for Multimedia

Course 2: Multi-Armed Bandits

M. Kieffer[1]

[1]Laboratoire des Signaux et Systèmes
Univ Paris-Saclay - CNRS - CentraleSupelec

Multimedia Networking, 2020

Outline

Outline

Introduction

Reinforcement learning

- Uses training information that evaluates the actions taken rather than instructs by giving correct actions.
- Need for active exploration, for explicit search for good behavior.

Purely evaluative feedback

- indicates how good the action taken was,
- not whether it was the best or the worst action possible.

Purely instructive feedback (supervised learning)

- indicates the correct action to take,
- independently of the action actually taken.

Introduction

Reinforcement learning

- Uses training information that evaluates the actions taken rather than instructs by giving correct actions.
- Need for active exploration, for explicit search for good behavior.

Purely evaluative feedback

- indicates how good the action taken was,
- not whether it was the best or the worst action possible.

Purely instructive feedback (supervised learning)

- indicates the correct action to take,
- independently of the action actually taken.

Introduction

Reinforcement learning

- Uses training information that evaluates the actions taken rather than instructs by giving correct actions.
- Need for active exploration, for explicit search for good behavior.

Purely evaluative feedback

- indicates how good the action taken was,
- not whether it was the best or the worst action possible.

Purely instructive feedback (supervised learning)

- indicates the correct action to take,
- independently of the action actually taken.

Introduction

This course presents the evaluative aspect of reinforcement learning

- in a simplified setting,
- involving learning to act in a simple situation,
- avoiding complexity of full reinforcement learning problems.

Reinforcement learning for the k-armed bandit problem

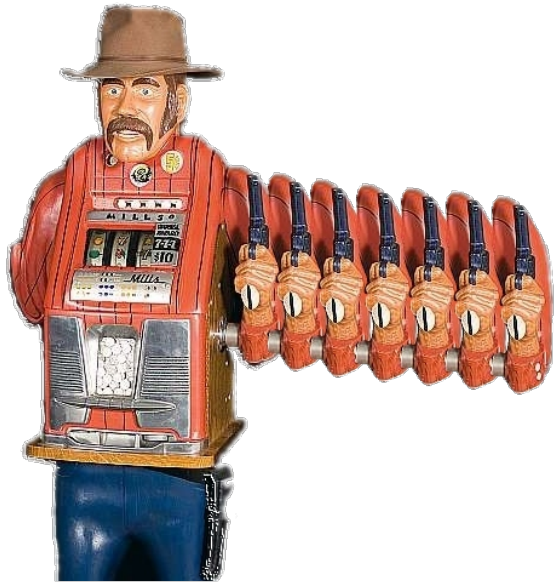
Outline

Multi-armed bandit

Consider the learning problem:

- You have to choose among k different options, or actions.
- After the choice, a numerical reward is obtained, chosen from a stationary pdf that depends on the selected action

Objective: maximize expected total reward over some time period (1000 action selections, or time steps).



Multi-armed bandit

In k-armed bandit problem

- A_t: action selected at time t (random variable)
- R_t: corresponding reward (random variable)

Each of the k actions has an *expected* or *mean reward*: *value* of that action

- Value of action a is

$$q^*(a) = \mathbb{E}(R_t \mid A_t = a)$$

If $q^*(a)$ would be known, k-armed bandit problem would be solved:

select the action with largest value.

In practice

- action values are not known with certainty,
- estimates may be available: $Q_t(a)$ estimate of $q^*(a)$ at time t.
- $Q_t(a)$ should be as close as possible to $q^*(a)$.

Multi-armed bandit

In k-armed bandit problem

- A_t: action selected at time t (random variable)
- R_t: corresponding reward (random variable)

Each of the k actions has an *expected* or *mean reward*: *value* of that action

- Value of action a is

$$q^*(a) = \mathbb{E}(R_t \mid A_t = a)$$

If $q^*(a)$ would be known, k-armed bandit problem would be solved:

select the action with largest value.

In practice

- action values are not known with certainty,
- estimates may be available: $Q_t(a)$ estimate of $q^*(a)$ at time t.
- $Q_t(a)$ should be as close as possible to $q^*(a)$.

Multi-armed bandit

In k-armed bandit problem

- A_t: action selected at time t (random variable)
- R_t: corresponding reward (random variable)

Each of the k actions has an *expected* or *mean reward*: *value* of that action

- Value of action a is

$$q^*(a) = \mathbb{E}(R_t \mid A_t = a)$$

If $q^*(a)$ would be known, k-armed bandit problem would be solved:

select the action with largest value.

In practice

- action values are not known with certainty,
- estimates may be available: $Q_t(a)$ estimate of $q^*(a)$ at time t.
- $Q_t(a)$ should be as close as possible to $q^*(a)$.

Exploration vs exploitation

Assume that at time t, $Q_t(a)$ is available

- The actions
$$a_t^* \in \arg\max_a Q_t(a)$$
are called the *greedy* actions: *exploitation* of current knowledge
- Selecting
$$a_t \notin \arg\max_a Q_t(a)$$
corresponds to *exploration*

Exploitation: maximizes expected reward on the *one step*,

Exploration: may produce the greater total reward in the *long run*.

Exploration vs exploitation

Assume that at time t, $Q_t(a)$ is available

- The actions
$$a_t^* \in \arg\max_a Q_t(a)$$
are called the *greedy* actions: *exploitation* of current knowledge
- Selecting
$$a_t \notin \arg\max_a Q_t(a)$$
corresponds to *exploration*

Exploitation: maximizes expected reward on the *one step*,

Exploration: may produce the greater total reward in the *long run*.

Exploration vs exploitation

Whether it is better to explore or exploit depends on

- precise values of the estimates,
- uncertainties,
- number of remaining steps.

Many sophisticated methods for balancing exploration and exploitation for particular mathematical formulations.
Rely on strong assumptions about

- stationarity
- prior knowledge

either violated or impossible to verify in applications and in full reinforcement learning problem

Outline

Action-value methods

Focus first on methods for

- estimating the values of actions
- using the estimates to make action selection decisions

these methods are *action-value* methods.

True value of an action is the *mean reward* when that action is selected

$\Downarrow$

Estimate by *averaging* the rewards actually received.

Action-value methods

Focus first on methods for

- estimating the values of actions
- using the estimates to make action selection decisions

these methods are *action-value* methods.

True value of an action is the *mean reward* when that action is selected

$$\Downarrow$$

Estimate by *averaging* the rewards actually received.

Action-value methods

Sample average

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \mathbb{I}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{I}_{A_i=a}}$$

where $\mathbb{I}_{predicate} = 1$ if $predicate = true$ and $\mathbb{I}_{predicate} = 0$ if $predicate = false$.

When $\sum_{i=1}^{t-1} \mathbb{I}_{A_t=a} = 0$, $Q_t(a)$ may take any arbitrary value, *e.g.*, $Q_t(a) = 0$.

By the law of large numbers, when $\sum_{i=1}^{t-1} \mathbb{I}_{A_t=a} \to \infty$,then $Q_t(a) \to q^*(a)$.

Action-value methods

Sample average

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \mathbb{I}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{I}_{A_i=a}}$$

where $\mathbb{I}_{predicate} = 1$ if $predicate = true$ and $\mathbb{I}_{predicate} = 0$ if $predicate = false$.

When $\sum_{i=1}^{t-1} \mathbb{I}_{A_t=a} = 0$, $Q_t(a)$ may take any arbitrary value, *e.g.*, $Q_t(a) = 0$.

By the law of large numbers, when $\sum_{i=1}^{t-1} \mathbb{I}_{A_t=a} \to \infty$,then $Q_t(a) \to q^*(a)$.

Action-value methods

Simplest action selection rule: greedy action selection

$$A_t = \arg\max_a Q_t(a)$$

with ties broken arbitrarily.

Better action selection rule: ε-greedy action selection

- greedy action selection with a probability $1 - \varepsilon$
- other action selection with a probability ε

With ε-greedy methods, estimates $Q_t(a)$ converge to $q^*(a)$ for all a as $t \to \infty$, since all actions are selected an infinite amount of times.

Action-value methods

Simplest action selection rule: greedy action selection

$$A_t = \arg\max_a Q_t(a)$$

with ties broken arbitrarily.

Better action selection rule: ε-greedy action selection

- greedy action selection with a probability $1 - \varepsilon$
- other action selection with a probability ε

With ε-greedy methods, estimates $Q_t(a)$ converge to $q^*(a)$ for all a as $t \to \infty$, since all actions are selected an infinite amount of times.

Action-value methods

Simplest action selection rule: greedy action selection

$$A_t = \arg\max_a Q_t(a)$$

with ties broken arbitrarily.

Better action selection rule: ε-greedy action selection

- greedy action selection with a probability $1 - \varepsilon$
- other action selection with a probability ε

With ε-greedy methods, estimates $Q_t(a)$ converge to $q^*(a)$ for all a as $t \to \infty$, since all actions are selected an infinite amount of times.

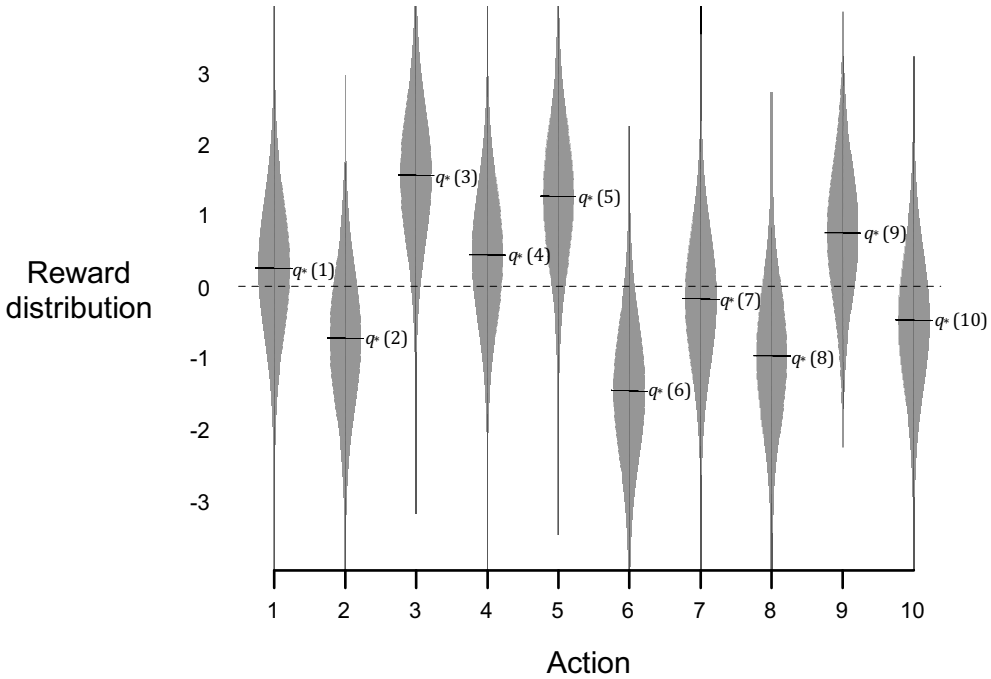
Outline

10-armed bandit testbed

Test problems used to assess performance of ε-greedy algorithm

10-armed bandit testbed:

- 2000 randomly generated k-armed bandit problems with $k = 10$.
- For each bandit problem, $q^*(a) \sim \mathcal{N}(m(a), 1)$, $a = 1, \ldots, 10$,
- with $m(a) \sim \mathcal{N}(0, 1)$, $a = 1, \ldots, 10$.

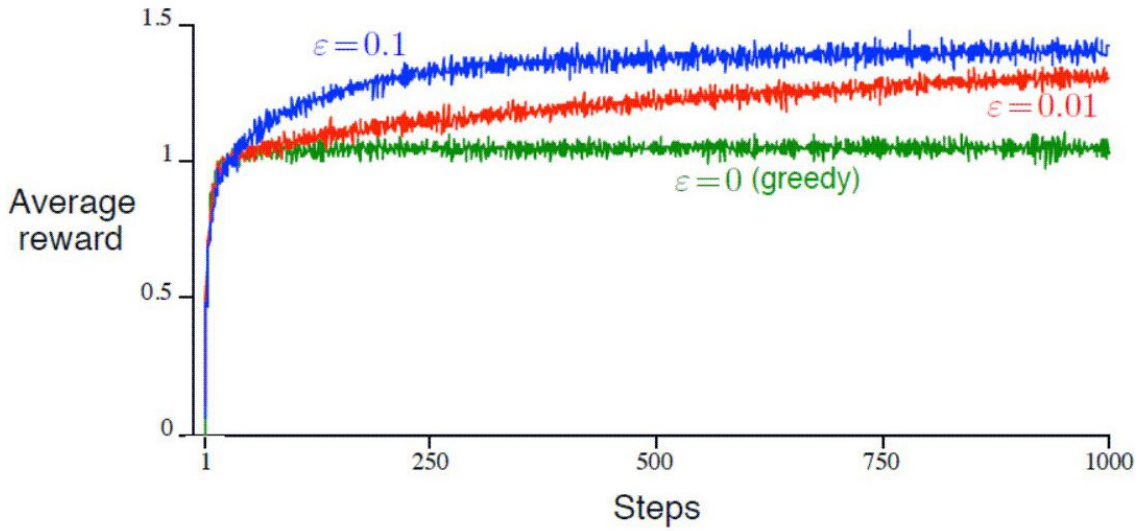


10-armed bandit testbed

For any learning method

- One run: behavior evaluated with experience over 1000 time steps of one bandit problem.
- Averaging for 2000 independent runs, each with a different bandit problem

Provides measures of the learning algorithm's average behavior.

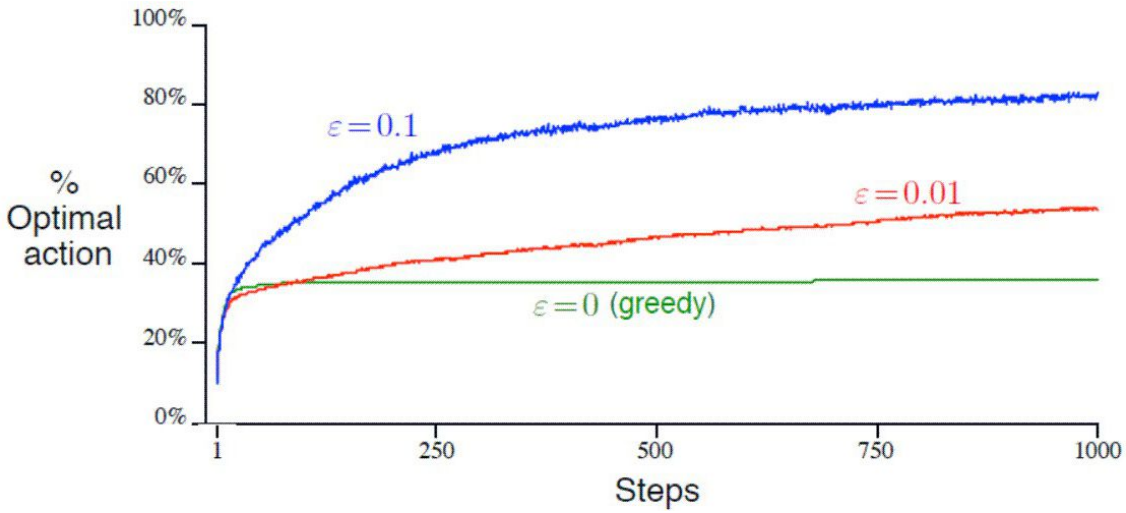


10-armed bandit testbed

For any learning method

- One run: behavior evaluated with experience over 1000 time steps of one bandit problem.
- Averaging for 2000 independent runs, each with a different bandit problem

Provides measures of the learning algorithm's average behavior.



10-armed bandit testbed

The advantage of ε-greedy over greedy methods depends on the task.

- With noisier rewards, more exploration required to find optimal action, ε-greedy methods should be even better relative to greedy method.
- With zero-variance rewards, greedy method would know true value of each action after trying it once.

Even in the deterministic case there is a large advantage to exploring, *e.g.*, in case the bandit task were nonstationary (true values of the actions changed over time).

Reinforcement learning requires a balance between *exploration* and *exploitation*.

10-armed bandit testbed

The advantage of ε-greedy over greedy methods depends on the task.

- With noisier rewards, more exploration required to find optimal action, ε-greedy methods should be even better relative to greedy method.
- With zero-variance rewards, greedy method would know true value of each action after trying it once.

Even in the deterministic case there is a large advantage to exploring, *e.g.*, in case the bandit task were nonstationary (true values of the actions changed over time).

Reinforcement learning requires a balance between *exploration* and *exploitation*.

Outline

Incremental implementation

The evaluation of

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \mathbb{I}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{I}_{A_i=a}}$$

requires

- to store all received rewards
- evaluations with increasing complexity.

Nevertheless $Q_t(a)$ can be evaluated iteratively.

Incremental implementation

The evaluation of

$$Q_t(a) = \frac{\sum_{i=1}^{t-1} R_i \mathbb{I}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{I}_{A_i=a}}$$

requires

- to store all received rewards
- evaluations with increasing complexity.

Nevertheless $Q_t(a)$ can be evaluated iteratively.

Incremental implementation

Focus on a single action

$$
\begin{aligned}
Q_{t+1} &= \frac{1}{t} \sum_{i=1}^{t} R_i \\
&= \frac{1}{t} \left(R_t + \sum_{i=1}^{t-1} R_i \right) \\
&= \frac{1}{t} \left(R_t + (t-1) Q_t \right) \\
&= Q_t + \frac{1}{t} \left(R_t - Q_t \right),
\end{aligned}
$$

which holds also at $t = 1$, $Q_2 = R_1$ whatever the value of Q_1.

Requires little memory and computations at each step.

Incremental implementation

The update rule

$$Q_{t+1} = Q_t + \frac{1}{t}(R_t - Q_t)$$

is of the generic form

$$NewEstimate \leftarrow OldEstimate + StepSize\,(Target - OldEstimate)$$

where

- $(Target - OldEstimate)$ is the estimation error
- $StepSize$ is the step with which the estimate is corrected, here time-varying.

Incremental implementation

The update rule

$$Q_{t+1} = Q_t + \frac{1}{t}\left(R_t - Q_t\right)$$

is of the generic form

$$\mathit{NewEstimate} \leftarrow \mathit{OldEstimate} + \mathit{StepSize}\left(\mathit{Target} - \mathit{OldEstimate}\right)$$

where

- $\left(\mathit{Target} - \mathit{OldEstimate}\right)$ is the estimation error
- *StepSize* is the step with which the estimate is corrected, here time-varying.

StepSize denoted α or $\alpha_t\left(a\right)$ in what follows.

A simple bandit algorithm

Following algorithm summarizes complete bandit algorithm using incrementally computed sample averages and ε-greedy action selection

Simple bandit algorithm

Initialize: for $a = 1$ to k:

$Q(a) \leftarrow 0$

$N(a) \leftarrow 0$

Loop forever:

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

$R \leftarrow bandit(A)$

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

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

Outline

Non-stationary problems

Previous results well-suited to stationary bandit problems with stationary reward probabilities

To track varying rewards probabilities, one may use

$$Q_{t+1} = Q_t + \alpha (R_t - Q_t)$$

with step-size parameter $\alpha \in (0, 1]$.

Q_{t+1} may then be expressed as a function of α and past values of R_t as

$$\begin{aligned} Q_{t+1} &= Q_t + \alpha (R_t - Q_t) \\ &= \alpha R_t + (1 - \alpha) Q_t \\ &= \alpha R_t + (1 - \alpha) (Q_{t-1} + \alpha (R_{t-1} - Q_{t-1})) \\ &= \alpha R_t + \alpha (1 - \alpha) R_{t-1} + (1 - \alpha)^2 Q_{t-1} \\ &= (1 - \alpha)^t Q_1 + \sum_{i=1}^{t} \alpha (1 - \alpha)^{t-i} R_i \end{aligned}$$

Non-stationary problems

Previous results well-suited to stationary bandit problems with stationary reward probabilities

To track varying rewards probabilities, one may use

$$Q_{t+1} = Q_t + \alpha \left(R_t - Q_t \right)$$

with step-size parameter $\alpha \in (0, 1]$.

Q_{t+1} may then be expressed as a function of α and past values of R_t as

$$\begin{aligned} Q_{t+1} &= Q_t + \alpha \left(R_t - Q_t \right) \\ &= \alpha R_t + (1 - \alpha) \, Q_t \\ &= \alpha R_t + (1 - \alpha) \left(Q_{t-1} + \alpha \left(R_{t-1} - Q_{t-1} \right) \right) \\ &= \alpha R_t + \alpha (1 - \alpha) \, R_{t-1} + (1 - \alpha)^2 \, Q_{t-1} \\ &= (1 - \alpha)^t \, Q_1 + \sum_{i=1}^{t} \alpha (1 - \alpha)^{t-i} \, R_i \end{aligned}$$

Non-stationary problems

Then

$$Q_{t+1} = (1-\alpha)^t Q_1 + \sum_{i=1}^{t} \alpha (1-\alpha)^{t-i} R_i$$

is a weighted average of the preceding rewards, with exponentially decreasing weights

- When α is close to 0, the average is over a large number of rewards
- When α is close to 1, the average considers the very last rewards

Non-stationary problems

Step-size may be time varying $\alpha_n(a)$, step-size after the n-th selection of action a

In stationary case, convergence is guaranteed provided that

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

First condition ensures that $\alpha_n(a)$ is large enough to compensate bad initial guesses

Second condition ensures that $\alpha_n(a)$ is not too large.

Outline

Optimistic Initial Values

Previous methods are dependent to some extent on initial action-value estimates, $Q_1(a)$, see

$$Q_{t+1} = (1-\alpha)^t Q_1 + \sum_{i=1}^{t} \alpha (1-\alpha)^{t-i} R_i.$$

The methods are *biased* by their initial estimates:

- May be bad: must be picked by the user, *e.g.*, set them all to zero.
- May be good: provide an easy way introduce prior knowledge about level of rewards which can be expected.

Initial action values may be used to encourage exploration: Optimistic Initial Values.

Optimistic Initial Values

Previous methods are dependent to some extent on initial action-value estimates, $Q_1(a)$, see

$$Q_{t+1} = (1-\alpha)^t Q_1 + \sum_{i=1}^{t} \alpha (1-\alpha)^{t-i} R_i.$$

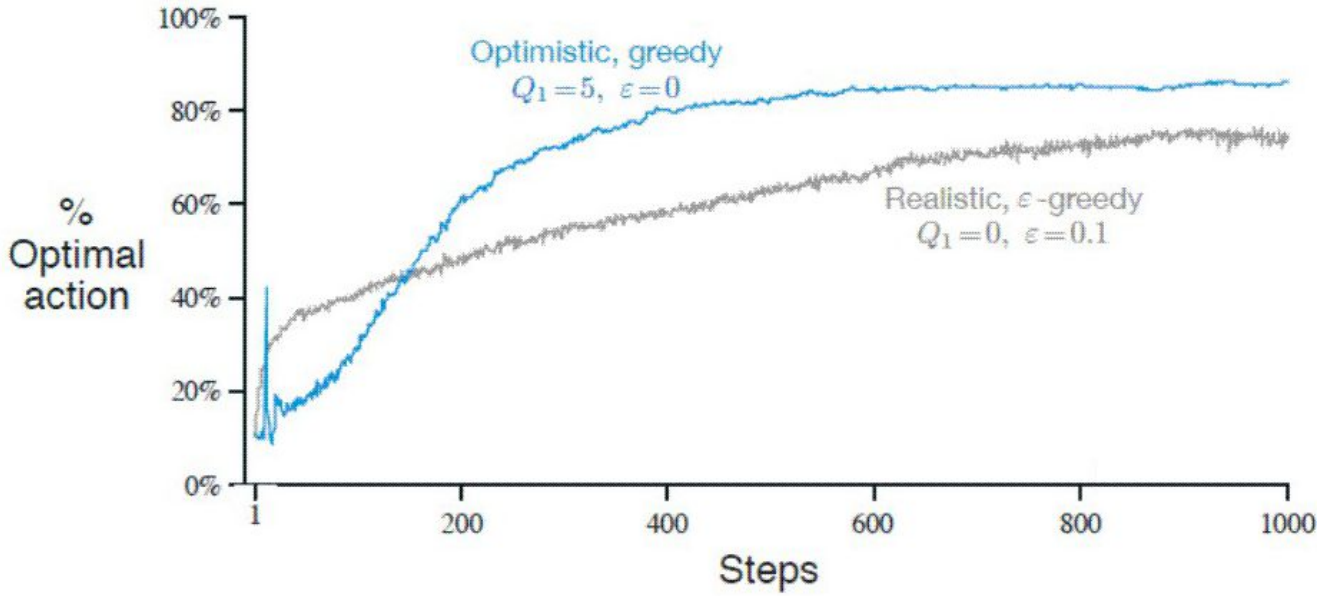
The methods are *biased* by their initial estimates:

- May be bad: must be picked by the user, *e.g.*, set them all to zero.
- May be good: provide an easy way introduce prior knowledge about level of rewards which can be expected.

Initial action values may be used to encourage exploration: Optimistic Initial Values.

Optimistic Initial Values

In 10-armed testbed: Mean values of $q^*(a)$ selected as $\mathcal{N}(0, 1)$.
Suppose that instead $Q_1(a) = 5$ for all a: optimistic initial value.



Encourages the initial explorations, even in greedy approach.

Outline

Upper-Confidence-Bound Action Selection

Exploration needed to reduce uncertainty about accuracy of action-value estimates.

Greedy actions are look best at present. Some of other actions may actually be better.

ε-greedy action selection forces the non-greedy actions to be tried, indiscriminately
No preference for

- nearly greedy
- particularly uncertain.

Ideally: select non-greedy actions according to their potential for actually being optimal. Take into account

- how close the estimates are to being maxima
- the uncertainties in those estimates.

Upper-Confidence-Bound Action Selection

Exploration needed to reduce uncertainty about accuracy of action-value estimates.

Greedy actions are look best at present. Some of other actions may actually be better.

ε-greedy action selection forces the non-greedy actions to be tried, indiscriminately
No preference for

- nearly greedy
- particularly uncertain.

Ideally: select non-greedy actions according to their potential for actually being optimal. Take into account

- how close the estimates are to being maxima
- the uncertainties in those estimates.

Upper-Confidence-Bound Action Selection

Exploration needed to reduce uncertainty about accuracy of action-value estimates.

Greedy actions are look best at present. Some of other actions may actually be better.

ε-greedy action selection forces the non-greedy actions to be tried, indiscriminately
No preference for

- nearly greedy
- particularly uncertain.

Ideally: select non-greedy actions according to their potential for actually being optimal. Take into account

- how close the estimates are to being maxima
- the uncertainties in those estimates.

Upper-Confidence-Bound Action Selection

Possible way of doing this is to select

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

where

- $N_t(a)$ is the number of times a was selected
- $\ln t$ increases with t
- c determines the *confidence level*

In initial steps, $N_t(a) = 0$ and a is in the set of actions to be chosen.

Selects actions

- which are greedy
- which are close to greedy
- which have not been selected for a long time
- for which $Q_t(a)$ is uncertain

Upper-Confidence-Bound Action Selection

Possible way of doing this is to select

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

where

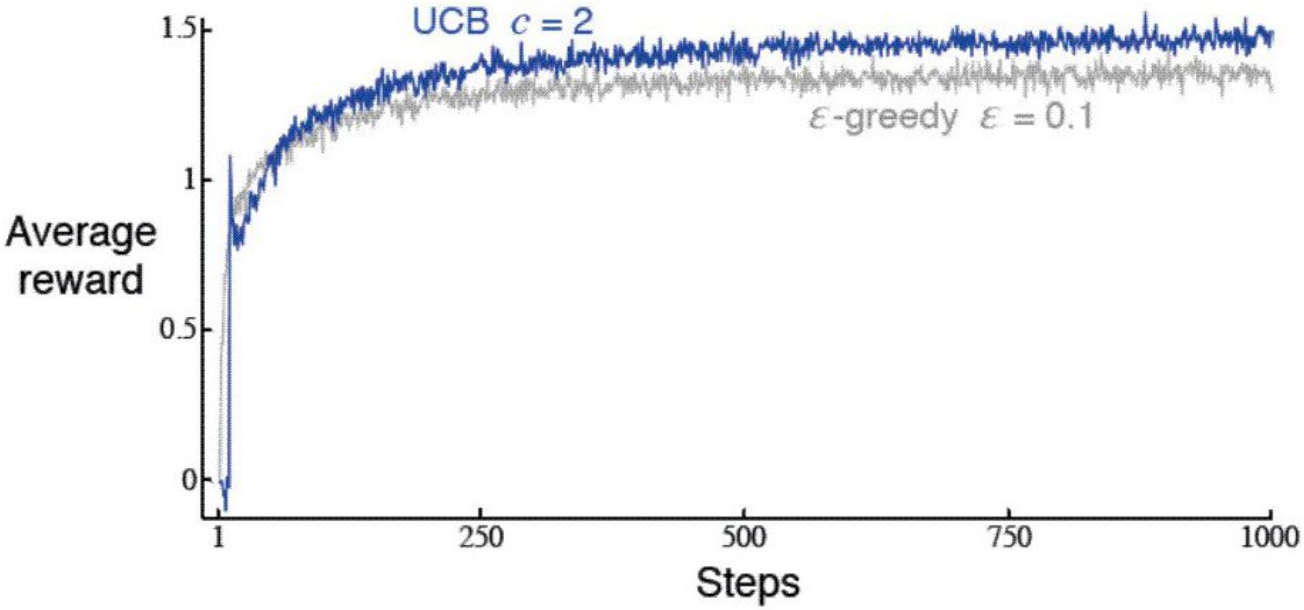
- $N_t(a)$ is the number of times a was selected
- $\ln t$ increases with t
- c determines the *confidence level*

In initial steps, $N_t(a) = 0$ and a is in the set of actions to be chosen.
Selects actions

- which are greedy
- which are close to greedy
- which have not been selected for a long time
- for which $Q_t(a)$ is uncertain

Upper-Confidence-Bound Action Selection

Illustration on 10-armed bandit problem



UCB:

- often performs well,
- is more difficult than ε-greedy to extend beyond bandits to the more general reinforcement learning settings
- difficulty when dealing with nonstationary problems.

Outline

Gradient Bandit Algorithm

Previous methods

- estimate action values
- use those estimates to select actions.

Here: consider learning a *numerical preference* $H_t(a)$ for each action a.

The larger the preference, the more often that action is taken.
The preference has no interpretation in terms of reward.
Only the relative preference of one action over another is important.

Gradient Bandit Algorithm

Previous methods

- estimate action values
- use those estimates to select actions.

Here: consider learning a *numerical preference* $H_t(a)$ for each action a.

The larger the preference, the more often that action is taken.
The preference has no interpretation in terms of reward.
Only the relative preference of one action over another is important.

Gradient Bandit Algorithm

Previous methods

- estimate action values
- use those estimates to select actions.

Here: consider learning a *numerical preference* $H_t(a)$ for each action a.

The larger the preference, the more often that action is taken.
The preference has no interpretation in terms of reward.
Only the relative preference of one action over another is important.

Gradient Bandit Algorithm

Probability of selection of an action evaluated according to soft-max distribution (Gibbs, Boltzmann)

$$\pi_t(a) = \Pr(A_t = a) = \frac{\exp(H_t(a))}{\sum_b \exp(H_t(b))}.$$

where initially

- all $H_t(a)$ are the same, *e.g.*, equal to 0.
- all $\pi_t(a)$ are equal.

Update of $H_t(a)$ is by stochastic gradient ascent.

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

$$H_{t+1}(a) = H_t(a) - \alpha\left(R_t - \overline{R}_t\right)\pi_t(a), \text{ for all } a \neq A_t.$$

where $\overline{R}_t$ is the average reward, whatever the chosen action.

Gradient Bandit Algorithm

Probability of selection of an action evaluated according to soft-max distribution (Gibbs, Boltzmann)

$$\pi_t(a) = \Pr(A_t = a) = \frac{\exp(H_t(a))}{\sum_b \exp(H_t(b))}.$$

where initially

- all $H_t(a)$ are the same, *e.g.*, equal to 0.
- all $\pi_t(a)$ are equal.

Update of $H_t(a)$ is by stochastic gradient ascent.

$$H_{t+1}(A_t) = H_t(A_t) + \alpha\left(R_t - \overline{R}_t\right)(1 - \pi_t(A_t)),$$
$$H_{t+1}(a) = H_t(a) - \alpha\left(R_t - \overline{R}_t\right)\pi_t(a), \text{ for all } a \neq A_t.$$

where $\overline{R}_t$ is the average reward, whatever the chosen action.

Gradient Bandit Algorithm

$$H_{t+1}(A_t) = H_t(A_t) + \alpha\left(R_t - \overline{R}_t\right)(1 - \pi_t(A_t)),$$
$$H_{t+1}(a) = H_t(a) - \alpha\left(R_t - \overline{R}_t\right)\pi_t(a), \text{ for all } a \neq A_t.$$

where $\overline{R}_t$ is the average reward, whatever the chosen action, with

$$\overline{R}_t = \frac{1}{t}\sum_{i=1}^{t} R_i$$
$$= \overline{R}_{t-1} + \frac{1}{t}\left(R_t - \overline{R}_{t-1}\right)$$

or

$$\overline{R}_t = \overline{R}_{t-1} + \beta\left(R_t - \overline{R}_{t-1}\right).$$

When the action provides a reward

- larger than average, preference is increased,
- smaller than average, preference is decreased.

Gradient Bandit Algorithm

$$H_{t+1}(A_t) = H_t(A_t) + \alpha\left(R_t - \overline{R}_t\right)\left(1 - \pi_t(A_t)\right),$$
$$H_{t+1}(a) = H_t(a) - \alpha\left(R_t - \overline{R}_t\right)\pi_t(a), \text{ for all } a \neq A_t.$$

where $\overline{R}_t$ is the average reward, whatever the chosen action, with

$$\begin{aligned}\overline{R}_t &= \frac{1}{t}\sum_{i=1}^{t} R_i \\ &= \overline{R}_{t-1} + \frac{1}{t}\left(R_t - \overline{R}_{t-1}\right)\end{aligned}$$

or

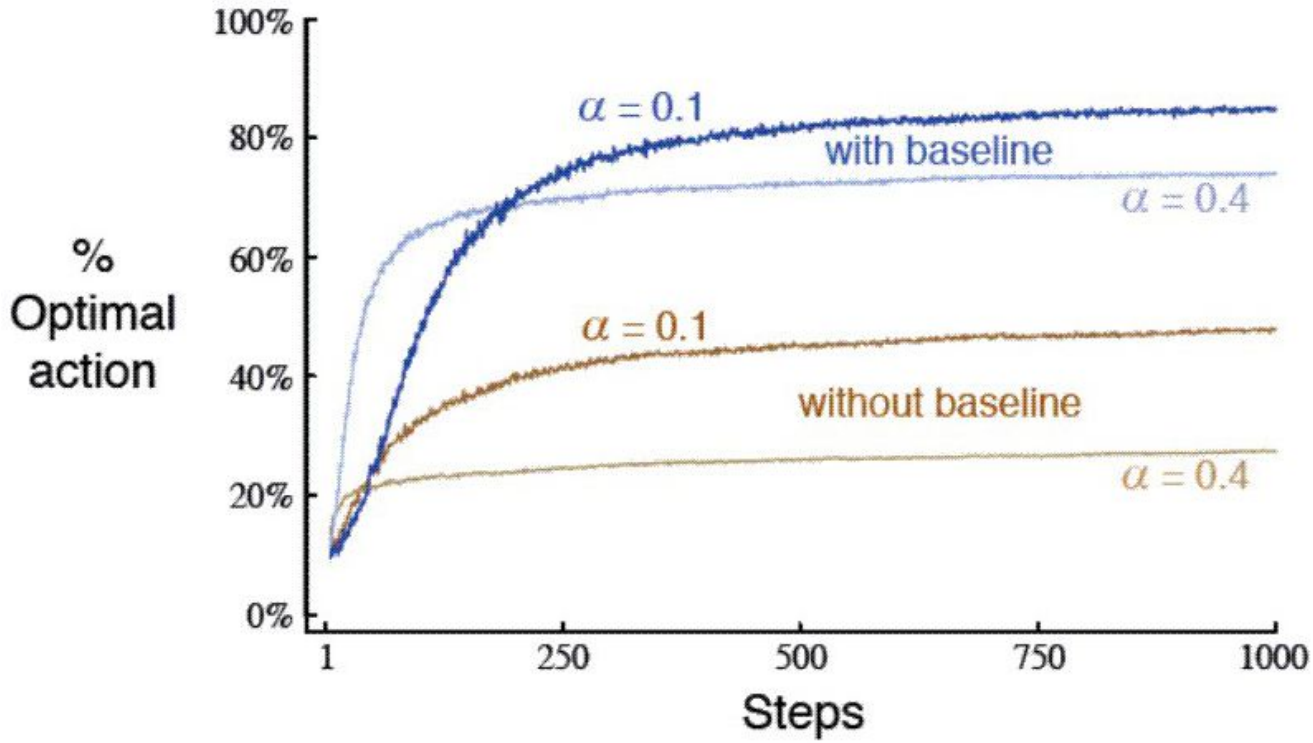
$$\overline{R}_t = \overline{R}_{t-1} + \beta\left(R_t - \overline{R}_{t-1}\right).$$

When the action provides a reward

- larger than average, preference is increased,
- smaller than average, preference is decreased.

Gradient Bandit Algorithm

Illustration on modified version of 10-armed bandit problem (true expected rewards are all shifted by $+4$)



Without shift (baseline), significant performance degradation.

Gradient Bandit Algorithm - interpretation

Consider measure of performance (to be maximized)

$$\mathbb{E}(R_t) = \sum_x \pi_t(x)\, q^*(x)$$

where the $\pi_t(x)$ are functions of the preferences.

Gradient ascent

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

Since $q^*(x)$ are not known, one considers *stochastic gradient ascent*

$$\frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} = \frac{\partial}{\partial H_t(a)} \left(\sum_x \pi_t(x)\, q^*(x) \right)$$

$$= \sum_x q^*(x) \frac{\partial \pi_t(x)}{\partial H_t(a)}$$

Gradient Bandit Algorithm - interpretation

Consider measure of performance (to be maximized)

$$\mathbb{E}(R_t) = \sum_x \pi_t(x)\, q^*(x)$$

where the $\pi_t(x)$ are functions of the preferences.

Gradient ascent

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

Since $q^*(x)$ are not known, one considers *stochastic gradient ascent*

$$\begin{aligned}\frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)} \left(\sum_x \pi_t(x)\, q^*(x) \right) \\ &= \sum_x q^*(x) \frac{\partial \pi_t(x)}{\partial H_t(a)}\end{aligned}$$

Gradient Bandit Algorithm - interpretation I

$$\frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} = \sum_x q^*(x) \frac{\partial \pi_t(x)}{\partial H_t(a)}$$
$$= \sum_x (q^*(x) - B_t) \frac{\partial \pi_t(x)}{\partial H_t(a)}$$

where B_t is the baseline which may be added since

$$\sum_x \frac{\partial \pi_t(x)}{\partial H_t(a)} = 0$$

as $\pi_t(x)$ is a probability mass function.

Then

$$\frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} = \sum_x \pi_t(x)(q^*(x) - B_t) \frac{\partial \pi_t(x)}{\partial H_t(a)} \frac{1}{\pi_t(x)}$$

Gradient Bandit Algorithm - interpretation II

which is an expected value of some function of the radom variable A_t

$$\frac{\partial \mathbb{E}\left(R_t\right)}{\partial H_t\left(a\right)} = \mathbb{E}\left(\left(q^*\left(A_t\right) - B_t\right) \frac{\partial \pi_t\left(A_t\right)}{\partial H_t\left(a\right)} \frac{1}{\pi_t\left(A_t\right)}\right).$$

Gradient Bandit Algorithm - interpretation I

Since $\mathbb{E}(R_t \mid A_t) = q^*(A_t)$, and taking $B_t = \overline{R}_t$, one gets

$$\frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} = \mathbb{E}\left(\left(R_t - \overline{R}_t\right) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} \frac{1}{\pi_t(A_t)}\right).$$

Assuming that

$$\frac{\partial \pi_t(x)}{\partial H_t(a)} = \pi_t(x)\left(\mathbb{I}_{a=x} - \pi_t(a)\right)$$

one has

$$\begin{aligned}\frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} &= \mathbb{E}\left(\left(R_t - \overline{R}_t\right) \pi_t(A_t)\left(\mathbb{I}_{a=A_t} - \pi_t(a)\right) \frac{1}{\pi_t(A_t)}\right) \\ &= \mathbb{E}\left(\left(R_t - \overline{R}_t\right)\left(\mathbb{I}_{a=A_t} - \pi_t(a)\right)\right).\end{aligned}$$

Gradient Bandit Algorithm - interpretation II

If one considers a single sample approximation of the mean (stochastic gradient approximation), one gets for all a

$$\begin{aligned} H_{t+1}(a) &= H_t(a) + \alpha \frac{\partial \mathbb{E}(R_t)}{\partial H_t(a)} \\ &= H_t(a) + \alpha \left(R_t - \overline{R}_t\right) \left(\mathbb{I}_{a=A_t} - \pi_t(a)\right). \end{aligned}$$

Gradient Bandit Algorithm - interpretation I

It remains to prove that

$$\frac{\partial \pi_t(x)}{\partial H_t(a)} = \pi_t(x)\left(\mathbb{I}_{a=x} - \pi_t(a)\right).$$

Gradient Bandit Algorithm - interpretation II

One has

$$\begin{aligned}
\frac{\partial \pi_t(x)}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)}\left(\frac{\exp(H_t(x))}{\sum_y \exp(H_t(y))}\right) \\
&= \frac{\frac{\partial \exp(H_t(x))}{\partial H_t(a)} \sum_y \exp(H_t(y)) - \exp(H_t(x)) \frac{\partial \sum_y \exp(H_t(y))}{\partial H_t(a)}}{\left(\sum_y \exp(H_t(y))\right)^2} \\
&= \frac{\mathbb{I}_{a=x} \exp(H_t(x)) \sum_y \exp(H_t(y)) - \exp(H_t(x)) \exp(H_t(a))}{\left(\sum_y \exp(H_t(y))\right)^2} \\
&= \frac{\mathbb{I}_{a=x} \exp(H_t(x))}{\sum_y \exp(H_t(y))} - \frac{\exp(H_t(x)) \exp(H_t(a))}{\left(\sum_y \exp(H_t(y))\right)^2} \\
&= \mathbb{I}_{a=x} \pi_t(x) - \pi_t(x) \pi_t(a) \\
&= \pi_t(x) \left(\mathbb{I}_{a=x} - \pi_t(a)\right).
\end{aligned}$$

Gradient Bandit Algorithm - interpretation III

Choice of baseline has no effect on the proof, but in practice affects the variance of the algorithm.

Outline

Associative Search (Contextual Bandits)

Only nonassociative tasks have been considered up to now
No need to associate different actions with different situations.
In these tasks the learner

- tries to find a single best action when the task is stationary,
- tries to track the best action as it changes over time when the task is nonstationary.

In general reinforcement learning task, more than one situation.

Goal: learn a *policy*, a mapping from situations to the actions that are best in those situations.

Associative Search (Contextual Bandits)

Only nonassociative tasks have been considered up to now
No need to associate different actions with different situations.
In these tasks the learner

- tries to find a single best action when the task is stationary,
- tries to track the best action as it changes over time when the task is nonstationary.

In general reinforcement learning task, more than one situation.

Goal: learn a *policy*, a mapping from situations to the actions that are best in those situations.

Associative Search (Contextual Bandits)

Only nonassociative tasks have been considered up to now
No need to associate different actions with different situations.
In these tasks the learner

- tries to find a single best action when the task is stationary,
- tries to track the best action as it changes over time when the task is nonstationary.

In general reinforcement learning task, more than one situation.

Goal: learn a *policy*, a mapping from situations to the actions that are best in those situations.

Outline

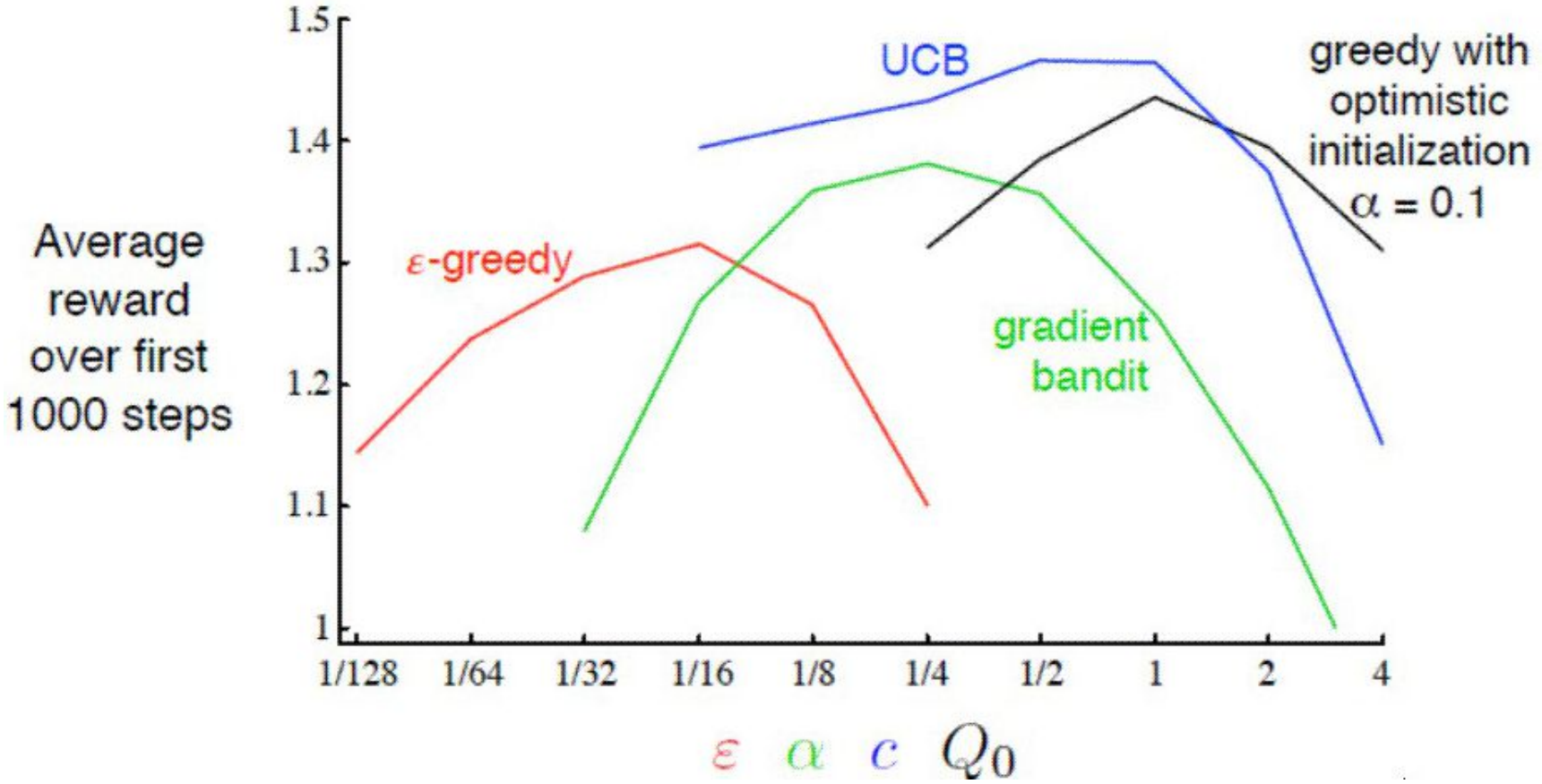
Summary I

In this course: several simple ways of balancing exploration and exploitation.

- ε-greedy methods choose randomly a small fraction of the time,
- UCB methods choose deterministically, achieving exploration by subtly favoring at each step actions having so far received fewer samples,
- Gradient bandit algorithms estimate action preferences, and favor more preferred actions in a graded, probabilistic manner using a soft-max distribution.
- Initializing estimates optimistically causes even greedy methods to explore significantly.

Which is the best approach?

Summary II



Performance of all methods depend on properly chosen parameter
Here UCB seems the best.

Perspectives I

Other well-studied approaches to balancing exploration and exploitation in k-armed bandit problems

- compute Gittins index, a special kind of action value
  - in special cases, leads directly to optimal solutions,
  - requires complete knowledge of the prior distribution of possible problems,
- use Bayesian methods
  - assume a known initial distribution over the action values
  - update the distribution exactly after each step.
  - update computations can be very complex, but for certain special distributions (called conjugate priors) they are easy.
  - select actions at each step according to posterior probability of being the best action = posterior sampling or Thompson sampling