Contents

Chapter 5

Exploration in MDPs

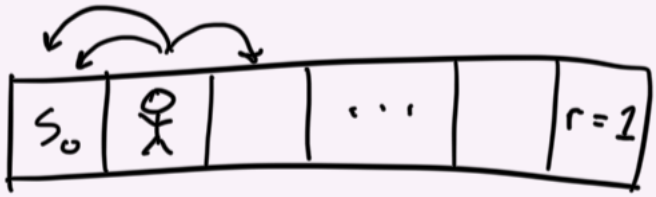
5.1 Introduction

In the chapter on fitted DP (**??**), we explored algorithms for finding the optimal value and policy in an MDP when the transition and reward functions are unknown. However, we swept the issue of *exploration* under the hood. Namely, our algorithms might easily *overfit* to certain areas of the state space, missing out on possible better paths. This issue is especially relevant in **sparse reward** problems where reward might not be achieved until after many steps, and algorithms which do not *systematically* explore new states may entirely fail to learn anything meaningful.

For example, policy gradient algorithms require *signal* in the gradient to learn. In other words, if we never observe any reward, the gradient will always be zero, and the policy will never 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. 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.

How can we address this issue?

Let's start by assuming that the MDP is *deterministic*, that is, taking action a in state s will always take you to the state $P(s, a) \in \mathcal{S}$. We will save issues of randomness for later.

Then, one algorithm to trade off exploration and exploitation is:

1. **Explore:** Visit every possible state-action pair to fully understand the MDP.
2. **Exploit:** Now the MDP is fully known, so we can use a planning algorithm like policy iteration (**??**) to solve for the optimal policy.

Definition 5.1.1: 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 pair (s, a) for which the reward $r(s, a)$ and the next state $s' = P(s, a)$ are unknown, and take the shortest path there.

$K \leftarrow \emptyset$
while $\exists (s, a)$ s.t. there is no $(s, a, r, s') \in K$ **do**
  Using our known transitions K, compute the shortest path to (s, a)
  Take the shortest path to (s, a)
  $K \leftarrow K \cup \{(s, a)\}$
end while
Compute the optimal policy $\pi^\star$ in the MDP K (e.g. using policy iteration).
return $\pi^\star$.

We leave it to the reader to design an efficient implementation of the shortest-path algorithm.

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. $|\mathcal{S}| \leq H$, this will eventually be able to explore all $|\mathcal{S}||\mathcal{A}|$ state-action pairs, adding one new transition per episode/trajectory.

We can measure the performance using the **regret** across episodes. That is, let K_t denote the value of K at iteration t, and let us compare the value of the optimal policy π_t derived from K_t with the value of the *true* optimal policy $\pi^\star$.

As we described above, we know that it will take at most SA iterations to explore the entire MDP, after which $\pi_t = \pi^\star$. For the π_t up until then, however, we can construct a very loose bound as follows. At each interaction, the reward from π_t might differ from $\pi^\star$ by at most 1. Since an episode is H steps long, this means the value of policy π_t will differ from that of $\pi^\star$ by at most H. So,

$$\sum_{t=1}^{T} V_0(\pi^\star) - V_0(\pi_t) \leq |\mathcal{S}||\mathcal{A}|H.$$

5.2 Treating an unknown MDP as a MAB

We also explored the exploration-exploitation tradeoff in the chapter on multi-armed bandits (**??**). Recall the overall framework of MAB: 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. gives the highest mean reward.

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

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

where N_t^k indicates the number of times arm k has been pulled up until time t, S_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*. This gives us $(|\mathcal{A}|^{\mathcal{S}})^H$ arms for 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. Substituting in the values above, we see that treating policies as arms and running UCB incurs regret

$$\widetilde{O}(|\mathcal{A}|^{|\mathcal{S}|H/2}N)$$

where N is the number of trajectories we get to observe. 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 a 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 policy $\widetilde{\pi}$ that takes action Y and then N. Do we need to collect data about $\widetilde{\pi}$? No! We can infer its behaviour on timestep 1 from our data on policy π_{Y} and its behaviour on timestep 2 from our data on policy π_{N}. But if we treat the

MDP as a bandit, we treat $\widetilde{\pi}$ as a new arm about which we know nothing.

5.3 UCB-VI

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.

Can we extend this idea to the case of an unknown MDP $\mathcal{M}^?$? That is, can we model a proxy MDP $\widetilde{\mathcal{M}}$ with a reward function that encourages exploration, and then use DP to solve for the optimal policy in $\widetilde{\mathcal{M}}$? This brings us to the **UCB-VI** algorithm.

Assumptions: For simplicity, here we assume the reward function of $M_?$ is known, so we only need to model the state transitions. We will 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.

Definition 5.3.1: UCB-VI

For $t \in [T]$:

1. **Modelling:** We use previous data to model the transitions $\widehat{P}_0, \dots, \widehat{P}_{H-2}$. We design a reward bonus $b_h(s,a) \in \mathbb{R}$ to encourage exploration, analogous to the UCB term.
2. **Optimistic planning:** Using DP, We solve for the optimal policy π_h in the modelled MDP
$$\widetilde{M} = (\mathcal{S}, \mathcal{A}, \{\widehat{P}_h\}_{h\in[H-1]}, \{r_h + b_h\}_{h\in[H-1]}, H).$$
Then we use π_h to collect a new trajectory, and repeat.

Note that the bonuses also *propagate backwards* via the DP algorithm. This effectively enables us to *plan to explore* unknown states.

We detail each of these steps below.

5.3.1 Modeling the transitions

Let $\mathcal{D}_h^t$ denote the dataset of transitions collected at timestep h from the first t trajectories. That is, $\mathcal{D}_h^t = \{s_h^i, a_h^i, s_{h+1}^i\}_{i\in[t]}$.

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

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

$$N_h^t(s,a) := \sum_{s' \in \mathcal{S}} N_h^t(s,a,s')$$

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 estimate, and doesn't assume any underlying structure of the MDP. Thus it would be considered a *nonparametric* model. We'll see how to improve this in the following section, and use underlying structure to improve our estimates.

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 $\widehat{\mu}_k^t$ 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 $V_h^\star \le \widehat{V}_h^t(s)$ with high probability. However, our construction will be more complex than the MAB case, since $\widehat{V}_h^t(s)$ depends on $b_h^t(s,a)$ implicitly via the DP algorithm. We claim that the bonus term that satisfies this property is

$$b_h^t(s,a) = 2H\sqrt{\frac{\log(|\mathcal{S}||\mathcal{A}|HT/\delta)}{N_h^t(s,a)}}.$$

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,

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

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 $\widetilde{M}^t$. There are two possible 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. $P_h^?$.

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, and we can combine these two bounds with the triangle inequality.

First, for a fixed s, a, h, t, we aim to bound the error caused by the state transitions:

$$\text{error} = \left| \mathop{\mathbb{E}}_{s' \sim \widehat{P}_h^t(\cdot|s,a)} \left[V_{h+1}^\star(s') \right] - \mathop{\mathbb{E}}_{s' \sim P_h^?(\cdot|s,a)} \left[V_{h+1}^\star(s') \right]. \right| \tag{5.1}$$

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 $s' \neq s_{h+1}^i$ vanish.

Now, in order to apply Hoeffding's inequality, we would like to express the second term in (5.1) 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):

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

Now we can apply Hoeffding's inequality to obtain that, with probability at least $1 - \delta$,

$$\text{error} = \left| \frac{1}{N_h^t(s,a)} \sum_{i=0}^{t-1} \left(X^i - \mathop{\mathbb{E}}_{s_{h+1}^i \sim P_h^?(\cdot|s_h^i, a_h^i)} X^i \right) \right| \leq 2H \sqrt{\frac{\ln(1/\delta)}{N_h^k(s,a)}}.$$

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

5.3.3 Performance of UCB-VI

How exactly does UCB-VI strike a good balance between exploration and exploitation? In UCB, the bonus exploration term was fairly interpretable, but now we must consider the value function of the policy returned by UCB-VI at iteration t, that is, $V_h^{\pi^t}(s)$.

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

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

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 $\widetilde{M}$, does not perform well in the true environment $M^?$. This indicates that either some $b_h^t(s,a)$ must be large, which corresponds to an action taken to *explore* the environment, or some $\widehat{P}_h^t(\cdot \mid s,a)$ is inaccurate; but notice that this correlates with a low visit count of (s,a), and thus $b_h^t(s,a)$ being large.

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

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

Comparing this to the UCB regret bound $\widetilde{O}(\sqrt{TK})$, we see that we've reduced the number of effective arms from

For some more intuition, consider how many episodes it takes for us to achieve an average regret of 1. Ignoring the H^2 factor (we'll see later why this is allowed), the average regret is

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

and for this to be constant, it would take $T = \Omega(|\mathcal{S}||\mathcal{A}|)$ steps. Note the number of entries of the $Q^\star$ matrix is of the same order. But note that the transition matrix has $|\mathcal{S}|^2|\mathcal{A}|$ entries. 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.

ssion of H^2

5.4 Linear MDPs

Above, when estimating the transition matrix $\widehat{P}_h^t$, we said that we can do in the case where we don't know anything about the structure of the MDP. But in many cases, for large or continuous $|\mathcal{S}|$ and $|\mathcal{A}|$, even some polynomial factor in $|\mathcal{S}|$ and $|\mathcal{A}|$ will become intractable. Instead, we're going to look at **linear MDPs**: an example of a *parameterized* MDP where the rewards and state transitions depend only on some *low-dimensional* parameter space.

Definition 5.4.1: Linear MDP

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

$$P_h(s' \mid s,a) = \phi(s,a)^\top \mu_h^\star(s')$$
$$r_h(s,a) = \phi(s,a)^\top \theta_h^\star$$

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

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

The ϕ feature mapping could 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 $\theta_h^\star$ are known.

Let's see what value iteration looks like in a linear MDP.

5.4.1 UCB-VI in a linear MDP

First, value iteration can be described as follows:

We initialize $V_H^\star(s) = 0 \forall s$. Then we iterate:

$$\begin{aligned}
Q_h^\star(s,a) &= r_h(s,a) + \mathop{\mathbb{E}}_{s' \sim P_h(\cdot|s,a)}[V_{h+1}^\star(s')] \\
&= \phi(s,a)^\top \theta_h^\star + (\mu_h^\star \phi(s,a))^\top V_{h+1}^\star \\
&= \phi(s,a)^\top \underbrace{(\theta_h^\star + (\mu_h^\star)^\top V_{h+1}^\star)}_{w_h} \\
V_h^\star(s) &= \max_a Q_h^\star(s,a) \\
\pi_h^\star(s) &= \arg\max_a Q_h^\star(s,a)
\end{aligned}$$

Now, we can use techniques from **supervised learning** to model the unknown MDP.

Let us write $\delta_s \in \mathcal{S}$ as a one-hot vector in $\mathbb{R}^{|\mathcal{S}|}$, with a 1 in the s-th entry and 0 everywhere else. Note that

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

Recall that **supervised learning** is precisely useful for estimating conditional expectations by minimizing mean squared error. Furthermore, since the expectation here is linear in terms of

(some function of) the observed quantities s, a, we can directly apply least-squares multi-target linear regression (with a regularization term) to construct the estimate

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

This has a well-known closed-form solution:

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

We can directly plug in this estimate into $\widehat{P}_h^t(\cdot \mid s, a) = \widehat{\mu}_h^t \phi(s, a)$.

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*.