What we learned last time

1. *Intelligence is the computational part of the ability to achieve goals*
    - looking deeper: 1) its a continuum, 2) its an appearance, 3) it varies with observer and purpose

2. We will (probably) figure out how to make intelligent systems in our lifetimes; it will change everything

3. But prior to that it will probably change our careers
    - as companies gear up to take advantage of the economic opportunities

4. This course has a demanding workload

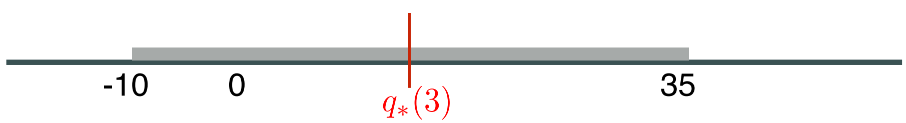
Multi-arm Bandits

Sutton and Barto, Chapter 2

The simplest reinforcement learning problem

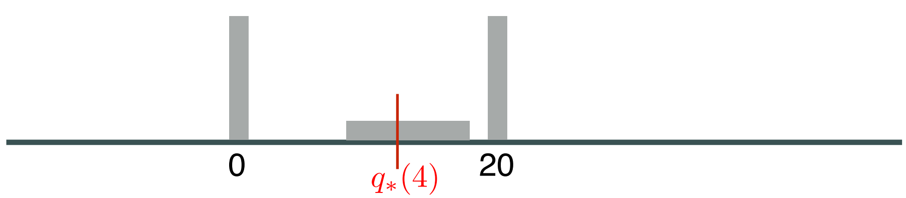
You are the algorithm! (bandit1)

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



$q_*(3) =$

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



$q_*(4) =$

The *k*-armed Bandit Problem

- On each of an infinite 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 indentically, 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)

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, but you need to do both

- You can never stop exploring, but maybe you should explore less with time. Or maybe not.

Action-Value Methods

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

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

- 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

A simple bandit algorithm

Initialize, for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Repeat forever:

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

$R \leftarrow bandit(A)$

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

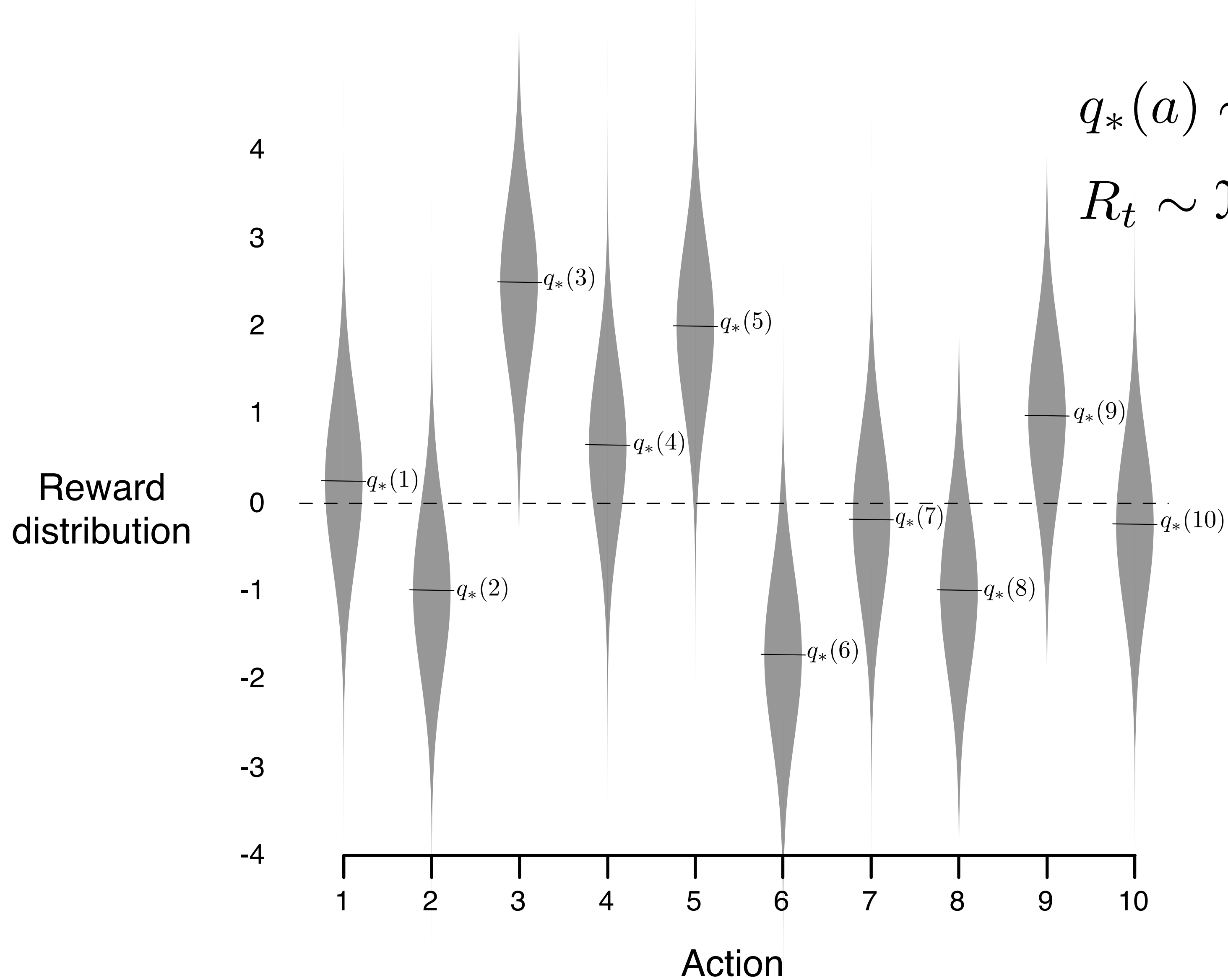
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\big[R - Q(A)\big]$

What we learned last time

1. *Multi-armed bandits* are a simplification of the real problem
    1. they have action and reward (a goal), but no input or sequentiality
2. A fundamental *exploitation-exploration tradeoff* arises in bandits
    1. *ε-greedy action selection* is the simplest way of trading off
3. *Learning action values* is a key part of solution methods
4. *The 10-armed testbed* illustrates all

One Bandit Task from

The 10-armed Testbed



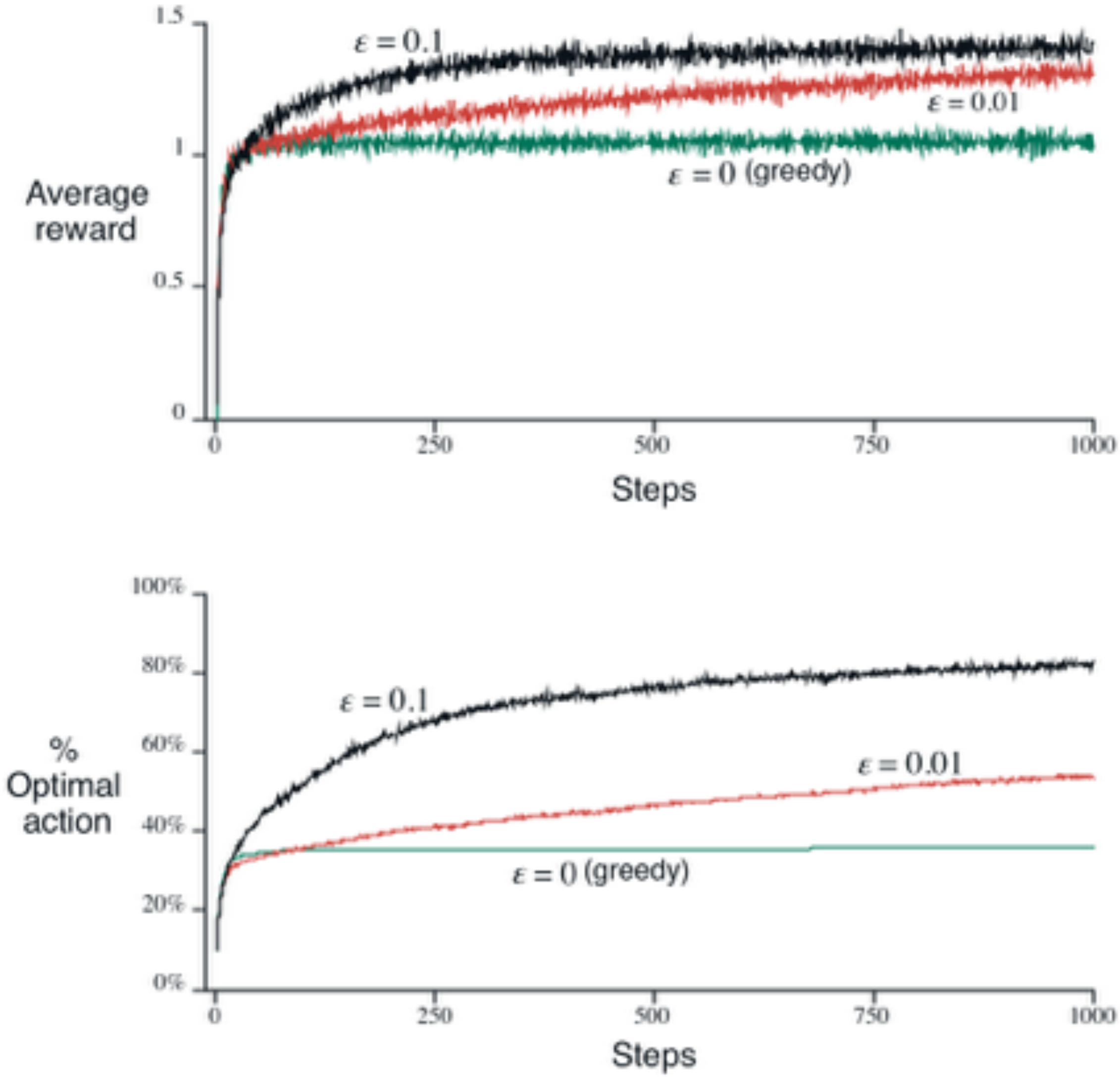
$$q_*(a) \sim \mathcal{N}(0, 1)$$

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

Run for 1000 steps

Repeat the whole thing 2000 times with different bandit tasks

ε-Greedy Methods on the 10-Armed Testbed



What we learned last time

1. *Multi-armed bandits* are a simplification of the real problem
    1. they have action and reward (a goal), but no input or sequentiality
2. *The exploitation-exploration tradeoff* arises in bandits
    1. *ε-greedy action selection* is the simplest way of trading off
3. *Learning action values* is a key part of solution methods
4. *The 10-armed testbed* illustrates all
5. Learning as averaging – a *fundamental learning rule*

Averaging $\longrightarrow$ learning rule

- 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:

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

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

$$NewEstimate \leftarrow OldEstimate \; + \; StepSize \Big[Target - OldEstimate\Big]$$

Derivation of incremental update

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

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

Averaging ⟶ learning rule

- 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:

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

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

$$NewEstimate \leftarrow OldEstimate \; + \; StepSize \Big[Target - OldEstimate\Big]$$

Tracking a Non-stationary Problem

- 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?)
- Better is an "exponential, recency-weighted average":

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

where α is a constant, *step-size parameter*, $0 < \alpha \leq 1$

- There is bias due to Q_1 that becomes smaller over time

Standard stochastic approximation convergence conditions

- To assure convergence with probability 1:

$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty \qquad \text{and} \qquad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$$

- e.g., $\alpha_n = \frac{1}{n}$
- not $\alpha_n = \frac{1}{n^2}$

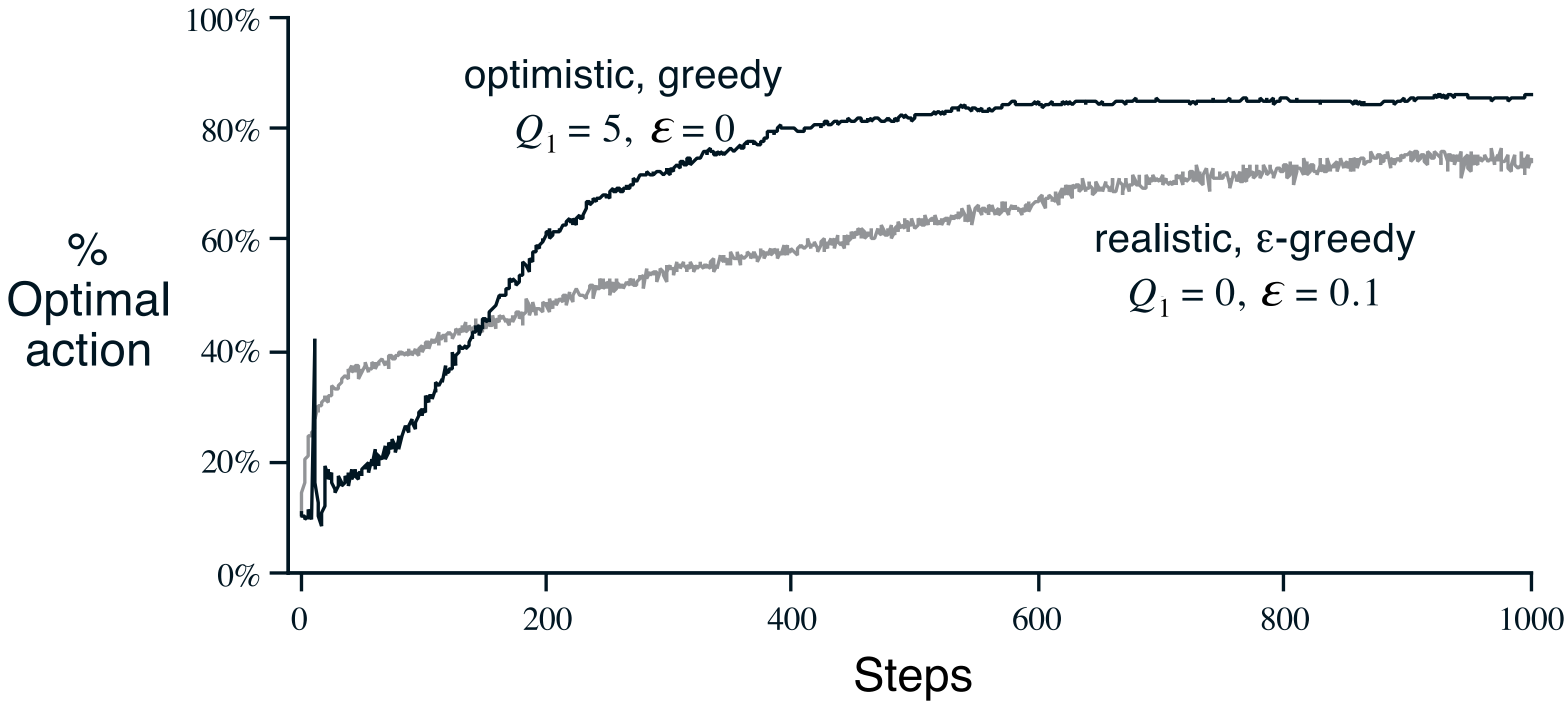
if $\alpha_n = n^{-p}, \quad p \in (0,1)$
then convergence is at the optimal rate:

$$O(1/\sqrt{n})$$

Optimistic Initial Values

- All methods so far depend on $Q_1(a)$, i.e., they are biased. So far we have used $Q_1(a) = 0$

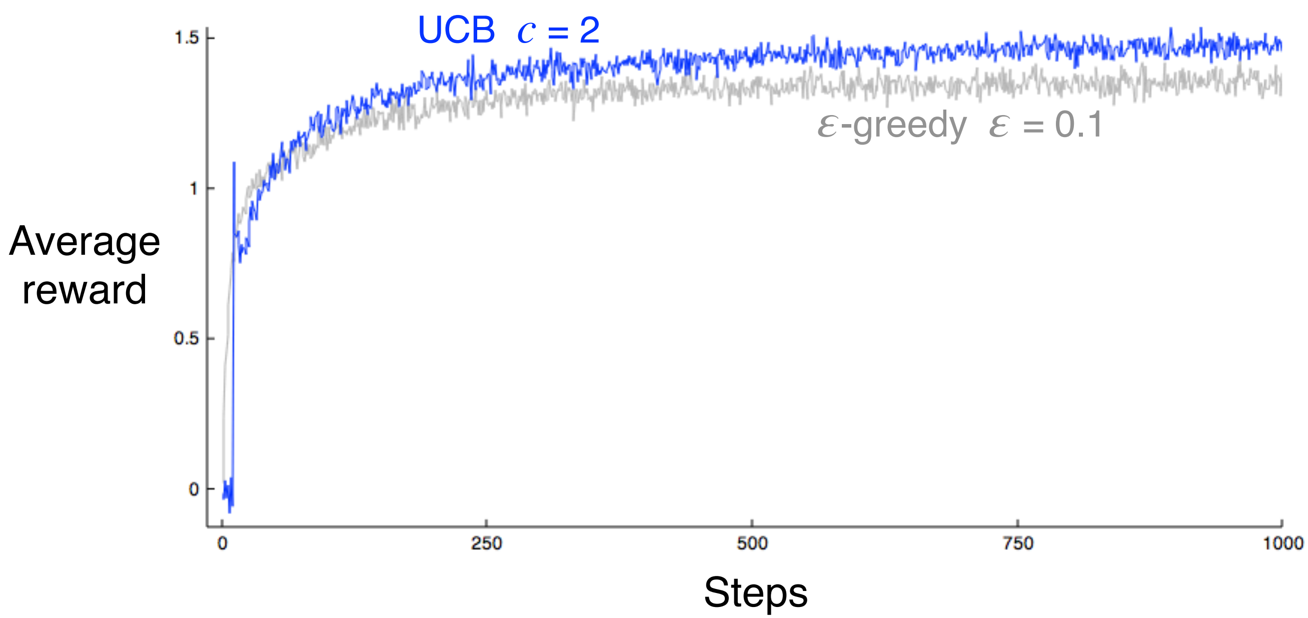
- Suppose we initialize the action values *optimistically* ($Q_1(a) = 5$), e.g., on the 10-armed testbed (with $\alpha = 0.1$)



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

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



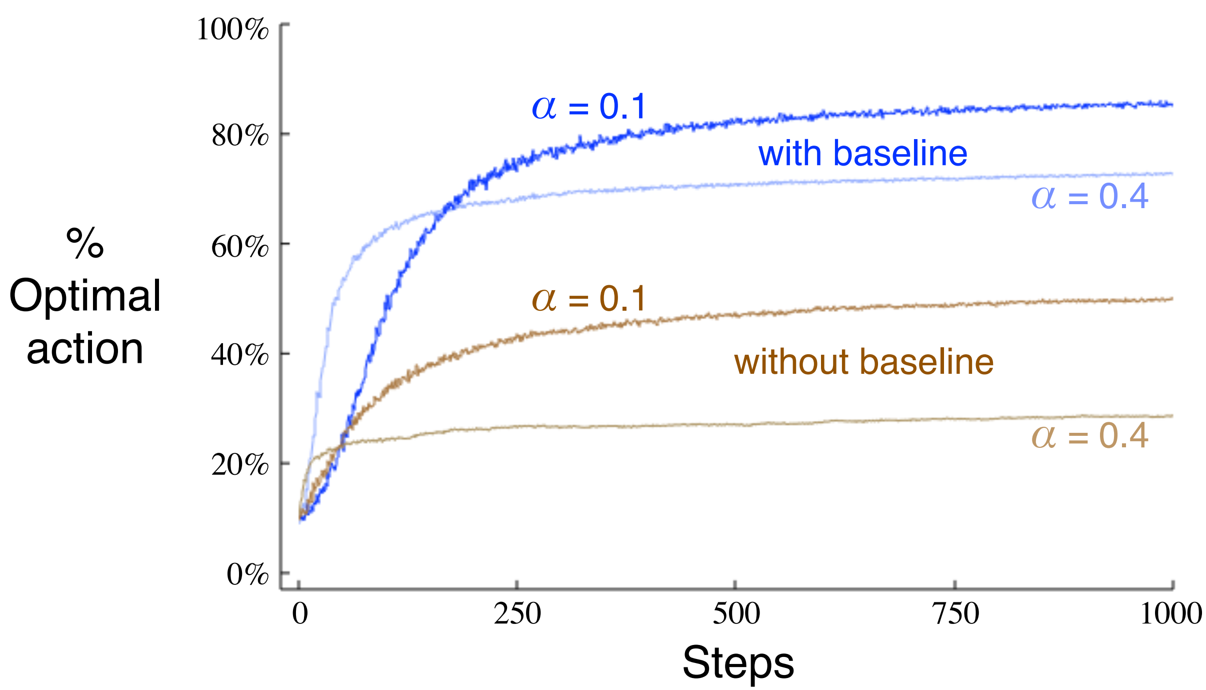
Gradient-Bandit Algorithms

- Let $H_t(a)$ be a learned *preference* for taking action a

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

$$H_{t+1}(a) \doteq H_t(a) + \alpha\Big(R_t - \bar{R}_t\Big)\Big(\mathbb{1}\{A_t = a\} - \pi_t(a)\Big), \qquad \forall a$$

$$\bar{R}_t \doteq \frac{1}{t}\sum_{i=1}^{t} R_i$$



Derivation of gradient-bandit algorithm

In exact *gradient ascent*:

$$H_{t+1}(a) \doteq H_t(a) + \alpha \frac{\partial \, \mathbb{E}\left[R_t\right]}{\partial H_t(a)}, \qquad (1)$$

where:

$$\mathbb{E}[R_t] \doteq \sum_b \pi_t(b) q_*(b),$$

$$\begin{aligned}
\frac{\partial \, \mathbb{E}[R_t]}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)} \left[\sum_b \pi_t(b) q_*(b) \right] \\
&= \sum_b q_*(b) \frac{\partial \, \pi_t(b)}{\partial H_t(a)} \\
&= \sum_b \left(q_*(b) - X_t \right) \frac{\partial \, \pi_t(b)}{\partial H_t(a)},
\end{aligned}$$

where X_t does not depend on b, because $\sum_b \frac{\partial \, \pi_t(b)}{\partial H_t(a)} = 0$.

$$
\begin{aligned}
\frac{\partial\, \mathbb{E}[R_t]}{\partial H_t(a)} &= \sum_b \big(q_*(b) - X_t\big) \frac{\partial\, \pi_t(b)}{\partial H_t(a)} \\
&= \sum_b \pi_t(b) \big(q_*(b) - X_t\big) \frac{\partial\, \pi_t(b)}{\partial H_t(a)} \Big/ \pi_t(b) \\
&= \mathbb{E}\left[\big(q_*(A_t) - X_t\big) \frac{\partial\, \pi_t(A_t)}{\partial H_t(a)} \Big/ \pi_t(A_t) \right] \\
&= \mathbb{E}\left[\big(R_t - \bar{R}_t\big) \frac{\partial\, \pi_t(A_t)}{\partial H_t(a)} \Big/ \pi_t(A_t) \right],
\end{aligned}
$$

where here we have chosen $X_t = \bar{R}_t$ and substituted R_t for $q_*(A_t)$, which is permitted because $\mathbb{E}[R_t | A_t] = q_*(A_t)$.

For now assume: $\frac{\partial\, \pi_t(b)}{\partial H_t(a)} = \pi_t(b) \big(\mathbf{1}_{a=b} - \pi_t(a)\big)$. Then:

$$
\begin{aligned}
&= \mathbb{E}\big[\big(R_t - \bar{R}_t\big) \pi_t(A_t) \big(\mathbf{1}_{a=A_t} - \pi_t(a)\big) / \pi_t(A_t) \big] \\
&= \mathbb{E}\big[\big(R_t - \bar{R}_t\big) \big(\mathbf{1}_{a=A_t} - \pi_t(a)\big) \big].
\end{aligned}
$$

$$
H_{t+1}(a) = H_t(a) + \alpha \big(R_t - \bar{R}_t\big) \big(\mathbf{1}_{a=A_t} - \pi_t(a)\big), \text{ (from (1), QED)}
$$

Thus it remains only to show that

$$\frac{\partial\, \pi_t(b)}{\partial H_t(a)} = \pi_t(b)\big(\mathbf{1}_{a=b} - \pi_t(a)\big).$$

Recall the standard quotient rule for derivatives:

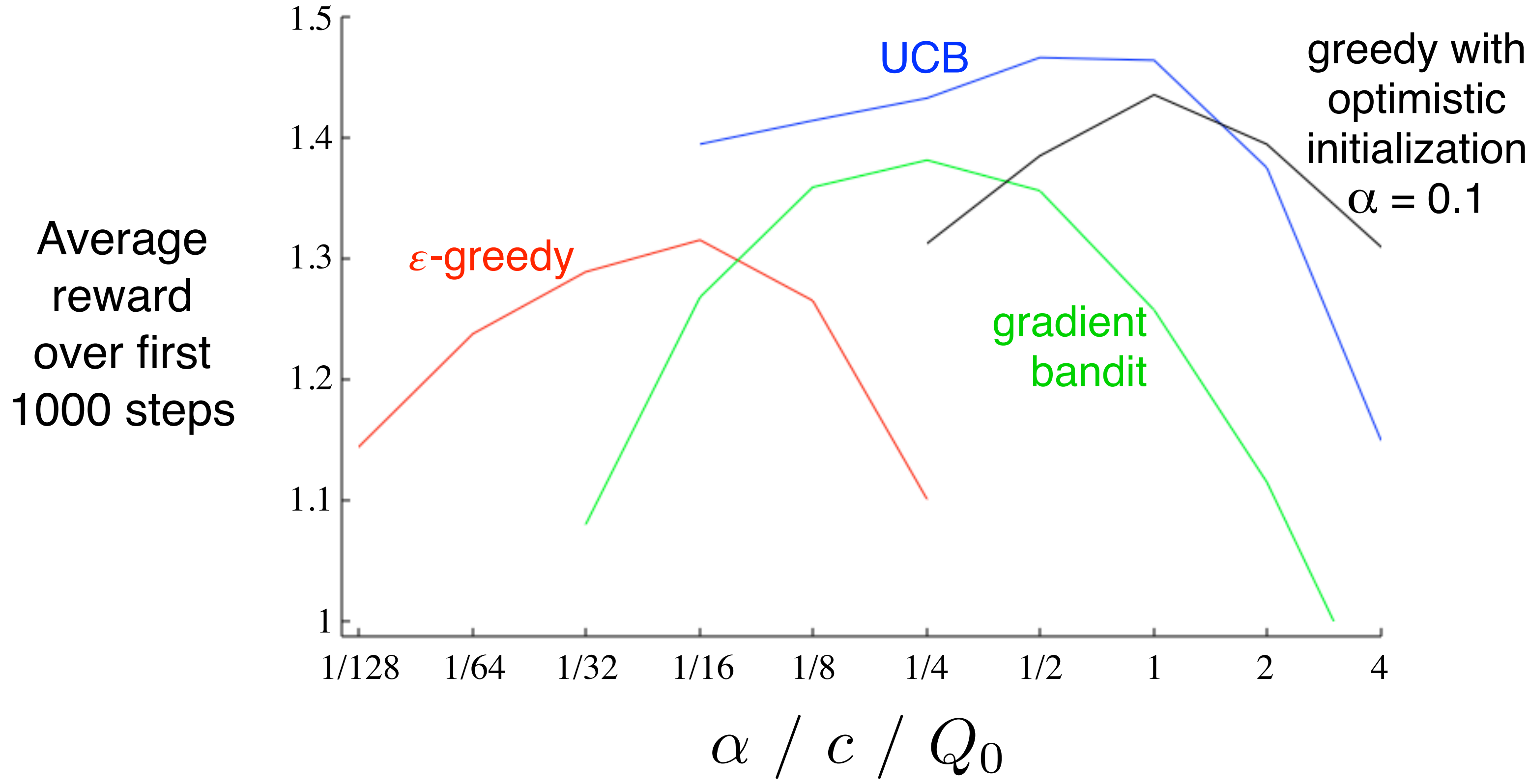
$$\frac{\partial}{\partial x}\left[\frac{f(x)}{g(x)}\right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x)\frac{\partial g(x)}{\partial x}}{g(x)^2}.$$

Using this, we can write...

Quotient Rule: $\frac{\partial}{\partial x}\left[\frac{f(x)}{g(x)}\right] = \frac{\frac{\partial f(x)}{\partial x} g(x) - f(x)\frac{\partial g(x)}{\partial x}}{g(x)^2}$

$$\begin{aligned}
\frac{\partial \pi_t(b)}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)} \pi_t(b) \\
&= \frac{\partial}{\partial H_t(a)} \left[\frac{e^{h_t(b)}}{\sum_{c=1}^{k} e^{h_t(c)}}\right] \\
&= \frac{\frac{\partial e^{h_t(b)}}{\partial H_t(a)} \sum_{c=1}^{k} e^{h_t(c)} - e^{h_t(b)} \frac{\partial \sum_{c=1}^{k} e^{h_t(c)}}{\partial H_t(a)}}{\left(\sum_{c=1}^{k} e^{h_t(c)}\right)^2} && \text{(Q.R.)} \\
&= \frac{\mathbf{1}_{a=b} e^{h_t(a)} \sum_{c=1}^{k} e^{h_t(c)} - e^{h_t(b)} e^{h_t(a)}}{\left(\sum_{c=1}^{k} e^{h_t(c)}\right)^2} && \left(\frac{\partial e^x}{\partial x} = e^x\right) \\
&= \frac{\mathbf{1}_{a=b} e^{h_t(b)}}{\sum_{c=1}^{k} e^{h_t(c)}} - \frac{e^{h_t(b)} e^{h_t(a)}}{\left(\sum_{c=1}^{k} e^{h_t(c)}\right)^2} \\
&= \mathbf{1}_{a=b} \pi_t(b) - \pi_t(b) \pi_t(a) \\
&= \pi_t(b)\big(\mathbf{1}_{a=b} - \pi_t(a)\big). && \text{(Q.E.D.)}
\end{aligned}$$

Summary Comparison of Bandit Algorithms



Conclusions

- These are all simple methods
  - but they are complicated enough—we will build on them
  - we should understand them completely
  - there are still open questions
- Our first algorithms that learn from evaluative feedback
  - and thus must balance exploration and exploitation
- Our first algorithms that appear to have a goal
  —that learn to maximize reward by trial and error

Our first dimensions!

- Problems vs Solution Methods
- Evaluative vs Instructive
- Associative vs Non-associative

Bandits?

Problem or Solution?

Problem space

	Single State	Associative
Instructive feedback		
Evaluative feedback		

Problem space

	Single State	Associative
Instructive feedback		
Evaluative feedback	Bandits (Function optimization)	

Problem space

	Single State	Associative
Instructive feedback		Supervised learning
Evaluative feedback	Bandits (Function optimization)	

Problem space

	Single State	Associative
Instructive feedback	Averaging	Supervised learning
Evaluative feedback	Bandits (Function optimization)	

Problem space

	Single State	Associative
Instructive feedback	Averaging	Supervised learning
Evaluative feedback	Bandits (Function optimization)	Associative Search (Contextual bandits)