Chapter 2
Multi-armed Bandits

2.1 A k-armed Bandit Problem

- Slot machine with k levers
- Action is a selection of a lever
- Reward is the payoffs for hitting the jackpot
- Aim:
  - Through repeated action selections, to maximize winnings
  - To find best lever(s)

- A_t: action = the selection of an arm on time step t
- R_t: reward = the corresponding reward
  = the value of the action
- $q_*(a)$: expected reward for the action a

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

- $Q_t(a)$: estimated value of action a at time step t

If you knew the value of each action, then it would be trivial to solve the k-armed bandit problem: you would always select the action with highest value.

We assume that you do not know the action values with certainty, although you may have estimates. We denote the estimated value of action a at time step t as $Q_t(a)$. We would like $Q_t(a)$ to be close to $q_*(a)$.

- Greedy action: action whose estimated value is greatest
- Exploiting: selecting greedy action
- Exploring: selecting nongreedy action

Exploitation is the right thing to do to maximize the expected reward on the one step, but exploration may produce the greater total reward in the long run.

2.2 Action-value Methods

$Q_t(a)$ by sample-average method

$$Q_t(a) = \frac{\text{sum of rewards when } a \text{ taken prior to } t}{\text{number of times } a \text{ taken prior to } t}$$

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

where

$$\mathbb{I}_p = \begin{cases} 1 \text{ if } p \text{ is true} \\ 0 \text{ otherwise} \end{cases}$$

Greedy action selection

selecting the optimal action

$$A_t = \underset{a}{\operatorname{argmax}}\, Q_t(a)$$

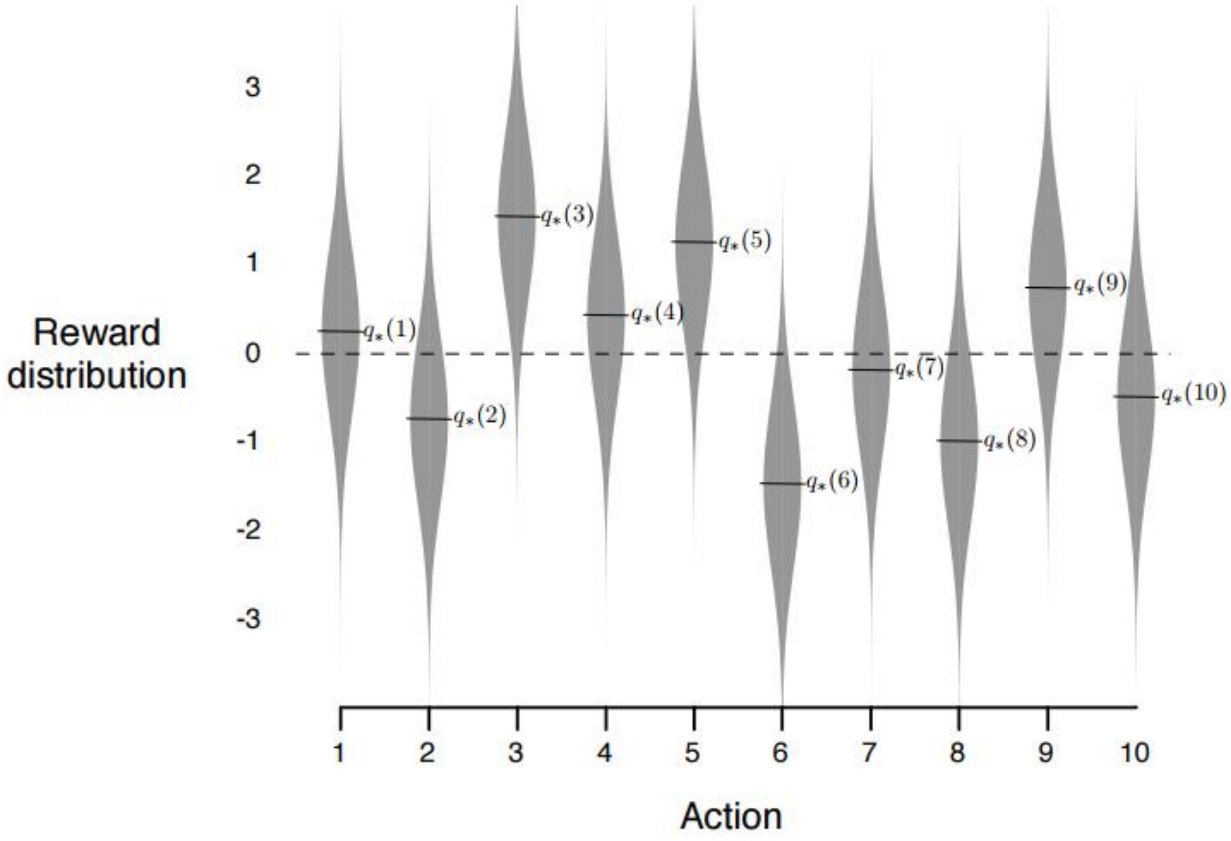
ϵ-greedy action selection

selecting action randomly with probability ϵ

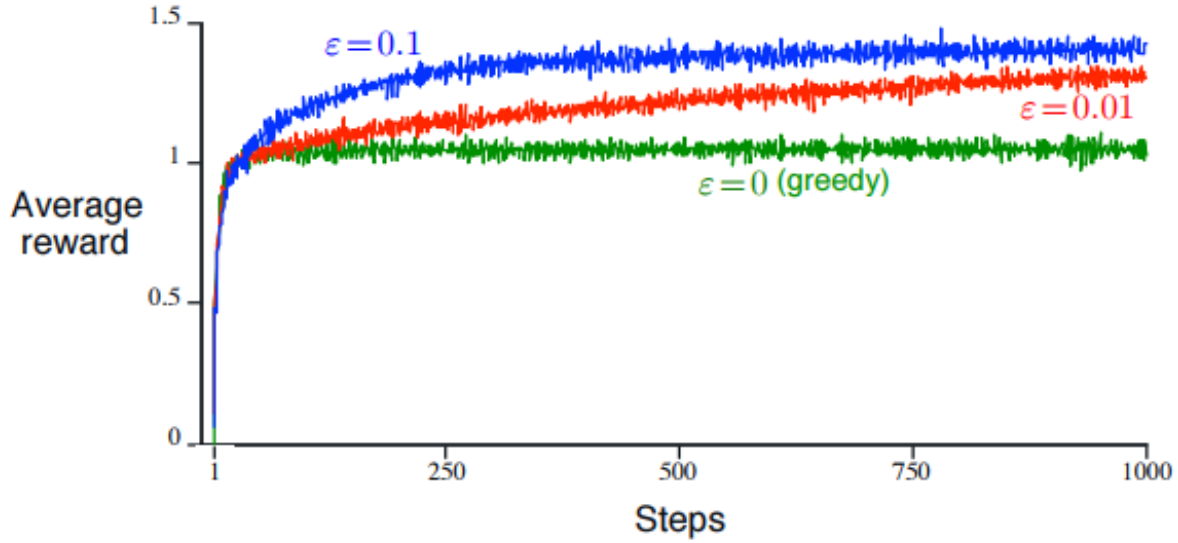
selecting the optimal action with probability $1 - \epsilon$

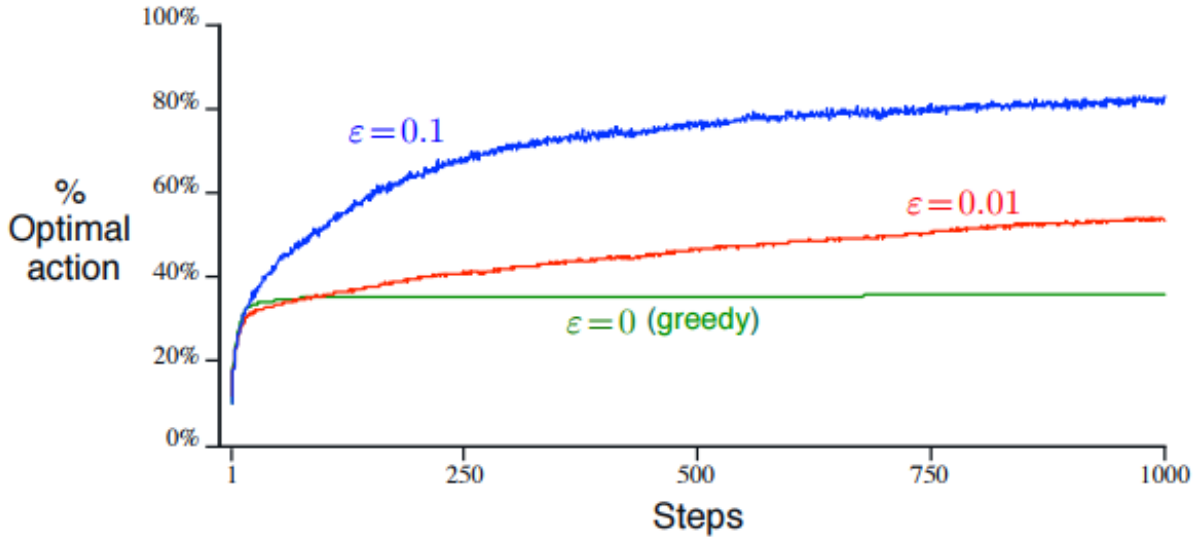
2.3 The 10-armed Testbed

- A 10-armed bandit problem
- $q_*(a) \sim \mathcal{N}(0,1)$
- $R_t(A_t) \sim \mathcal{N}(q_*(A_t), 1)$



- 2000 Randomly generated
  10-armed bandit problems
- Selecting 1000 action
  for each problem





2.4 Incremental Implementation

Strategy for efficient memory and CPU usage

$$Q_n = \frac{R_1 + \cdots + R_{n-1}}{\mathrm{n} - 1}$$

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

2.5 Tracking a Nonstationary Problem

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

Replace $\frac{1}{n}$ to α

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha[R_n - Q_n] \\ &= \alpha R_n + (1-\alpha)Q_n \\ &= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\ &= \alpha R_n + (1-\alpha)\alpha R_{n-1} + \cdots + (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1 \\ &= (1-\alpha)^n Q_1 + \alpha \sum_{i=1}^{n} (1-\alpha)^{n-i} R_i \end{aligned}$$