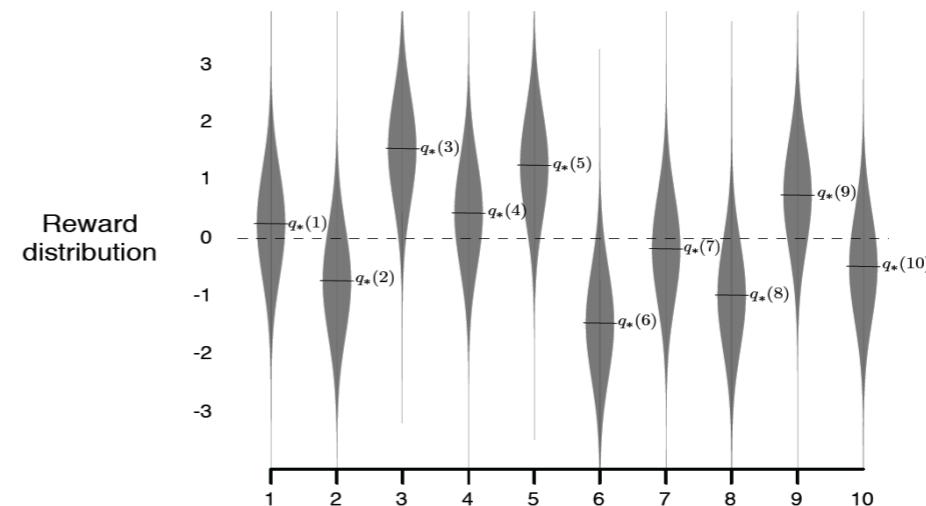


Q: In the limit (after infinite number of steps), which method will result in the largest average reward?

Optimal action for three algorithms

We sample 10 arm bandits instantiations:

$$q_*(a) \sim \mathcal{N}(0, 1)$$
$$R_t \sim \mathcal{N}(q_*(a), 1)$$

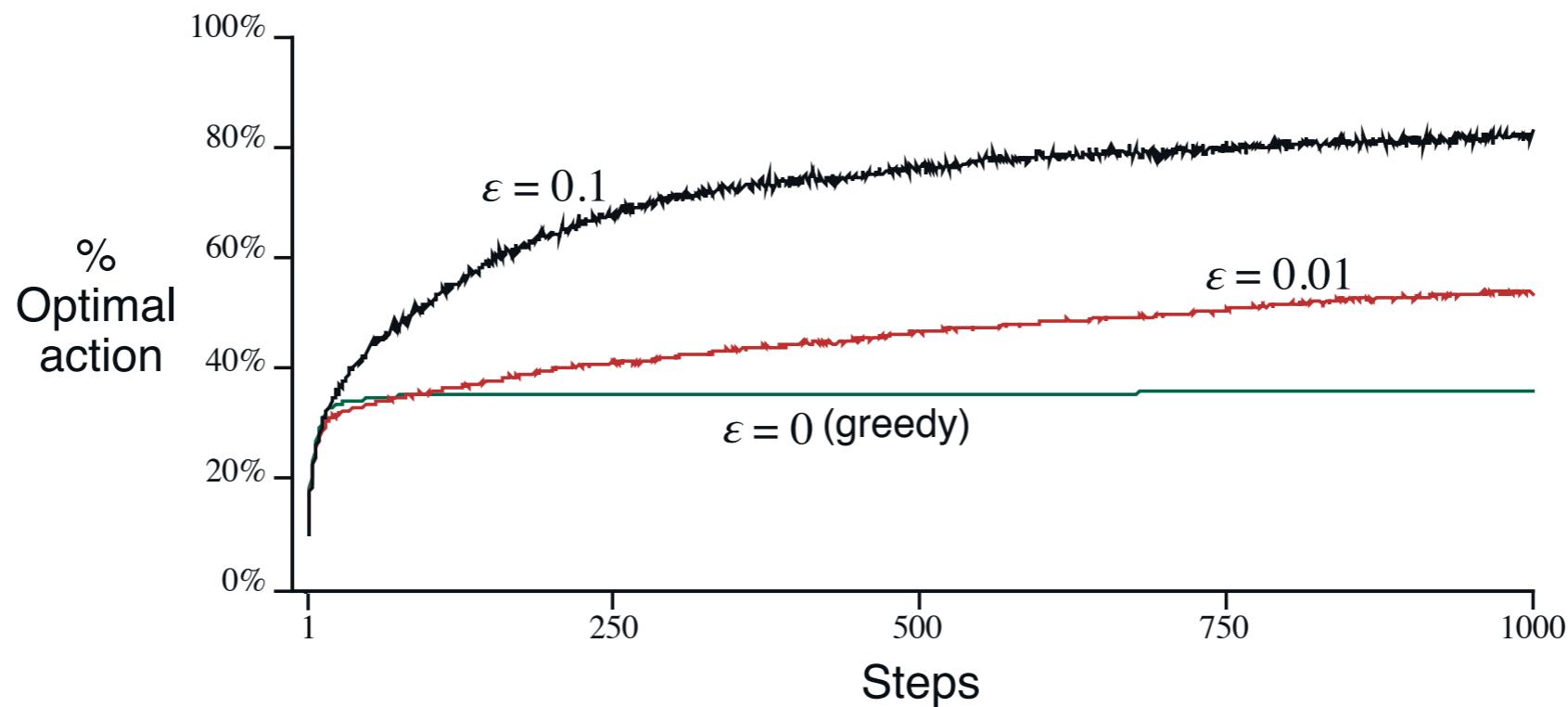
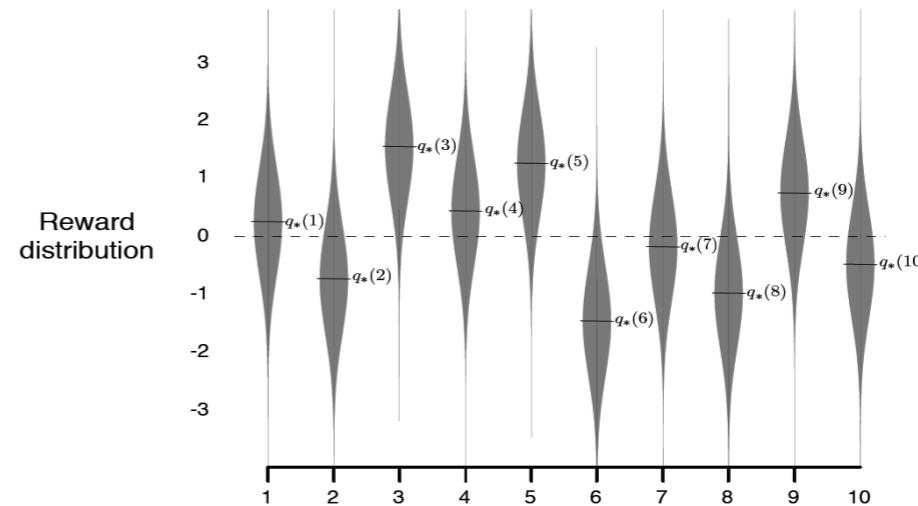


Q: Which method will find the optimal action in the limit?

Optimal action for three algorithms

We sample 10 arm bandits instantiations:

$$q_*(a) \sim \mathcal{N}(0, 1)$$
$$R_t \sim \mathcal{N}(q_*(a), 1)$$



Q: Does the performance of those methods depend on the initialization of the action value estimates?

Optimistic Initialization

- Simple and practical ideas: initialise $Q(a)$ to a high value
- Update action value by incremental Monte-Carlo evaluation
- Starting with $N(a) > 0$,

$$Q_t(a_t) = Q_{t-1}(a_t) + \frac{1}{N_t(a_t)} (r_t - Q_{t-1}(a_t))$$

just an incremental estimate of sample mean, including one 'hallucinated' initial optimistic value

- Encourages systematic exploration early on
- But optimistic greedy can still lock onto a suboptimal action if rewards are stochastic.

Optimistic Initial Values

We initialize with the following reward estimates for Bernoulli bandits



$$Q_t(a_1) = 1$$



$$Q_t(a_2) = 1$$



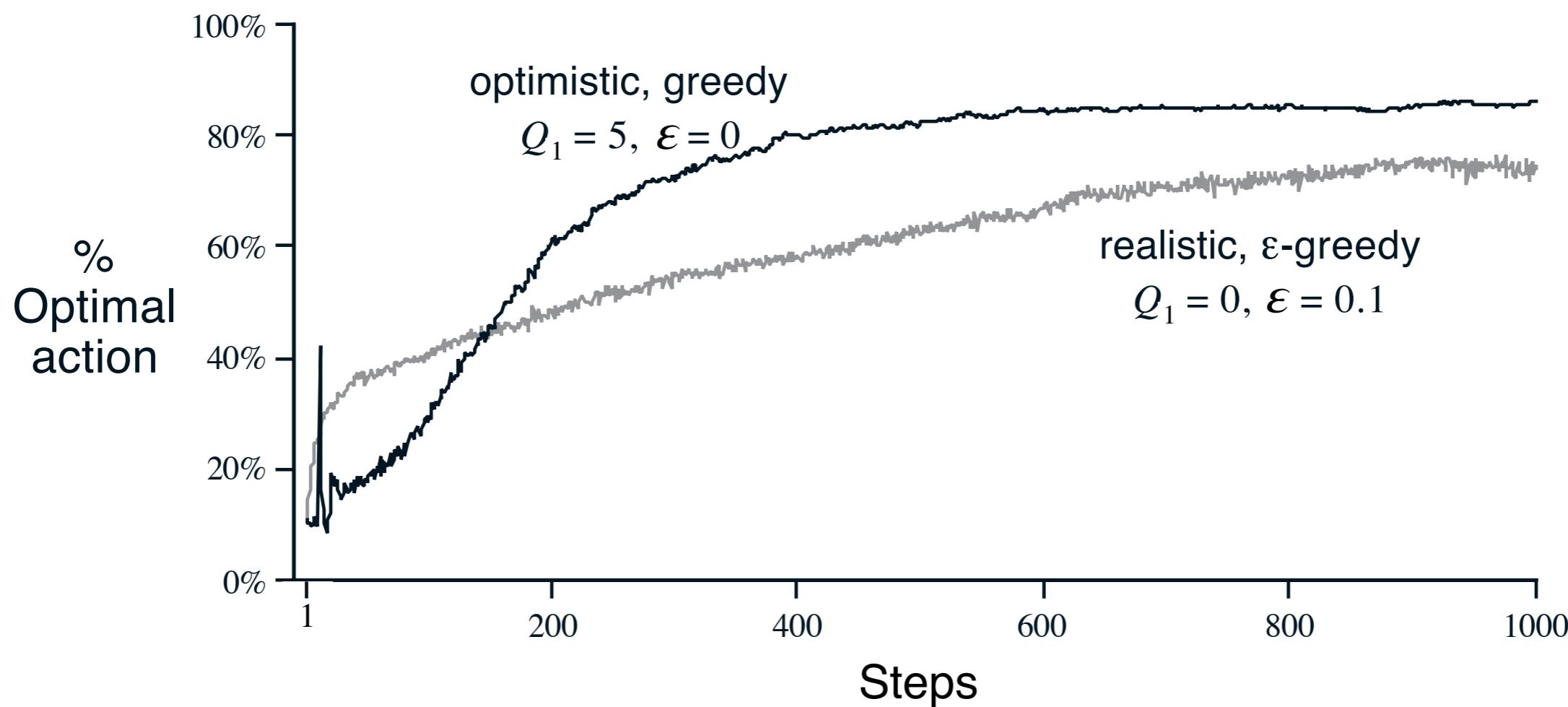
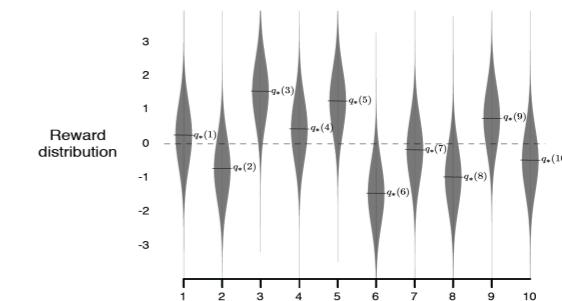
$$Q_t(a_3) = 1$$

Q: When it is possible that greedy action selection will not try out all the actions?

Optimistic Initial Values

- Suppose we initialize the action values optimistically ($Q_1(a) = 5$), e.g., on the 10-armed testbed

$$q_*(a) \sim \mathcal{N}(0, 1)$$
$$R_t \sim \mathcal{N}(q_*(a), 1)$$



Achieving sub-linear total regret

We need to reason about **uncertainty** of our action value estimates



1000 pulls,
600 wins
 $Q_t(a_1)=0.6$



1000 pulls,
400 wins
 $Q_t(a_2)=0.4$



10 pulls,
4 wins
 $Q_t(a_1)=0.4$

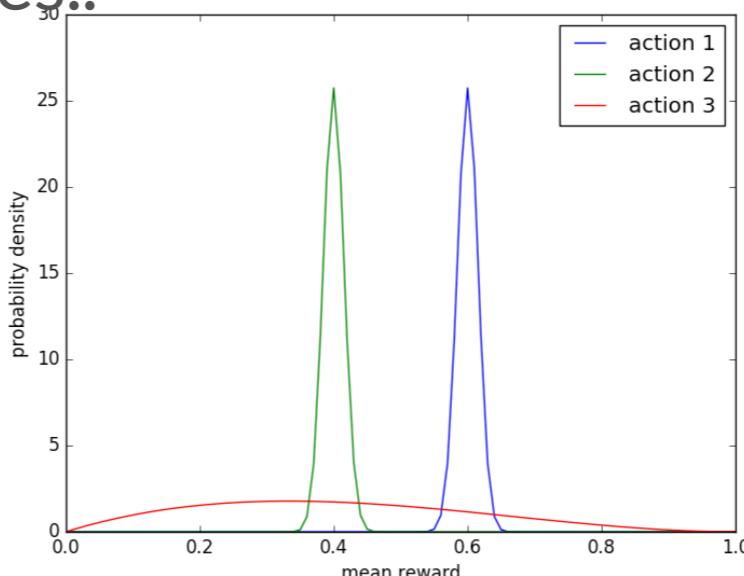
Epsilon-greedy

Repeat forever:

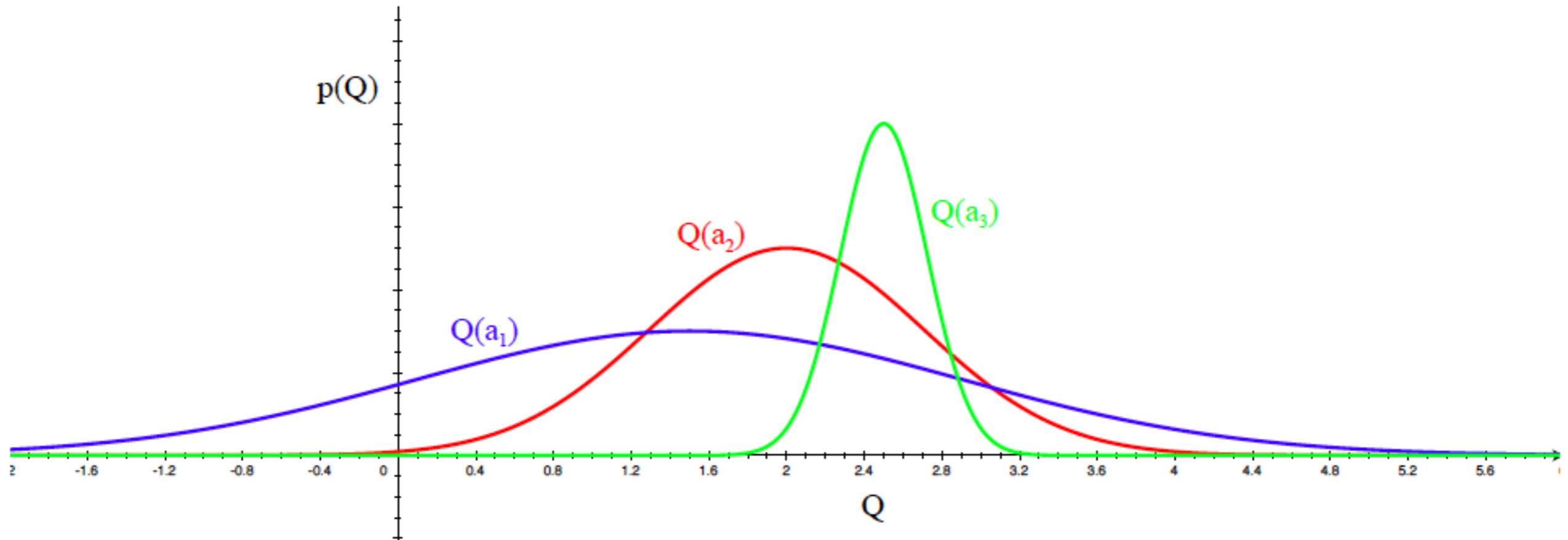
```

$$A \leftarrow \begin{cases} \arg \max_a Q(a) & \text{with probability } 1 - \varepsilon \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$
 (breaking ties randomly)  
 $R \leftarrow \text{bandit}(A)$   
 $N(A) \leftarrow N(A) + 1$   
 $Q(A) \leftarrow Q(A) + \frac{1}{N(A)} [R - Q(A)]$ 
```

The problem with using mean estimates is that we cannot reason about uncertainty of those estimates..

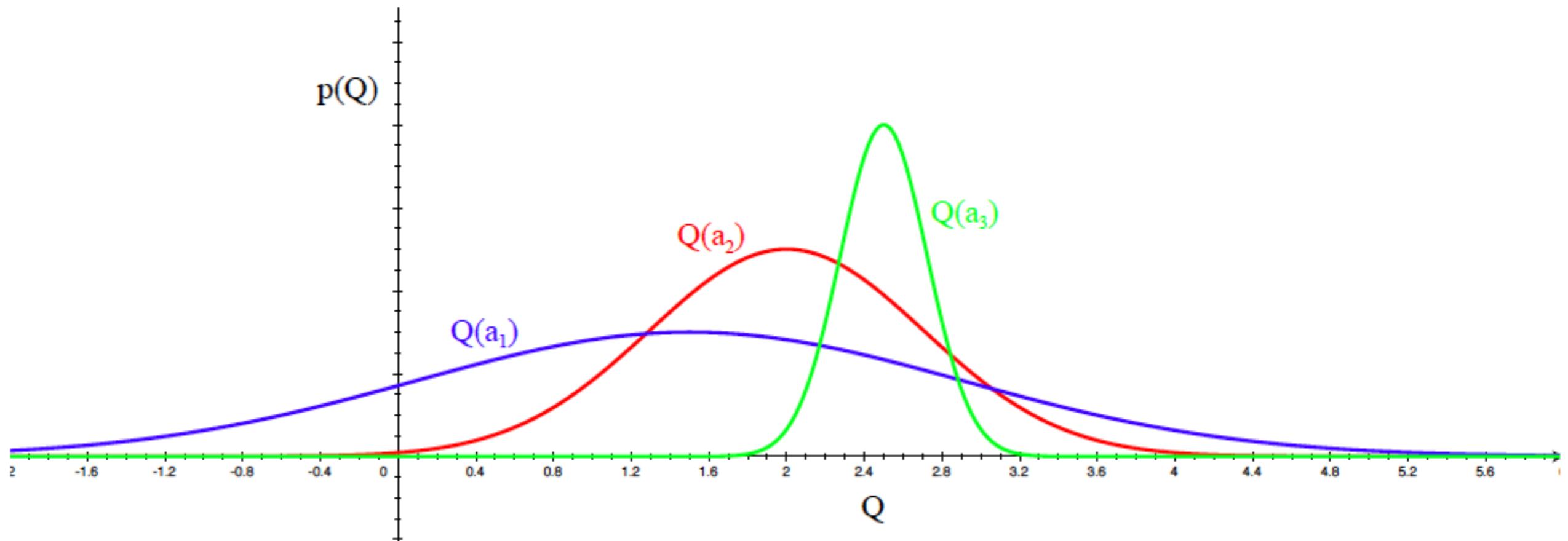


Uncertainty guides Exploration



- The more **uncertain** we are about an action-value
- The more important it is to explore that action
- It could turn out to be the best action

Uncertainty guides Exploration



- We are then less uncertain about the value
- And more likely to pick another action
- Until we converge to the best action

Upper Confidence Bounds

- Estimate an **upper confidence** $U_t(a)$ for each action value
- Such that with high probability

$$q_*(a) \leq Q_t(a) + U_t(a)$$

Estimated mean

Estimated Upper
Confidence

- This upper confidence depends on the number of times action a has been selected
 - Small $N_t(a) \Rightarrow$ large $U_t(a)$ (estimated value is uncertain)
 - Large $N_t(a) \Rightarrow$ small $U_t(a)$ (estimated value is accurate)
- Select action maximizing **Upper Confidence Bound (UCB)**

$$a_t = \operatorname{argmax}_{a \in \mathcal{A}} Q_t(a) + U_t(a)$$

Hoeffding's Inequality

Theorem (Hoeffding's Inequality)

Let X_1, \dots, X_t be i.i.d. random variables in $[0,1]$, and let $\bar{X}_t = \frac{1}{t} \sum_{\tau=1}^t X_\tau$ be the sample mean. Then

$$\mathbb{P} [\mathbb{E}[X] > \bar{X}_t + u] \leq e^{-2tu^2}$$

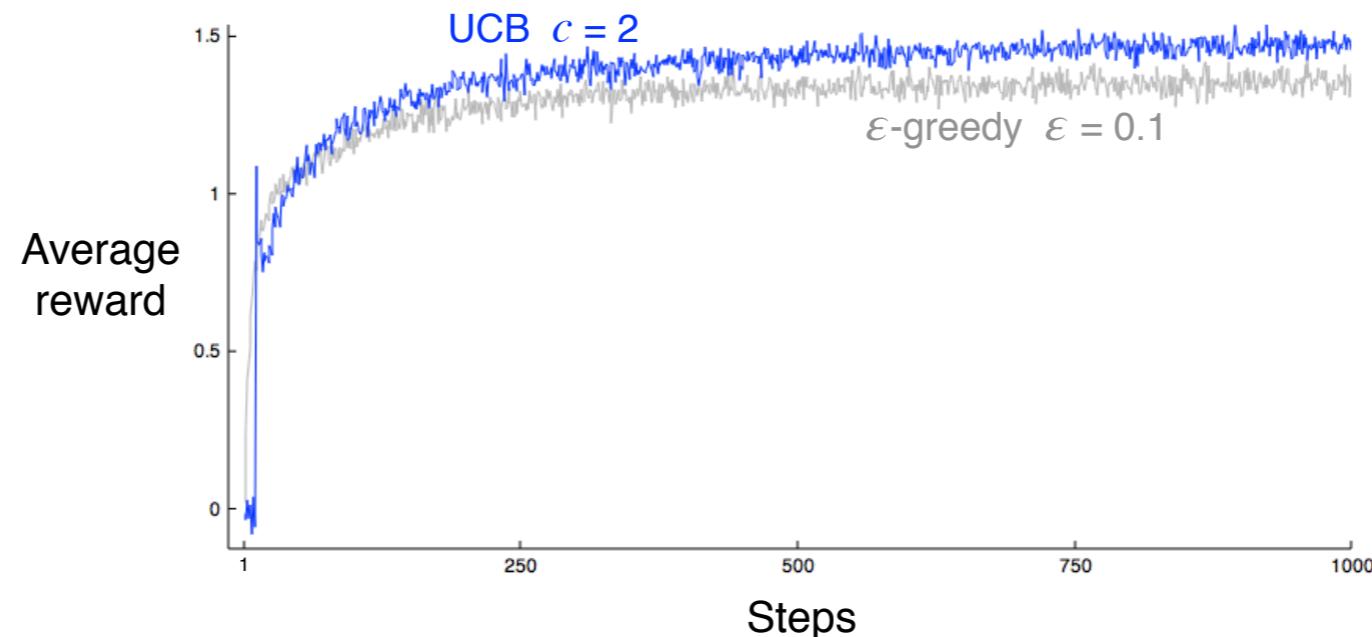
- We will apply Hoeffding's Inequality to rewards of the bandit conditioned on selecting action a

$$\mathbb{P} [Q(a) > \hat{Q}_t(a) + U_t(a)] \leq e^{-2N_t(a)U_t(a)^2}$$

Upper Confidence Bound (UCB)

- A clever way of reducing exploration over time
- Estimate an upper bound on the true action values
- Select the action with the largest (estimated) upper bound

$$A_t \doteq \arg \max_a \left[Q_t(a) + c \sqrt{\frac{\log t}{N_t(a)}} \right]$$



Bayesian Bandits

- So far we have made no assumptions about the reward distributions.
 - In UCB we just considered some bounds on rewards
- Bayesian bandits exploit **prior knowledge of rewards**, $p[\mathcal{R}]$
- They compute **posterior** distribution of rewards $p[\mathcal{R} \mid h_t]$
$$h_t = a_1, r_1, \dots, a_{t-1}, r_{t-1}$$
- Use posterior to guide exploration: we simply sample from the posterior!

Bayesian learning for model parameters

Step 1: Given n data, $D = x_{1\dots n} = \{x_1, x_2, \dots, x_n\}$ write down the expression for likelihood:

$$p(\textcolor{red}{D} | \theta)$$

Step 2: Specify a prior: $p(\theta)$

Step 3: Compute the posterior:

$$p(\theta | D) = \frac{p(\textcolor{red}{D} | \theta)p(\theta)}{p(D)}$$

$$p(\theta|D) = p(\theta \text{ and } D)/p(D)$$

Thompson Sampling

Represent a distribution for the reward of each bandit as opposed to its mean estimate alone. At each timestep:

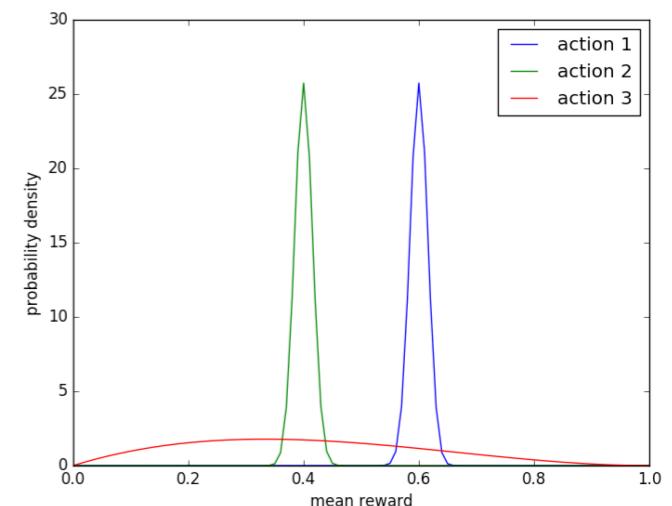
1. Sample from the reward distribution:

$$\bar{\theta}_1 \sim \hat{p}(\theta_1), \bar{\theta}_2 \sim \hat{p}(\theta_2), \dots, \bar{\theta}_k \sim \hat{p}(\theta_k)$$

2. Choose action $a = \arg \max_a \mathbb{E}_{\bar{\theta}}[r(a)]$

3. Observe the reward

4. Update the reward posterior distributions: $\hat{p}(\theta_1), \hat{p}(\theta_2) \dots \hat{p}(\theta_k)$



Q: why we use argmax in step 2 and we do not add any noise?

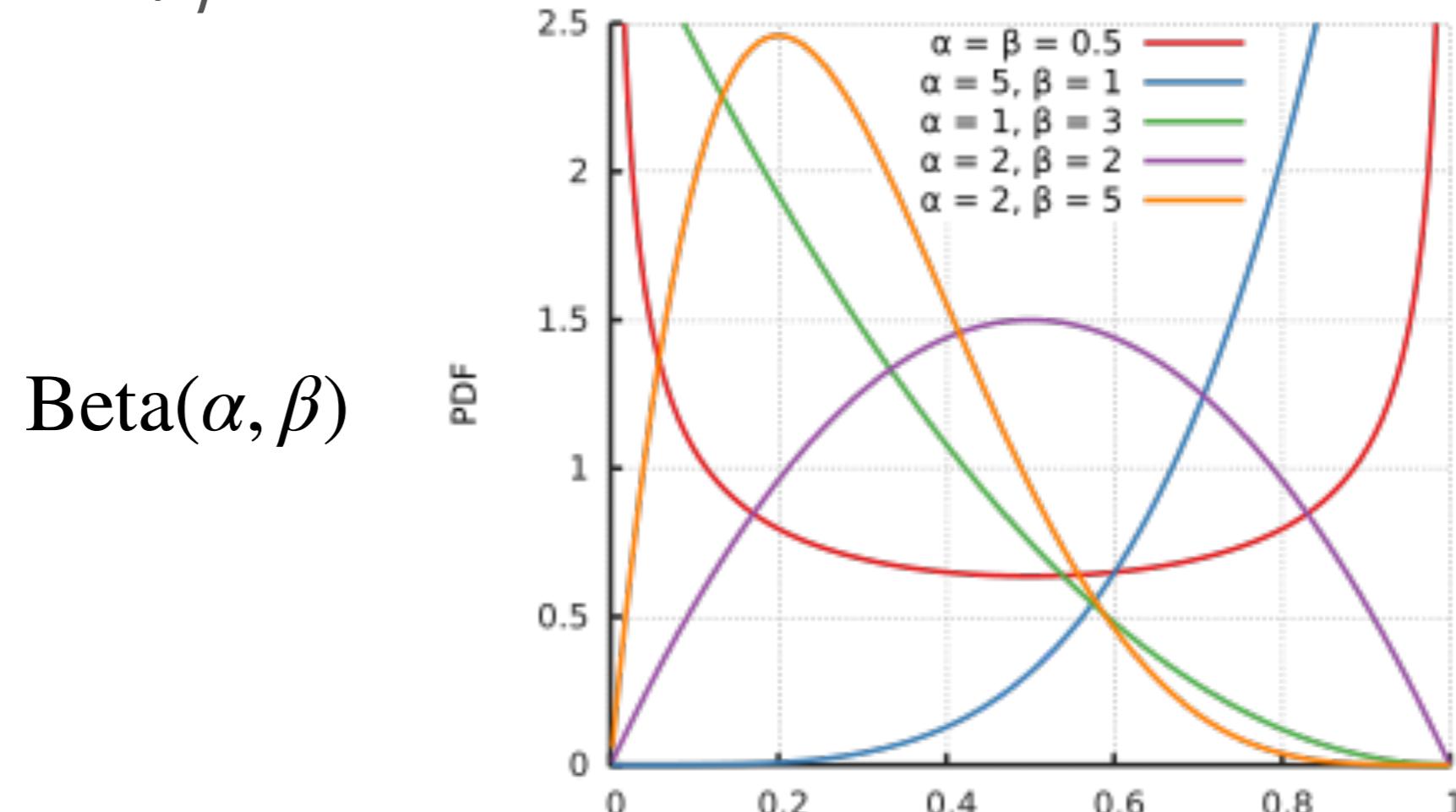
Bernoulli bandits - Prior

Let's consider a Beta distribution **prior** over the mean rewards of the Bernoulli bandits:

$$p(\theta_k) = \frac{\Gamma(\alpha_k + \beta_k)}{\Gamma(\alpha_k)\Gamma(\beta_k)} \theta_k^{\alpha_k-1} (1-\theta_k)^{\beta_k-1}$$
$$\Gamma(n) = (n-1)!$$

The mean is $\frac{\alpha}{\alpha + \beta}$

The larger the $\alpha + \beta$ the more concentrated the distribution



Bernoulli bandits-Posterior

Let's consider a Beta distribution prior over the mean rewards of the Bernoulli bandits:

$$p(\theta_k) = \frac{\Gamma(\alpha_k + \beta_k)}{\Gamma(\alpha_k)\Gamma(\beta_k)} \theta_k^{\alpha_k-1} (1 - \theta_k)^{\beta_k-1} \quad \Gamma(n) = (n - 1)!$$

The posterior is also a Beta! Because beta is conjugate distribution for the Bernoulli distribution.

A closed form solution for the bayesian update, possible only for conjugate distributions!

$$(\alpha_k, \beta_k) \leftarrow \begin{cases} (\alpha_k, \beta_k) & \text{if } x_t \neq k \\ (\alpha_k, \beta_k) + (r_t, 1 - r_t) & \text{if } x_t = k. \end{cases}$$

Greedy VS Thompson for Bernoulli bandits

Algorithm 1 BernGreedy(K, α, β)

```

1: for  $t = 1, 2, \dots$  do
2:   #estimate model:
3:   for  $k = 1, \dots, K$  do
4:      $\hat{\theta}_k \leftarrow \alpha_k / (\alpha_k + \beta_k)$ 
5:   end for
6:
7:   #select and apply action:
8:    $x_t \leftarrow \text{argmax}_k \hat{\theta}_k$ 
9:   Apply  $x_t$  and observe  $r_t$ 
10:
11:  #update distribution:
12:   $(\alpha_{x_t}, \beta_{x_t}) \leftarrow (\alpha_{x_t}, \beta_{x_t}) + (r_t, 1 - r_t)$ 
13: end for

```

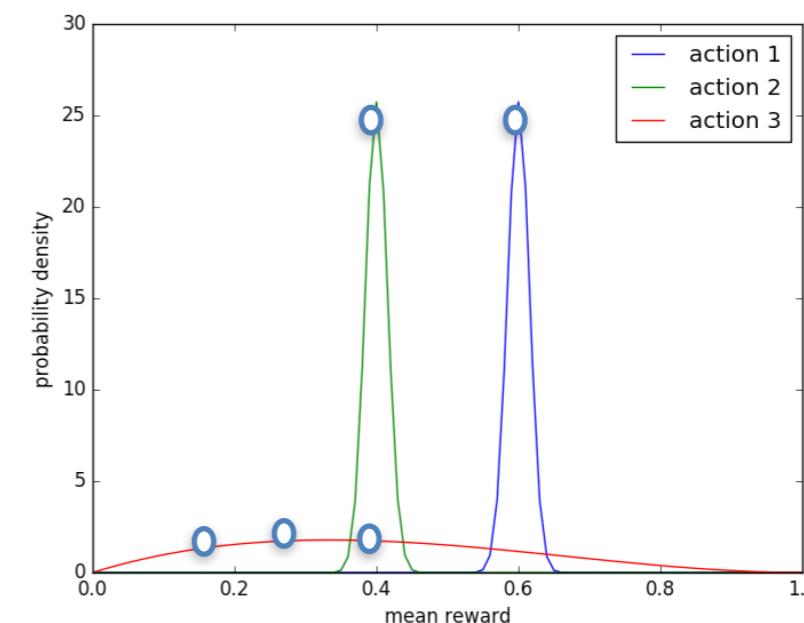
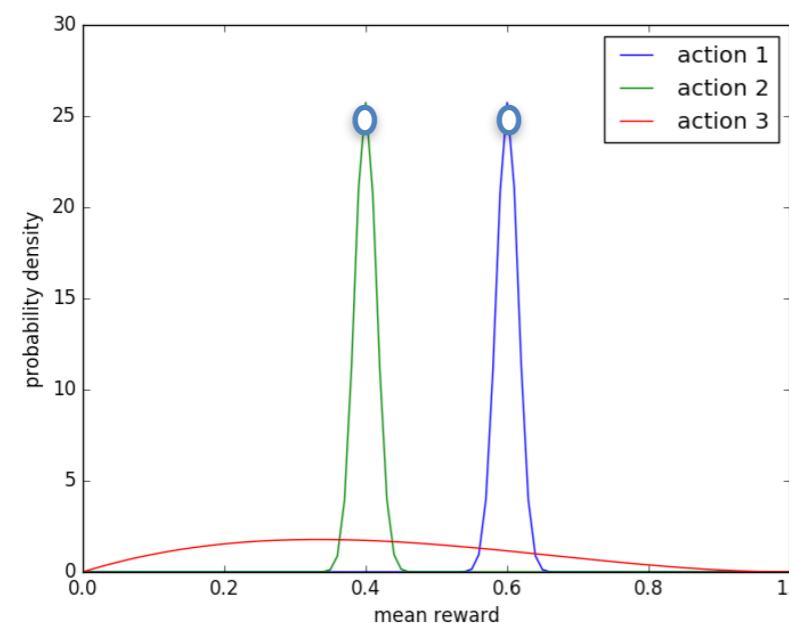
a: success
b: failure

Algorithm 2 BernThompson(K, α, β)

```

1: for  $t = 1, 2, \dots$  do
2:   #sample model:
3:   for  $k = 1, \dots, K$  do
4:     Sample  $\hat{\theta}_k \sim \text{beta}(\alpha_k, \beta_k)$ 
5:   end for
6:
7:   #select and apply action:
8:    $x_t \leftarrow \text{argmax}_k \hat{\theta}_k$ 
9:   Apply  $x_t$  and observe  $r_t$ 
10:
11:  #update distribution:
12:   $(\alpha_{x_t}, \beta_{x_t}) \leftarrow (\alpha_{x_t}, \beta_{x_t}) + (r_t, 1 - r_t)$ 
13: end for

```



Contextive Bandits (a.k.a Associative Search)

- A contextual bandit is a tuple (A, S, R)
- A is a known set of k actions (or “arms”)
- $\mathcal{S} = \mathbb{P}[s]$ is an unknown distribution over states (or “contexts”)
- $\mathcal{R}_s^a(r) = \mathbb{P}[r | s, a]$ is an unknown probability distribution over rewards
- At each time t
 - Environment generates state $s_t \sim \mathcal{S}$
 - Agent selects action $a_t \in \mathcal{A}$
 - Environment generates reward $r_t \sim \mathcal{R}_{s_t}^{a_t}$
- The goal is to maximise cumulative reward $\sum_{\tau=1}^t r_\tau$

Bayesian Bandits VS A/B Hypothesis testing

- Given two actions, A,B, traditional hypothesis testing would require to gather enough samples to decide whether a preference of A versus B is statistically significant.
- Bayesian bandits adapt the actions over time based on the rewards seen thus far: they accomplish smaller cumulative regret

<https://analytics.googleblog.com/2013/01/multi-armed-bandit-experiments.html>

Real world motivation: Personalized NETFLIX artwork



For a particular title **and a particular user**, we will use the **contextual** multi-armed bandit formulation to decide what image to show per title **per user**

- Actions: uploading an image (available for this movie title) to a user's home screen
- Mean rewards (unknown): the % of NETFLIX users that will click on the title and watch the movie
- Estimated mean rewards: the average click rate (+quality engagement, not clickbait)
- **Context** (s) : user attributes, e.g., language preferences, gender of films she has watched, time and day of the week, etc.

Netflix Artwork: <https://medium.com/netflix-techblog/artwork-personalization-c589f074ad76>

Real world motivation: Google

<https://towardsdatascience.com/beyond-a-b-testing-multi-armed-bandit-experiments-1493f709f804>

<https://analytics.googleblog.com/2013/01/multi-armed-bandit-experiments.html>