Contents

Chapter 5

Exploration in MDPs

5.1 Introduction

One of the key challenges of reinforcement learning is the exploration-exploitation tradeoff. Should we *exploit* actions we know will give high reward, or should we *explore* different actions to discover potentially better strategies? An algorithm that doesn't explore effectively might easily *overfit* to certain areas of the state space, and fail to generalize once they enter a region they haven't yet seen. The algorithms we saw in the chapter on fitted DP (**??**) suffer from this issue.

In the multi-armed bandit chapter (**??**), where the state never changes so all we care about are the actions, we saw algorithms like UCB **??** and Thompson sampling **??** that incentivize the learner to explore arms that it is uncertain about. In this chapter, we will see how to generalize these ideas to the MDP setting.

Definition 5.1.1: Per-episode regret

To quantify the performance of a learning algorithm, we will consider its per-episode regret over T timesteps/episodes:

$$\text{Regret}_T = \mathbb{E}\left[\sum_{t=0}^{T-1} V_0^\star(s_0) - V_0^{\pi^t}(s_0)\right]$$

where π^t is the policy generated by the algorithm at the tth iteration.

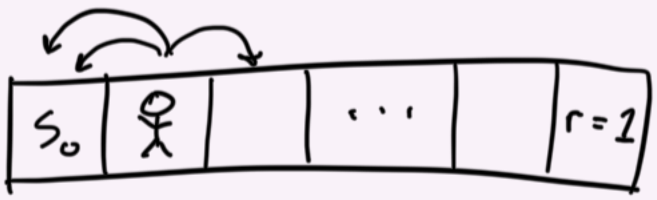
5.1.1 Sparse reward

Exploration is especially crucial in **sparse reward** problems where reward doesn't come until after many steps, and algorithms which do not *systematically* explore new states may fail to learn anything meaningful (within a reasonable amount of time).

For example, policy gradient algorithms require the gradient to be nonzero in order to learn. If we never observe any reward, the gradient will always be zero, and the policy will never change or improve.

Example 5.1.1: Sparse Reward MDP

Here's a simple example of an MDP with sparse reward:



There are $|\mathcal{S}|$ states. The agent starts in the leftmost state. In every state, there are three possible actions, two of which move the agent left and one which moves the agent right. The reward function assigns $r = 1$ to the rightmost cell.

5.1.2 Exploration in deterministic MDPs

Let us address the exploration problem in a *deterministic* MDP where taking action a in state s always leads to the state $P(s, a) \in \mathcal{S}$. In this simple setting, there will be no "automatic" exploration due to randomness, so our strategy must actively explore new states. One simple strategy is to visit every possible state-action pair to learn the entire MDP. Then, once the MDP is known, we can use DP to solve for the optimal policy. (This should remind you of the explore-then-commit algorithm for MABs (**??**).)

Definition 5.1.2: Explore-then-exploit (for deterministic MDPs)

We'll keep a set K of all the (s, a, r, s') pairs we've observed. Each episode, we'll choose an unseen state-action pair for which the reward and the next state are unknown, and take the shortest path there. We assume that every state can be reached from the initial state within a single episode.

```
$K \leftarrow \emptyset$
while $\exists(s, a)$ not explored (i.e. there is no $(s, a, r, s') \in K$) do
    Using our known transitions $K$, compute the shortest path $\widetilde{\pi}$ to $(s, a)$
    Execute $\widetilde{\pi}$ to visit $(s, a)$ and observe $r = r(s, a), s' = P(s, a)$
    $K \leftarrow K \cup \{(s, a, r, s')\}$
end while
Compute the optimal policy $\pi^\star$ in the MDP $K$ (e.g. using policy iteration).
return $\pi^\star$.
```

The shortest path computation can be implemented using DP. We leave this as an exercise.

Theorem 5.1.1: Performance of explore-then-exploit

As long as every state can be reached from s_0 within a single episode, i.e. $|\mathcal{S}| \le H$, this will eventually be able to explore all $|\mathcal{S}||\mathcal{A}|$ state-action pairs, adding one new transition per episode. We know it will take at most $|\mathcal{S}||\mathcal{A}|$ iterations to explore the entire MDP, after which $\pi^t = \pi^\star$, incurring no additional regret. For each π^t up until then, corresponding to the shortest-path policies $\widetilde{\pi}$, the value of policy π^t will differ from that of $\pi^\star$ by at most H, since the policies will differ by at most 1 reward at each timestep. So,

$$\sum_{t=0}^{T-1} V_0^\star - V_0^{\pi^t} \le |\mathcal{S}||\mathcal{A}|H.$$

(Note that this MDP and algorithm are deterministic, so the regret is not random.)

5.2 Treating an unknown MDP as a MAB

We also explored the exploration-exploitation tradeoff in the chapter on multi-armed bandit (**??**). Recall tthat in the MAB setting, we have K arms, each of which has an unknown reward distribution, and we want to learn which of the arms is *optimal*, i.e. has the highest mean reward.

One algorithm that struck a good balance between exploration and exploitation was the **upper confidence bound** algorithm (**??**): For each arm, we construct a *confidence interval* for its true mean award, and then choose the arm with the highest upper confidence bound. In summary,

$$k_{t+1} \leftarrow \arg\max_{k \in [K]} \frac{R_t^k}{N_t^k} + \sqrt{\frac{\ln(2t/\delta)}{2N_t^k}}$$

where N_t^k indicates the number of times arm k has been pulled up until time t, R_t^k indicates the total reward obtained by pulling arm k up until time t, and $\delta > 0$ controls the width of the confidence interval. How might we extend UCB to the MDP case?

Let us formally describe an unknown MDP as an MAB problem. In an unknown MDP, we want to learn which *policy* is optimal. So if we want to apply MAB techniques to solving an MDP, it makes sense to think of *arms* as *policies*. There are $K = (|\mathcal{A}|^{|\mathcal{S}|})^H$ deterministic policies in a finite MDP. Then, "pulling" arm π corresponds to using π to act through a trajectory in the MDP, and observing the total reward.

Exercise: Which quantity that we have seen so far equals the mean reward from arm π?

Recall that UCB incurs regret $\widetilde{O}(\sqrt{TK})$, where T is the number of pulls and K is the number of arms. So in the MDP-as-MAB problem, using UCB for T episodes would achieve regret

$$\widetilde{O}(\sqrt{|\mathcal{A}|^{|\mathcal{S}|H} T}). \tag{5.1}$$

This scales *exponentially* in $|\mathcal{S}|$ and H, which quickly becomes intractable. Notably, this method doesn't consider the information that we gain across different policies. We can illustrate this with the following example:

Example 5.2.1: Treating an MDP as a MAB is ineffective

Consider a "coin MDP" with two states "heads" and "tails", two actions "Y" and "N", and a time horizon of $H = 2$. The state transition flips the coin, and doesn't depend on the action. The reward only depends on the action: Taking action Y gives reward 1, and taking action N gives reward 0.

Suppose we collect data from the two constant policies $\pi_{\mathsf{Y}}(s) = \mathsf{Y}$ and $\pi_{\mathsf{N}}(s) = \mathsf{N}$. Now we want to learn about the policy $\widetilde{\pi}$ that takes action Y and then N. Do we need to collect data from $\widetilde{\pi}$ to evaluate it? No: Since the reward only depends on the action, we can infer its value from our data on the policies π_{Y} and π_{N}. However, if we treat the MDP as a bandit in which $\widetilde{\pi}$ is a new, unknown arm, we ignore the known correlation between the action and the reward.

5.3 UCB-VI

The approach above is inefficient: We shouldn't need to consider all $|\mathcal{A}|^{|\mathcal{S}|H}$ deterministic policies to achieve low regret. Rather, all we need to describe the optimal policy is $Q^\star$, which has $H|\mathcal{S}||\mathcal{A}|$ entries to be learned. Can we borrow ideas from UCB to reduce the regret to this order (i.e. polynomial in $|\mathcal{S}|$, $|\mathcal{A}|$, and H)?

One way to frame the UCB algorithm is that, when choosing arms, we optimize over a *proxy reward* that is the sum of the estimated mean reward and an exploration term. In the **UCB-VI** algorithm, we will extend this idea to the case of an unknown MDP $\mathcal{M}^?$ by modelling a proxy MDP $\widetilde{\mathcal{M}}$ with a reward function that encourages exploration. Then, we will use DP to solve for the optimal policy in $\widetilde{\mathcal{M}}$.

Assumptions: For simplicity, here we assume the reward function of $\mathcal{M}^?$ is known, so we only need to model the state transitions, though the rewards can be modelled similarly. We will also consider the more general case of a **time-varying** MDP, where the transition and reward functions can change over time. We take the convention that P_h is the distribution of $s_{h+1} \mid s_h, a_h$ and r_h is applied to s_h, a_h.

At a high level, the UCB-VI algorithm can be described as follows:

1. **Modelling:** Use previous data to model the transitions $\widehat{P}_0, \dots, \widehat{P}_{H-1}$.
2. **Reward bonus:** Design a reward bonus $b_h(s, a) \in \mathbb{R}$ to encourage exploration, analogous to the UCB term.

3. **Optimistic planning:** Use DP to compute the optimal policy $\widehat{\pi}_h(s)$ in the modelled MDP
$$\widetilde{\mathcal{M}} = (\mathcal{S}, \mathcal{A}, \{\widehat{P}_h\}_{h \in [H]}, \{r_h + b_h\}_{h \in [H]}, H).$$

4. **Execution:** Use $\widehat{\pi}_h(s)$ to collect a new trajectory, and repeat.

We detail each of these steps below. The full definition follows in (5.3.1).

5.3.1 Modelling the transitions

We seek to approximate $P_h(s_{h+1} \mid s_h, a_h) = \frac{\mathbb{P}(s_h, a_h, s_{h+1})}{\mathbb{P}(s_h, a_h)}$. We can estimate these using their sample probabilities from the dataset. That is, define

$$N_h^t(s, a, s') := \sum_{i=0}^{t-1} \mathbf{1}\left\{(s_h^i, a_h^i, s_{h+1}^i) = (s, a, s')\right\}$$
$$N_h^t(s, a) := \sum_{i=0}^{t-1} \mathbf{1}\left\{(s_h^i, a_h^i) = (s, a)\right\}$$

Then we can model
$$\widehat{P}_h^t(s' \mid s, a) = \frac{N_h^t(s, a, s')}{N_h^t(s, a)}.$$

Remark: Note that this is also a fairly naive, nonparametric estimator that doesn't assume any underlying structure of the MDP. We'll see how to incorporate assumptions about the MDP in the following section.

5.3.2 Reward bonus

To motivate the reward bonus term $b_h^t(s, a)$, recall how we designed the reward bonus term for UCB:

1. We used Hoeffding's inequality to bound, with high probability, how far the sample mean $\widehat{\mu}_t^k$ deviated from the true mean μ^k.
2. By inverting this inequality, we obtained a $(1 - \delta)$-confidence interval for the true mean, centered at our estimate.
3. To make this bound *uniform* across all timesteps $t \in [T]$, we applied the union bound and multiplied δ by a factor of T.

We'd like to do the same for UCB-VI, and construct the bonus term such that $V_h^\star(s) \le \widehat{V}_h^t(s)$ with high probability. However, our construction will be more complex than the MAB case, since $\widehat{V}_h^t(s)$ depends on the bonus $b_h^t(s, a)$ implicitly via DP. We claim that the bonus term that gives the proper bound is
$$b_h^t(s, a) = 2H\sqrt{\frac{\log(|\mathcal{S}||\mathcal{A}|HT/\delta)}{N_h^t(s, a)}}. \tag{5.2}$$

We will only provide a heuristic sketch of the proof; see [1, Section 7.3] for a full proof.

Derivation 5.3.1: UCB-VI reward bonus construction

We aim to show that, with high probability,

$$V_h^\star(s) \le \widehat{V}_h^t(s) \quad \forall t \in [T], h \in [H], s \in \mathcal{S}.$$

We'll do this by bounding the error incurred at each step of DP. Recall that DP solves for $\widehat{V}_h^t(s)$ recursively as follows:

$$\widehat{V}_h^t(s) = \max_{a \in \mathcal{A}} \left[\widetilde{r}_h^t(s, a) + \mathop{\mathbb{E}}_{s' \sim \widehat{P}_h^t(\cdot|s,a)} \left[\widehat{V}_{h+1}^t(s') \right] \right]$$

where $\widetilde{r}_h^t(s,a) = r_h(s,a) + b_h^t(s,a)$ is the reward function of our modelled MDP $\widetilde{\mathcal{M}}^t$. On the other hand, we know that $V^\star$ must satisfy

$$V_h^\star(s) = \max_{a \in \mathcal{A}} \left[\widetilde{r}_h^t(s, a) + \mathop{\mathbb{E}}_{s' \sim P_h^?(\cdot|s,a)} [V_{h+1}^\star(s')] \right]$$

so it suffices to bound the difference between the two inner expectations. There are two sources of error:

1. The value functions $\widehat{V}_{h+1}^t$ v.s. $V_{h+1}^\star$
2. The transition probabilities $\widehat{P}_h^t$ v.s. $P_h^?$.

We can bound these individually, and then combine them by the triangle inequality. For the former, we can simply bound the difference by H, assuming that the rewards are within $[0, 1]$. Now, all that is left is to bound the error from the transition probabilities:

$$\text{error} = \left| \mathop{\mathbb{E}}_{s' \sim \widehat{P}_h^t(\cdot|s,a)} \left[V_{h+1}^\star(s') \right] - \mathop{\mathbb{E}}_{s' \sim P_h^?(\cdot|s,a)} \left[V_{h+1}^\star(s') \right]. \right| \tag{5.3}$$

Let us bound this term for a fixed s, a, h, t. (Later we can make this uniform across s, a, h, t using the union bound.) Note that expanding out the definition of $\widehat{P}_h^t$ gives

$$\begin{aligned}
\mathop{\mathbb{E}}_{s' \sim \widehat{P}_h^t(\cdot|s,a)} \left[V_{h+1}^\star(s') \right] &= \sum_{s' \in \mathcal{S}} \frac{N_h^t(s,a,s')}{N_h^t(s,a)} V_{h+1}^\star(s') \\
&= \frac{1}{N_h^t(s,a)} \sum_{i=0}^{t-1} \sum_{s' \in \mathcal{S}} \mathbf{1}\left\{ (s_h^i, a_h^i, s_{h+1}^i) = (s, a, s') \right\} V_{h+1}^\star(s') \\
&= \frac{1}{N_h^t(s,a)} \sum_{i=0}^{t-1} \underbrace{\mathbf{1}\left\{ (s_h^i, a_h^i) = (s, a) \right\} V_{h+1}^\star(s_{h+1}^i)}_{X^i}
\end{aligned}$$

since the terms where $s' \ne s_{h+1}^i$ vanish.

Now, in order to apply Hoeffding's inequality, we would like to express the second term in (5.3) as a sum over t random variables as well. We will do this by redundantly averaging over all desired trajectories (i.e. where we visit state s and action a at time h):

$$\begin{aligned}\mathbb{E}_{s' \sim P^?_h(\cdot|s,a)}\left[V^\star_{h+1}(s')\right] &= \sum_{s' \in \mathcal{S}} P^?_h(s' \mid s,a) V^\star_{h+1}(s') \\ &= \sum_{s' \in \mathcal{S}} \frac{1}{N^t_h(s,a)} \sum_{i=0}^{t-1} \mathbf{1}\left\{(s^i_h, a^i_h) = (s,a)\right\} P^?_h(s' \mid s,a) V^\star_{h+1}(s') \\ &= \frac{1}{N^t_h(s,a)} \sum_{i=0}^{t-1} \mathbb{E}_{s^i_{h+1} \sim P^?_h(\cdot|s^i_h,a^i_h)} X^i.\end{aligned}$$

Now we can apply Hoeffding's inequality to $X^i - \mathbb{E}_{s^i_{h+1} \sim P^?_h(\cdot|s^i_h,a^i_h)} X^i$, which is bounded by H, to obtain that, with probability at least $1-\delta$,

$$\text{error} = \left| \frac{1}{N^t_h(s,a)} \sum_{i=0}^{t-1} \left(X^i - \mathbb{E}_{s^i_{h+1} \sim P^?_h(\cdot|s^i_h,a^i_h)} X^i \right) \right| \le 2H \sqrt{\frac{\ln(1/\delta)}{N^t_h(s,a)}}.$$

Applying a union bound over all $s \in \mathcal{S}, a \in \mathcal{A}, t \in [T], h \in [H]$ gives the $b^t_h(s,a)$ term above.

5.3.3 Definition

Putting these parts together, we can define the algorithm as follows:

Definition 5.3.1: UCB-VI

```
for $t \in [T]$ do
    $N_h(s,a,s') \leftarrow \sum_{i=0}^{t-1} \mathbf{1}\left\{(s^i_h, a^i_h, s^i_{h+1}) = (s,a,s')\right\}$
    $N_h(s,a) \leftarrow \sum_{i=0}^{t-1} \mathbf{1}\left\{(s^i_h, a^i_h) = (s,a)\right\}$
    $\widehat{P}_h \leftarrow \frac{N_h(s,a,s')}{N_h(s,a)}$                              ▷ Modelled transitions
    $b_h(s,a) \leftarrow 2H\sqrt{\frac{\log(|\mathcal{S}||\mathcal{A}|HT/\delta)}{N_h(s,a)}}$              ▷ Reward bonus
    $\widetilde{\mathcal{M}} \leftarrow (\mathcal{S}, \mathcal{A}, \{\widehat{P}_h\}_{h \in [H-1]}, \{r_h + b_h\}_{h \in [H-1]}, H)$    ▷ Modelled MDP
    $\widehat{\pi} \leftarrow \text{VI}(\widetilde{\mathcal{M}})$                  ▷ Planning with exploration bonus
    Use $\widehat{\pi}_h(s)$ to collect a new trajectory $(s^t_h, a^t_h, s^t_{h+1})_{h \in [H]}$
end for
```

5.3.4 Performance of UCB-VI

How exactly does UCB-VI strike a good balance between exploration and exploitation? In UCB for MABs, the bonus exploration term is simple to interpret: It encourages the learner to take actions with a high exploration term. Here, the policy depends on the bonus term indirectly: The policy is obtained by planning in an MDP where the bonus term is added to the reward function. Note that the bonuses *propagate backwards* in DP, effectively enabling the learner to *plan to explore* unknown states. This effect takes some further interpretation.

Recall we constructed b_h^t so that, with high probability, $V_h^\star(s) \le \widehat{V}_h^t(s)$ and so

$$V_h^\star(s) - V_h^{\pi^t}(s) \le \widehat{V}_h^t(s) - V_h^{\pi^t}(s).$$

That is, the l.h.s. measures how suboptimal policy π^t is in the true environment, while the r.h.s. is the difference in the policy's value when acting in the modelled MDP $\widetilde{\mathcal{M}}^t$ instead of the true one $\mathcal{M}^?$.

If the r.h.s. is *small*, this implies that the l.h.s. difference is also small, i.e. that π^t is *exploiting* actions that are giving high reward.

If the r.h.s. is *large*, then we have overestimated the value: π^t, the optimal policy of $\widetilde{\mathcal{M}}^t$, does not perform well in the true environment $\mathcal{M}^?$. This indicates that one of the $b_h^t(s,a)$ terms must be large, or some $\widehat{P}_h^t(\cdot \mid s,a)$ must be inaccurate, indicating a state-action pair with a low visit count $N_h^t(s,a)$ that the learner was encouraged to explore.

It turns out that UCB-VI achieves a per-episode regret of

Theorem 5.3.1: UCB-VI regret

$$\mathbb{E}\left[\sum_{t=0}^{T-1}\left(V_0^\star(s_0) - V_0^{\pi^t}(s_0)\right)\right] = \widetilde{O}(H^2\sqrt{|\mathcal{S}||\mathcal{A}|T})$$

Comparing this to the UCB regret bound $\widetilde{O}(\sqrt{TK})$, where K is the number of arms of the MAB, we see that we've reduced the number of effective arms from $|\mathcal{A}|^{|\mathcal{S}|H}$ (in (5.1)) to $H^4|\mathcal{S}||\mathcal{A}|$, which is indeed polynomial in $|\mathcal{S}|$, $|\mathcal{A}|$, and H, as desired. This is also roughly the number of episodes it takes to achieve constant-order average regret:

$$\frac{1}{T}\,\mathbb{E}[\text{Regret}_T] = \widetilde{O}\left(\sqrt{\frac{H^4|\mathcal{S}||\mathcal{A}|}{T}}\right)$$

Note that the time-dependent transition matrix has $H|\mathcal{S}|^2|\mathcal{A}|$ entries. Assuming $H \ll |\mathcal{S}|$, this shows that it's possible to achieve low regret, and achieve a near-optimal policy, while only understanding a $1/|\mathcal{S}|$ fraction of the world's dynamics.

5.4 Linear MDPs

A polynomial dependency on $|\mathcal{S}|$ and $|\mathcal{A}|$ is manageable when the state and action spaces are small. But for large or continuous state and action spaces, even this polynomial factor will become intractable. Can we find algorithms that don't depend on $|\mathcal{S}|$ or $|\mathcal{A}|$ at all, effectively reducing the dimensionality of the MDP? In this section, we'll explore **linear MDPs**: an example of a *parameterized* MDP where the rewards and state transitions depend only on some parameter space of dimension d that is independent from $|\mathcal{S}|$ or $|\mathcal{A}|$.

Definition 5.4.1: Linear MDP

We assume that the transition probabilities and rewards are *linear* in some feature vector $\phi(s, a) \in \mathbb{R}^d$:

$$\begin{aligned} P_h(s' \mid s, a) &= \phi(s, a)^\top \mu^\star_h(s') \\ r_h(s, a) &= \phi(s, a)^\top \theta^\star_h \end{aligned}$$

Note that we can also think of $P_h(\cdot \mid s, a) = \mu^\star_h$ as an $|\mathcal{S}| \times d$ matrix, and think of $\mu^\star_h(s')$ as indexing into the s'-th row of this matrix (treating it as a column vector). Thinking of $V^\star_{h+1}$ as an $|\mathcal{S}|$-dimensional vector, this allows us to write

$$\mathop{\mathbb{E}}_{s' \sim P_h(\cdot|s,a)}[V^\star_{h+1}(s)] = (\mu^\star_h \phi(s, a))^\top V^\star_{h+1}.$$

The ϕ feature mapping can be designed to capture interactions between the state s and action a. In this book, we'll assume that the feature map $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$ and the reward function (described by $\theta^\star_h$) are known to the learner.

5.4.1 Planning in a linear MDP

It turns out that $Q^\star_h$ is also linear with respect to this feature mapping. We can prove this by simply computing it using DP. We initialize $V^\star_H(s) = 0 \forall s$. Then we iterate:

$$\begin{aligned} Q^\star_h(s, a) &= r_h(s, a) + \mathop{\mathbb{E}}_{s' \sim P_h(\cdot|s,a)}[V^\star_{h+1}(s')] \\ &= \phi(s, a)^\top \theta^\star_h + (\mu^\star_h \phi(s, a))^\top V^\star_{h+1} \\ &= \phi(s, a)^\top \underbrace{(\theta^\star_h + (\mu^\star_h)^\top V^\star_{h+1})}_{w_h} \\ V^\star_h(s) &= \max_a Q^\star_h(s, a) \\ \pi^\star_h(s) &= \arg\max_a Q^\star_h(s, a) \end{aligned}$$

Exercise: Show that Q^π_h is also linear with respect to $\phi(s, a)$ for any policy π.

5.4.2 UCB-VI in a linear MDP

Modelling the transitions

This linear assumption on the MDP will also allow us to model the unknown dynamics $P^?_h(s' \mid s, a)$ with techniques from **supervised learning** (SL). Recall that SL is useful for estimating conditional expectations by minimizing mean squared error. We can rephrase the estimation of $P^?_h(s' \mid s, a)$

as a least-squares problem as follows: Write δ_s to denote a one-hot vector in $\mathbb{R}^{|\mathcal{S}|}$, with a 1 in the s-th entry and 0 everywhere else. Note that

$$\mathop{\mathbb{E}}_{s' \sim P_h(\cdot \mid s,a)}[\delta_{s'}] = P_h(\cdot \mid s, a) = \mu_h^\star \phi(s, a).$$

Furthermore, since the expectation here is linear with respect to $\phi(s, a)$, we can directly apply least-squares multi-target linear regression to construct the estimate

$$\widehat{\mu} = \arg\min_{\mu \in \mathbb{R}^{|\mathcal{S}| \times d}} \sum_{t=0}^{T-1} \|\mu \phi(s_h^i, a_h^i) - \delta_{s_{h+1}^i}\|_2^2.$$

This has a well-known closed-form solution:

$$\begin{aligned} \widehat{\mu}^\top &= (A_h^t)^{-1} \sum_{i=0}^{t-1} \phi(s_h^i, a_h^i) \delta_{s_{h+1}^i}^\top \\ \text{where} \quad A_h^t &= \sum_{i=0}^{t-1} \phi(s_h^i, a_h^i) \phi(s_h^i, a_h^i)^\top + \lambda I \end{aligned}$$

where we include a λI term to ensure that the matrix A_h^t is invertible. (This can also be derived by adding a $\lambda \|\mu\|_{\mathsf{F}}^2$ regularization term to the objective.) We can directly plug in this estimate into $\widehat{P}_h^t(\cdot \mid s, a) = \widehat{\mu}_h^t \phi(s, a)$.

Reward bonus

Now, to design the reward bonus, we can't apply Hoeffding anymore, since the terms no longer involve sample means of bounded random variables; Instead, we're incorporating information across different states and actions. Rather, we can construct an upper bound using *Chebyshev's inequality* in the same way we did for the LinUCB algorithm in the MAB setting **??**:

$$b_h^t(s, a) = \beta \sqrt{\phi(s, a)^\top (A_h^t)^{-1} \phi(s, a)}, \quad \beta = \widetilde{O}(dH).$$

Note that this isn't explicitly inversely proportional to $N_h^t(s, a)$ as in the original UCB-VI bonus term 5.2. Rather, it is inversely proportional to the amount that the direction $\phi(s, a)$ has been explored in the history. That is, if A_h^t has a large component in the direction $\phi(s, a)$, implying that this direction is well explored, then the bonus term will be small, and vice versa.

We can now plug in these transition estimates and reward bonuses into the UCB-VI algorithm 5.3.1.

Performance

Theorem 5.4.1: LinUCB-VI regret

The LinUCB-VI algorithm achieves expected regret

$$\mathbb{E}[\text{Regret}_T] = \mathbb{E}\left[\sum_{t=0}^{T-1} V_0^\star(s_0) - V_0^{\pi^t}(s_0)\right] \le \widetilde{O}(H^2 d^{1.5} \sqrt{T})$$

Comparing this to our bound for UCB-VI in an environment without this linear assumption, we see that we go from a sample complexity of $\widetilde{\Omega}(H^4|\mathcal{S}||\mathcal{A}|)$ to $\widetilde{\Omega}(H^4 d^3)$. This new sample complexity only depends on the feature dimension and not on the state or action space of the MDP!

5.5 Summary

In this chapter, we've explored how to explore in an unknown MDP.

- We first discussed the explore-then-exploit algorithm (5.1.2), a simple way to explore a deterministic MDP by visiting all state-action pairs.
- We then discussed how to treat an unknown MDP as a MAB (5.2), and how this approach is inefficient since it doesn't make use of relationships between policies.
- We then introduced the UCB-VI algorithm (5.3.1), which models the unknown MDP by a proxy MDP with a reward bonus term that encourages exploration.
- Finally, assuming that the transitions and rewards are linear with respect to a feature transformation of the state and action, we introduced the LinUCB-VI algorithm (5.4.2), which has a sample complexity independent of the size of the state and action spaces.