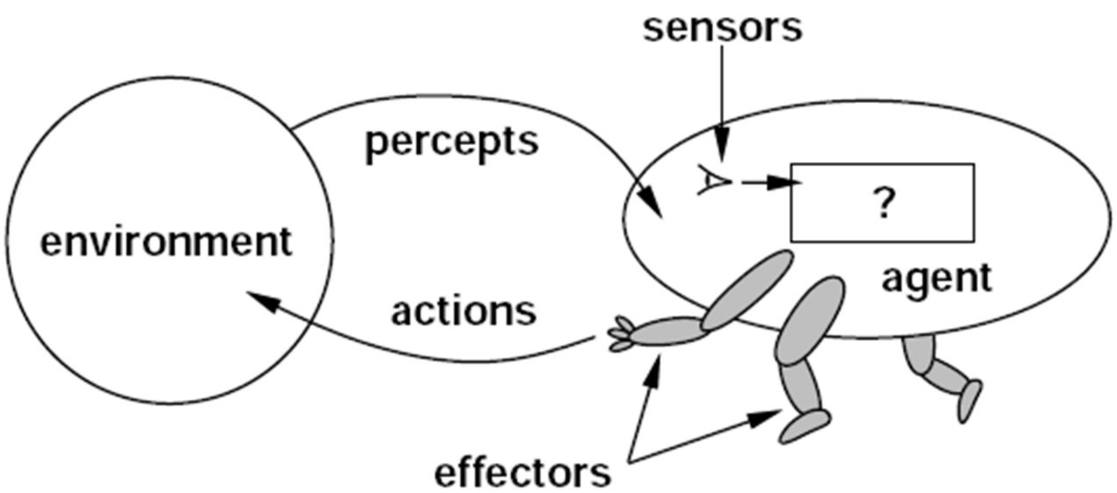
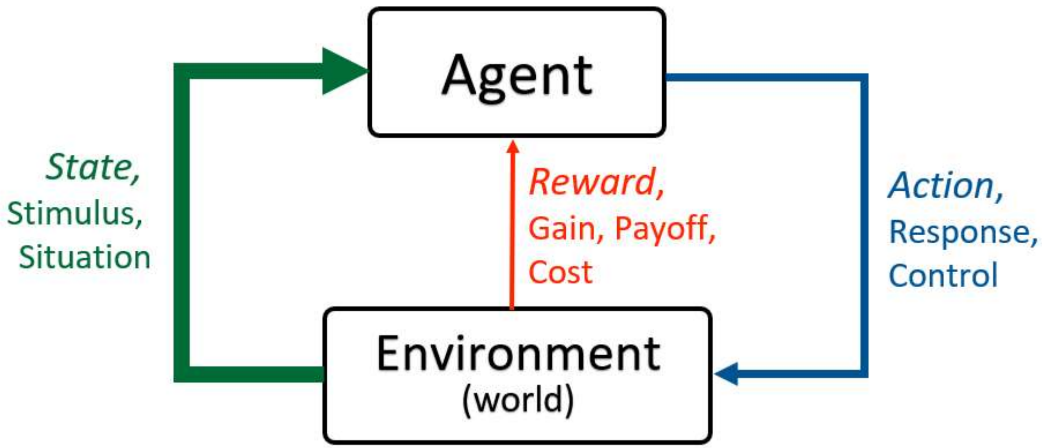
Reinforcement Learning 1

Anders Lyhne Christensen


sensors
percepts
environment
?
agent
actions
effectors


Agent
State,
Stimulus,
Situation
Reward,
Gain, Payoff,
Cost
Action,
Response,
Control
Environment
(world)

What is the reward?

Challenges

- An agent that typically has **no prior knowledge** must learn how to solve a task through interaction with an environment.

- The agent may receive a reward after each action, or only from time to time – perhaps only very infrequently.

Example: Learning to play football from scratch is challenging: when should an agent be rewarded?

Sparse and deceptive rewards can complicate learning.

Multiarmed/*k*-armed Bandit Problem

Multiarmed/*k*-armed Bandit Problem

Suppose there are *k* bandits.

Action a_i is like pulling a slot machine arm with random payoff function $R(a_i)$

…

$R(a_1)$ $R(a_2)$ … $R(a_k)$

Definitions

Reward at time t

$$q_*(a) \doteq \mathbb{E}[R_t \mid A_t = a]$$

Expected reward

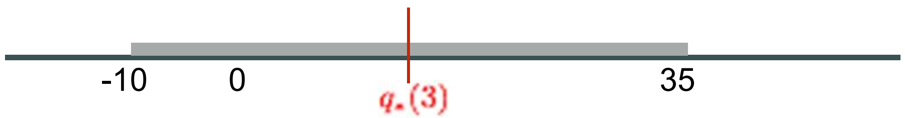
Arbitrary action

Expected value

Action selected at time t

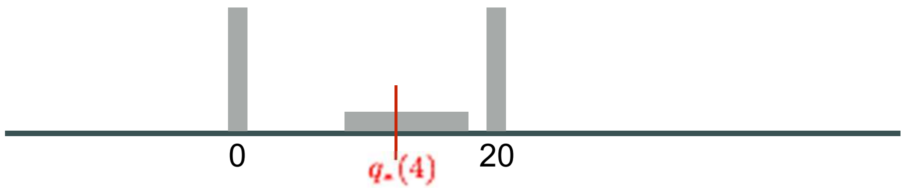
Assume that there are *k*=4 bandits

- Action: pull arm 1 — Reward is always 8
  - value of arm 1 is $q_*(1) =$
- Action: pull arm 2 — 88% chance of 0 and 12% chance of 100!
  - value of arm 2 is $q_*(2) = .88 \times 0 + .12 \times 100 =$
- Action: pull arm 3 — Randomly between -10 and 35, equiprobable



$q_*(3) =$

- Action: pull arm 4 — a third 0, a third 20, a third from {8,9,…, 18}



$q_*(4) =$

We don't know $q_*(a)$, but we want to estimate it. We call our estimate

$$\underbrace{Q_t}_{\text{Estimated reward at time } t}(\underbrace{a}_{\text{Action}})$$

The *k*-armed Bandit Problem

- On each of a sequence of *time steps*, t=1, 2, 3, …,
  you choose an action A_t from k possibilities, and receive a real-valued *reward* R_t
- The reward depends only on the action taken;
  it is identically, independently distributed (i.i.d.):

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

true values

- These true values are *unknown.* The distribution is unknown.
- Nevertheless, you must maximize your total reward
- You must both try actions to learn their values (*explore*),
  and prefer those that appear best (*exploit*)

Bounded Reward Assumption

A common assumption we will make is that rewards are in a bounded interval [-R_{max}, R_{max}].

I.e., for each i, $\Pr(R(a_i) \in [-R_{max}, R_{max}]) = 1$.

Note that results are available for other types of assumptions, e.g. Gaussian distributions of rewards.

Multi-Armed Bandits: Application examples

- **Clinical Trials**
  - Arms: possible treatments
  - Arm pulls: application of treatment to individual
  - Rewards: outcome of treatment
  - Objective: maximize cumulative reward = maximize benefit to trial population (or find best treatment quickly)

- **Online Advertising**
  - Arms: different ads/ad-types for a webpage
  - Arm pulls: displaying an ad upon a page access
  - Rewards: click through
  - Objective: maximize cumulative reward = maximize clicks (or find best add quickly)

Assuming that you have *k* bandits, how would you maximize the rewards?

…

$R(a_1)$ $R(a_2)$ … $R(a_k)$

Remember: bandits pay out a reward chosen from probability distributions. Each bandit has its own probability distribution, and you have no idea what they are.

UniformBandit Algorithm

1. Pull each arm a fixed number (***w***) times
2. Return arm with best average reward

…

r_{11} r_{12} … r_{1w} r_{21} r_{22} … r_{2w} r_{k1} r_{k2} … r_{kw}

Imagine that we have to maximize the reward over a total of 1000 arm pulls.

What should we do?

The Exploration/Exploitation Dilemma

- Suppose you form estimates

$$Q_t(a) \approx q_*(a), \quad \forall a$$

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 at the same time.

Action-Value Methods

- Methods that learn **action-value estimates** and nothing else
- For example, estimate action values as *sample averages*:

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}}$$

- The sample-average estimates converge to the true values *If* the action is taken an infinite number of times

$$\lim_{N_t(a)\to\infty} Q_t(a) \;=\; q_*(a)$$

The number of times action a has been taken by time t

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

Assume that an action has been taken $n - 1$ times already. Our estimated reward of taking the action an n'th time is then:

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

Not computationally efficient

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

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Repeat forever:

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

$R \leftarrow bandit(A)$

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

$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\big[R - Q(A)\big]$

Averaging as the learning rule

This is a standard form for learning/update rules:

Potentially noisy

$$\mathit{NewEstimate} \leftarrow \mathit{OldEstimate} + \mathit{StepSize}\,\underbrace{\left[\mathit{Target} - \mathit{OldEstimate}\right]}_{\text{Error}}$$

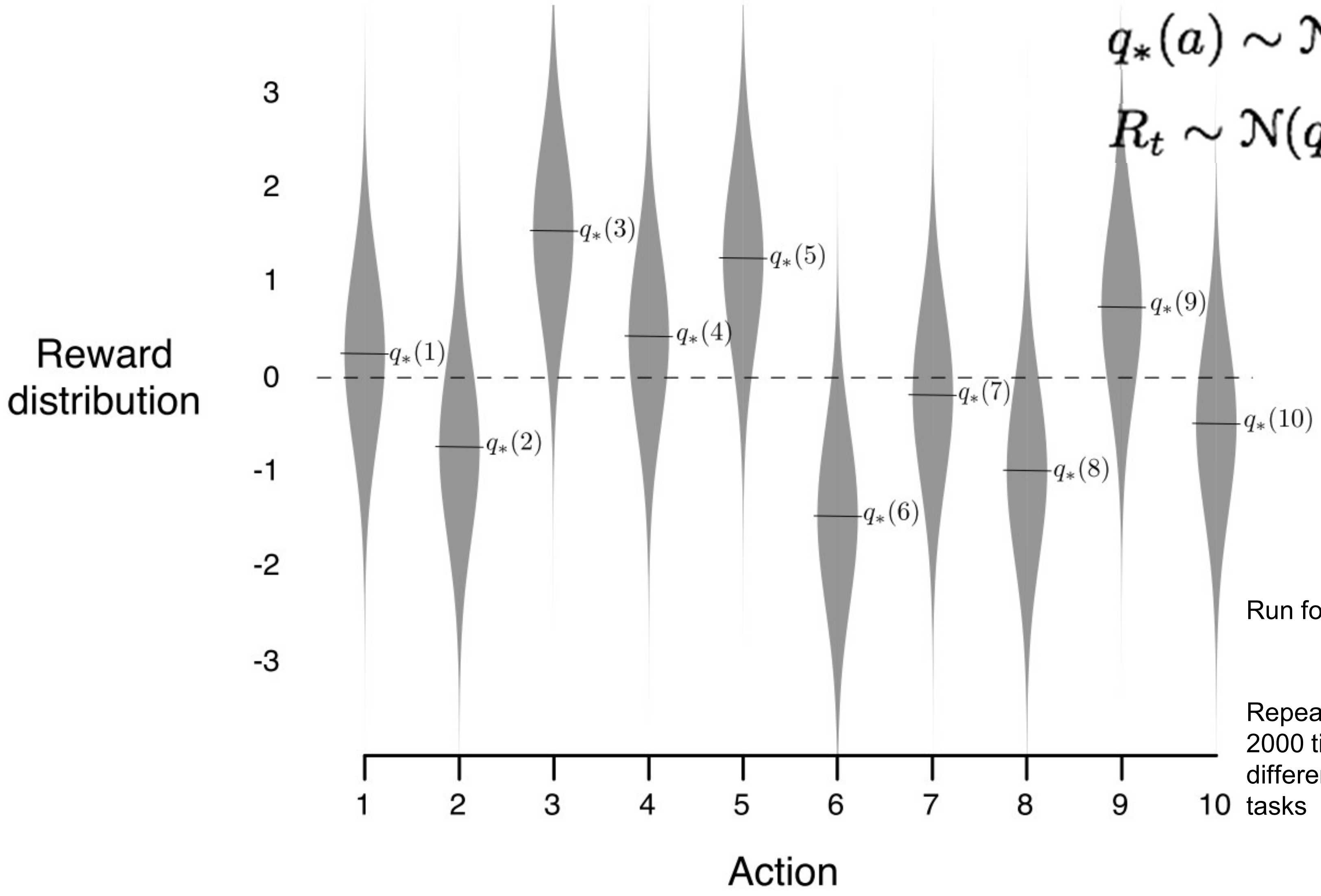
Learning rate α

Note, that the "*Target*" is presumed to indicate a desirable direction in which to move (may be noisy). The "*Target*" here is the *n*'th reward.

How well does ε-greedy action selection work?

Let's test for different values of ε.

The 10-armed Testbed



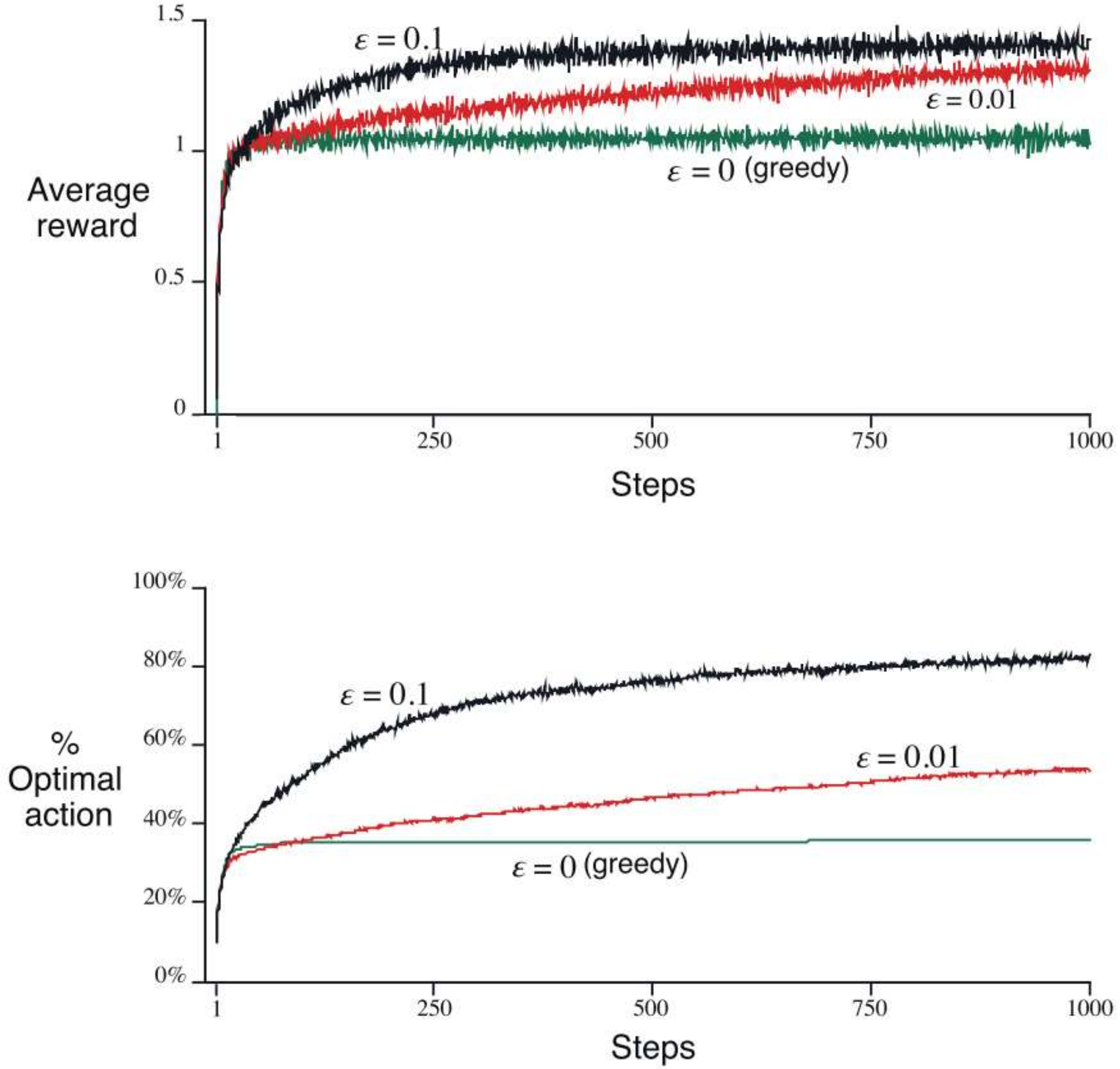
$$q_*(a) \sim \mathcal{N}(0, 1)$$

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

Run for 1000 steps

Repeat the experiment 2000 times with different bandit tasks

ε-Greedy Methods on the 10-Armed Testbed



Tracking a Non-stationary Problem

- Suppose the true action values **change slowly over time -** then we say that the problem is *non-stationary*
- In this case, sample averages are not a good idea (Why?)
- Better is an "*exponential, recency-weighted average*":

$$Q_{n+1} \doteq Q_n + \alpha\Big[R_n - Q_n\Big]$$

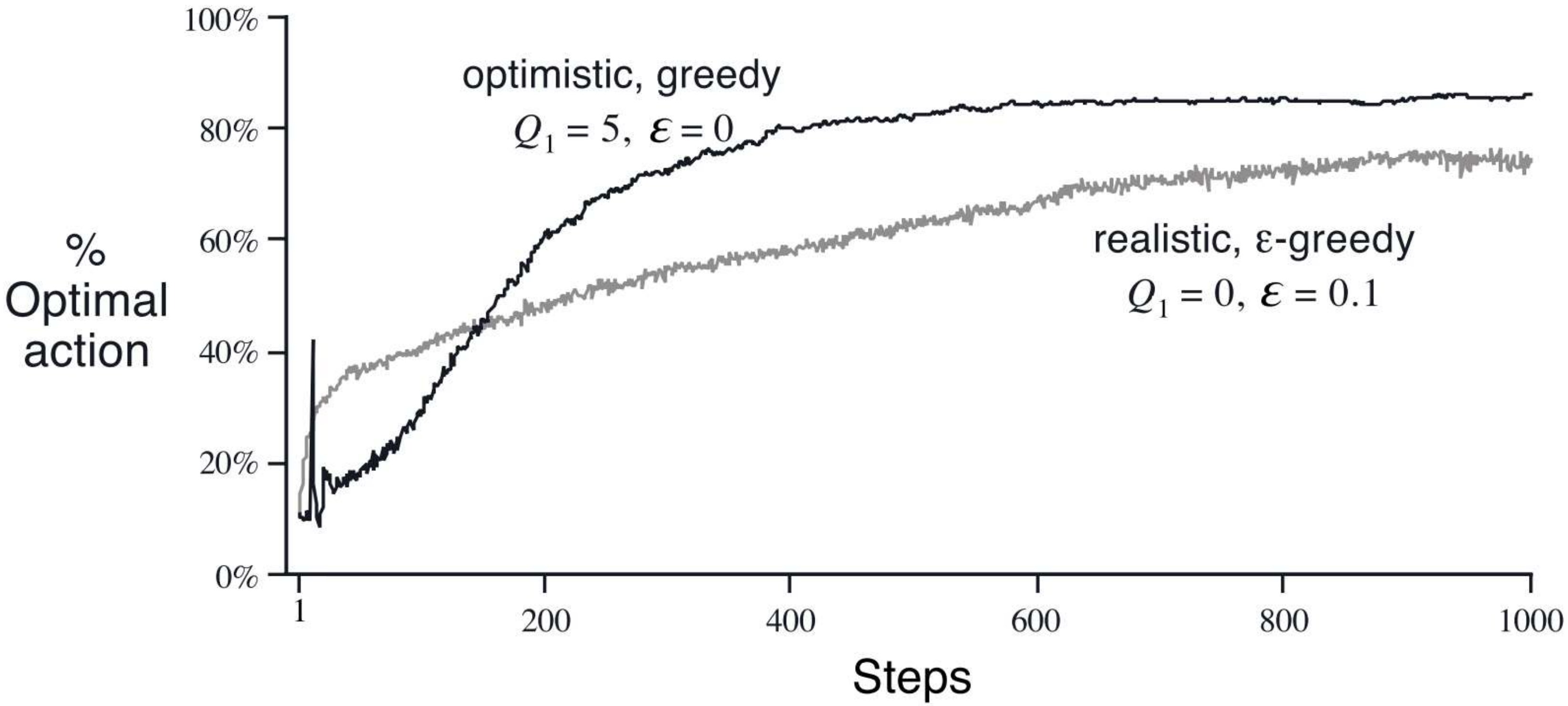
Where α, the step-size or learning rate, is constant.

- What is the effect if setting α = 1?
- What is the effect if setting α = 0?
- Which value should we assign to α?

We will still focus on stationary problems, but remove the initial bias over time by setting α to a constant value.

Optimistic Initial Values

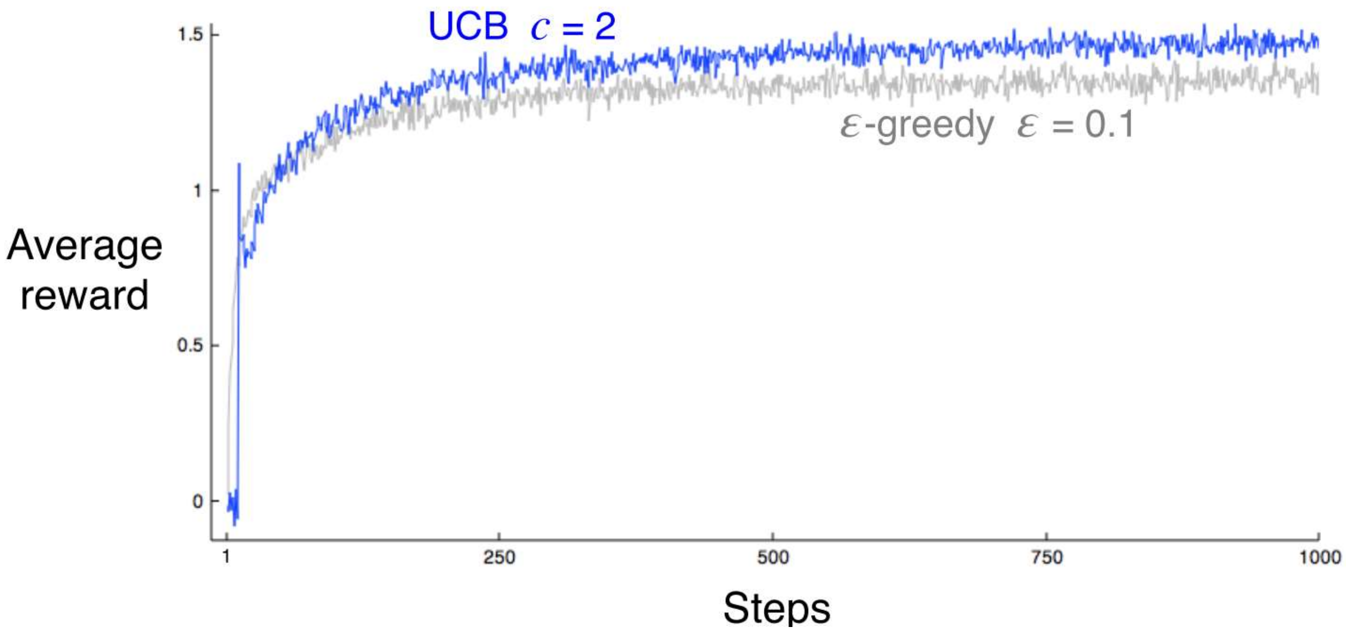
- All methods so far depend on $Q_1(a)$, i.e., they are biased. So far, we have used $Q_1(a) = 0$.
- Suppose we initialize the action values *optimistically* ($Q_1(a) = 5$), on the 10-armed testbed (with $\alpha = 0.1$)



Upper Confidence Bound (UCB) action selection

- 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

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



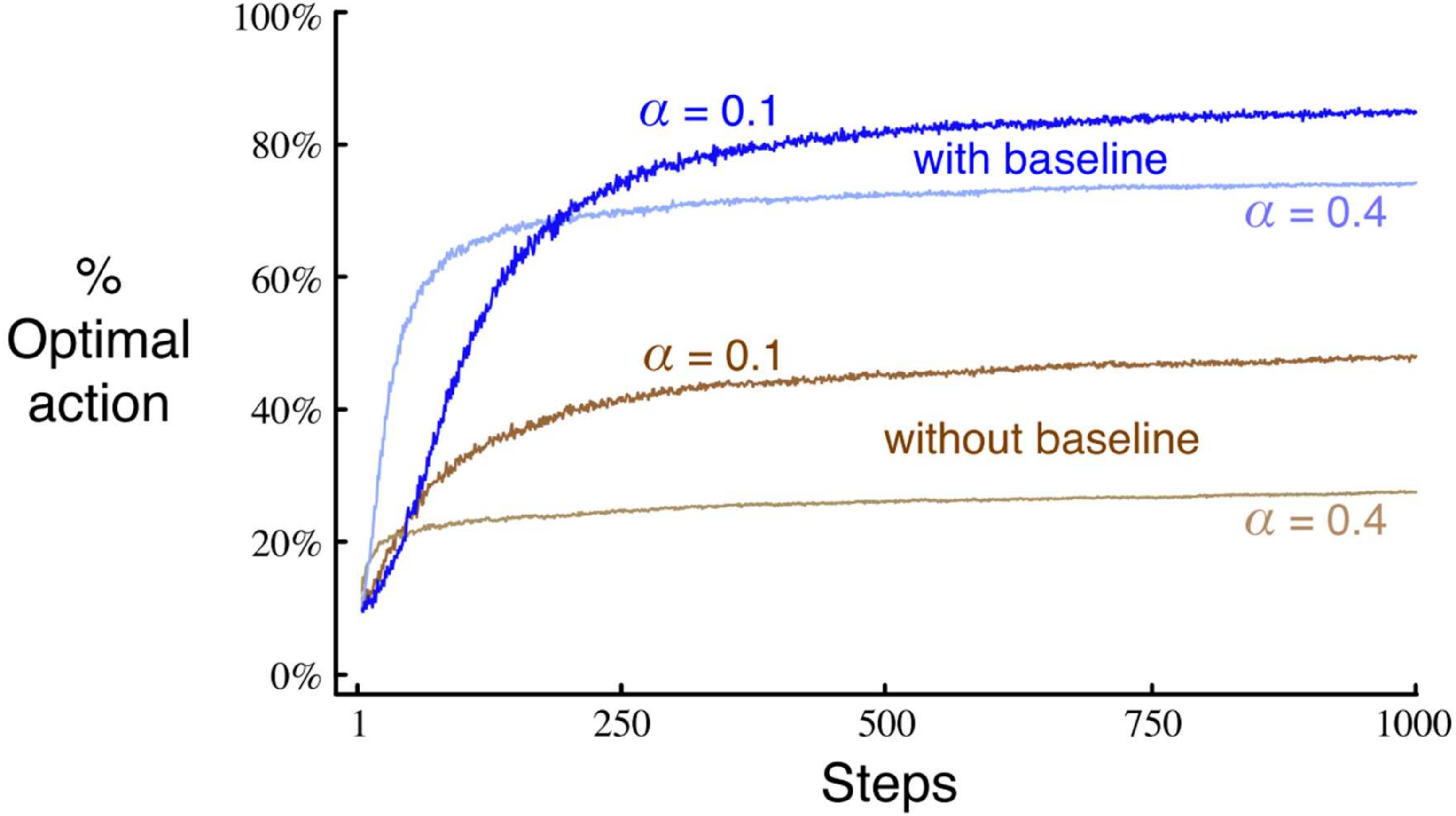
Gradient Bandit Algorithm

- We don't try to estimate or rewards, we just keep track of our preference (or probability) for choosing an action:

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

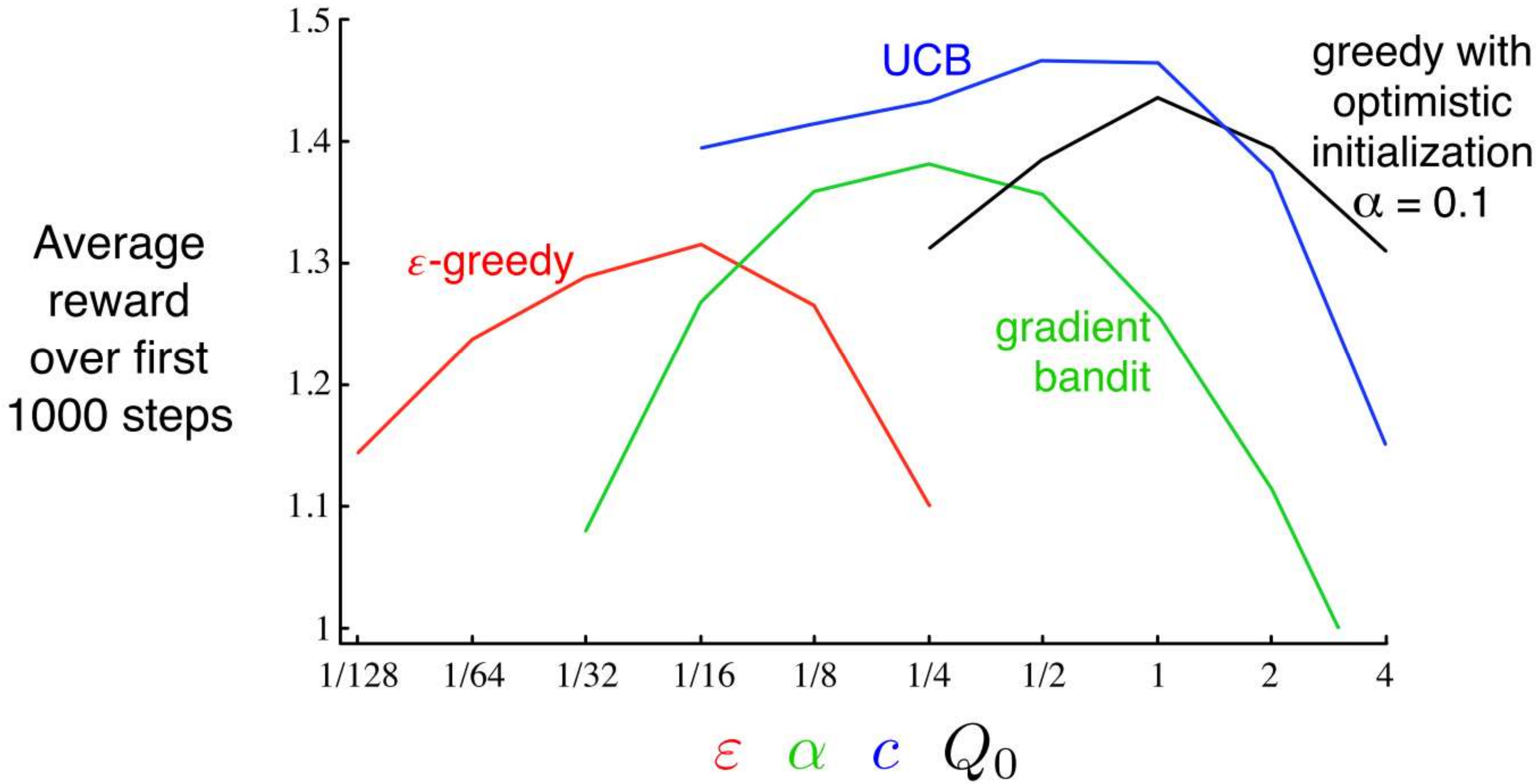
- We learn preferences according to:

$$\begin{aligned} H_{t+1}(A_t) &\doteq H_t(A_t) + \alpha\big(R_t - \bar{R}_t\big)\big(1 - \pi_t(A_t)\big), && \text{and} \\ H_{t+1}(a) &\doteq H_t(a) - \alpha\big(R_t - \bar{R}_t\big)\pi_t(a), && \text{for all } a \neq A_t, \end{aligned}$$



100%
80%
60%
40%
20%
0%
%
Optimal
action
α = 0.1
with baseline
α = 0.4
α = 0.1
without baseline
α = 0.4
1
250
500
750
1000
Steps

Results


Average reward over first 1000 steps
1.5
1.4
1.3
1.2
1.1
1
UCB
greedy with optimistic initialization $\alpha = 0.1$
ε-greedy
gradient bandit
1/128
1/64
1/32
1/16
1/8
1/4
1/2
1
2
4
ε α c Q_0

Summary 1/2

- *Reinforcement learning*: an agent learns through interactions with an environment.

- *k*-armed bandit problem: The true distribution of rewards is not known a priori, but must be learned.

$$q_*(a) \doteq \mathbb{E}[R_t \mid A_t = a]$$

$$Q_t(a) \approx q_*(a), \quad \forall a$$

Summary 2/2

We discussed:

- UniformBandit
- Exploration vs. exploitation tradeoff
- ε-Greedy
- How to reduce the bias of initial estimates $Q_1(a)$, namely a fixed step-size α.
- Optimistic initialization ($Q_1(a) = 5$)
- Upper confidence bound: $A_t \doteq \arg\max_a \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$
- Gradient bandit algorithm: preferences are used instead of estimates of q_*