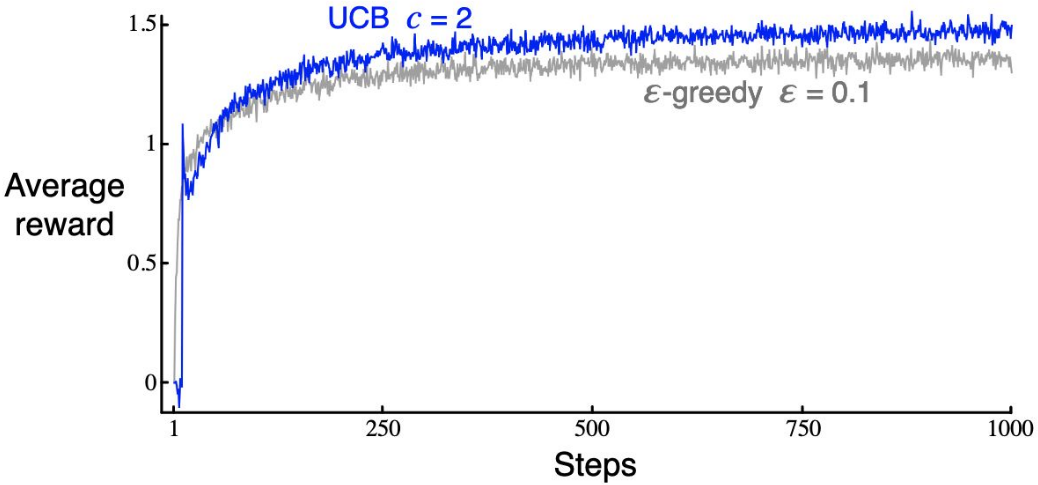
Quiz 1 Review

Plan for today

- 60 min of practice questions and interactive discussion
- 30 min of open Q&A

Q1: Bandits

Q: In a 10-armed-bandits problem, the UCB algorithm usually shows a distinct spike in performance on the 11th step. Why is this? (make sure you answer why 10th is greater than both 9th and 12th)



Q1: Bandits

Q: In a 10-armed-bandits problem, the UCB algorithm usually shows a distinct spike in performance on the 11th step. Why is this? (make sure you answer why 10th is greater than both 9th and 12th)

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

Ans: The agent will try every action in the first 10 steps. On the 11th step, the second term in UCB bonus is the same for all actions, so the agent will likely pick the action with the greatest expected reward. After picking that action, the UCB bonus for that action decreases, so we pick a different action on the 12th step, which is likely worse.

Q2: Bandits

Q: Why does the expected reward curve for UCB look so noisy, especially compared to epsilon-greedy or Boltzmann?

Q2: Bandits

Q: Why does the expected reward curve for UCB look so noisy, especially compared to Boltzmann?

Ans: At every timestep, the UCB policy is deterministic and rapidly switches between actions in order to explore. In contrast, Boltzmann uses a stochastic policy, so its expected reward changes smoothly as the action distribution changes.

Q3: Scaling rewards in bandit problems

Q: Let's say that we have 2 3-armed bandits:

- mean rewards (-1, 0, 1) and noisy Gaussian reward with variance=1
- Mean rewards (-10, 0, 10) and noisy Gaussian reward with variance=100

For the same random seed, does epsilon-greedy take the same sequence of actions? How about Boltzmann exploration?

Q3: Scaling rewards in bandit problems

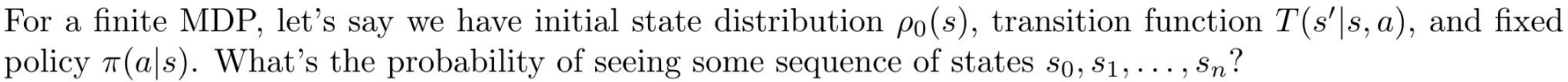
Q: Let's say that we have 2 3-armed bandits:

- Mean rewards (-1, 0, 1) and noisy Gaussian reward with variance=1
- Mean rewards (-10, 0, 10) and noisy Gaussian reward with variance=100

For the same random seed, does epsilon-greedy take the same sequence of actions? How about Boltzmann exploration?

Ans: Epsilon-greedy is scale-invariant; only the maximum Q-value determines the policy. Therefore, it takes the same sequence of actions on both. However, Boltzmann exploration depends on the gap between the Q-values, so it is much more stochastic in the first bandit problem than in the second.

Q4: MDPs



For a finite MDP, let's say we have initial state distribution $\rho_0(s)$, transition function $T(s'|s, a)$, and fixed policy $\pi(a|s)$. What's the probability of seeing some sequence of states $s_0, s_1, \ldots, s_n$?

Q4: MDPs

For a finite MDP, let's say we have initial state distribution $\rho_0(s)$, transition function $T(s'|s,a)$, and fixed policy $\pi(a|s)$. What's the probability of seeing some sequence of states $s_0, s_1, \ldots, s_n$?

$$\begin{aligned}
p(s_0, s_1, \ldots, s_n) &= p(s_0) \prod_{t=1}^{n} p(s_t | s_{t-1}) && \text{(Markov property)} \\
&= p(s_0) \prod_{t=1}^{n} \left(\sum_{a_{t-1}} p(s_t | s_{t-1}, a_{t-1}) \right) && \text{(unmarginalize action)} \\
&= \rho_0(s_0) \prod_{t=1}^{n} \left(\sum_{a_{t-1}} \pi(a_{t-1} | s_{t-1}) T(s_t | s_{t-1}, a_{t-1}) \right) && \text{(applying policy and dynamics)}
\end{aligned}$$

Q5: Definition of value and q-value

Given: $Q^{\pi,\gamma}(s,a) = \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 R_2 \cdots \mid S_0 = s, A_0 = a]$

Using **V(s)**, apply bootstrapping to timesteps $t \geq 2$ for the expression above.

Assume the State transitions are known: $S_0, S_1, S_2, \cdots$

$$= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 V^{\pi}(S_2) \mid S_0 = s, A_0 = a]$$

Do the same, but in terms of **Q(s,a)**

$$= \mathbb{E}[R_0 + \gamma R_1 + \gamma^2 \max_{a' \in A} Q^{\pi,\gamma}(S_2, a') | S_0 = s, A_0 = a]$$

Q6: Value and Policy Iteration

Q:

(True/False): At each timestep of the policy iteration algorithm, the expected reward of the current policy is guaranteed to improve.

A:

True. Policy Improvement Theorem

Q7: SARSA

Q: Can you use SARSA to learn Q-values using samples from a replay buffer of transitions collected from old policies? What about expected SARSA?

Ans:

Q7: SARSA

Q: Does SARSA naturally learn Q-values using samples from a replay buffer of transitions collected from old policies? What about expected SARSA?

Ans:

SARSA: $Q(S, A) \leftarrow Q(S, A) + \alpha[R + \gamma Q(S', A') - Q(S, A)]$

Expected SARSA: $Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma \mathbb{E}[Q(S_{t+1}, A_{t+1}) \mid S_{t+1}] - Q(S_t, A_t)\Big]$

SARSA is on-policy: A' is supposed to be drawn from the policy. (It can be extended to use a replay buffer, but this isn't in the original formulation). In contrast, expected SARSA is designed to use (s, a, s', r) from any source to perform updates.

Q8: MCTS, DQN

Q:

Which of the following statements about MCTS and DQNs is incorrect:

A. MCTS uses a tabular representations of action-values whereas DQNs uses a functional approximation.

B. Agents using DQNs are faster at choosing actions compared to those using MCTS.

C. DQNs are an example of off-policy RL whereas MCTS is on-policy.

D. MCTS have been shown to outperform comparable DQNs on some tasks.

Q8: MCTS, DQN

Q:

Which of the following statements about MCTS and DQNs is incorrect:

A. MCTS uses a tabular representations of action-values whereas DQNs uses a functional approximation.

B. Agents using DQNs are faster at choosing actions compared to those using MCTS.

C. DQNs are an example of off-policy RL whereas MCTS is on-policy.

D. MCTS have been shown to outperform comparable DQNs on some tasks.

Q8: MCTS, DQN

True: “DQNs are off-policy”: value function updated independently of the policy.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha\Big[R_{t+1} + \gamma \max_a Q(S_{t+1}, a) - Q(S_t, A_t)\Big]$$

False: “MCTS is on-policy”:

- Rollout policy outside the tree
- Tree policy inside the tree
- Actual policy when finally taking an action

Q9: REINFORCE

Q (True/False): Adding an action-dependent baseline is unbiased.

Q9: REINFORCE

Q (True/False): Adding an action-dependent baseline is unbiased.

Ans: False. We can add a constant or state-dependent baseline, but action-dependent baselines bias the REINFORCE gradient.

Q10

Q: Compare/Contrast REINFORCE with learned baseline and Actor-Critic.

A: Both methods learn a value function. Both use the value function to reduce variance in estimate of expected return:

$$\boldsymbol{\theta}_{t+1} \doteq \boldsymbol{\theta}_t + \alpha\Big(G_t - b(S_t)\Big)\frac{\nabla\pi(A_t|S_t,\boldsymbol{\theta}_t)}{\pi(A_t|S_t,\boldsymbol{\theta}_t)}. \qquad \boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha\Big(R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}) - \hat{v}(S_t,\mathbf{w})\Big)\frac{\nabla\pi(A_t|S_t,\boldsymbol{\theta}_t)}{\pi(A_t|S_t,\boldsymbol{\theta}_t)}$$

REINFORCE with learned baseline only uses the learned value function as baseline, but does not incorporate this into the actual estimate of the expected return G.

Actor-Critic uses a learned value function in bootstrapping to estimate the return G. Similarly to TD-Learning, this enables faster updates without necessarily waiting for episode termination.

General Q&A