Reinforcement Learning and REINFORCE

David S. Rosenberg

NYU: CDS

April 7, 2021

Contents

Markov Decision Processes

[Online] Stochastic k-armed contextual bandit

Stochastic k-armed contextual bandit

1. Environment samples **context** and **rewards vector** jointly, iid, for each round:

$$(X, R), (X_1, R_1), \ldots, (X_T, R_T) \in \mathcal{X} \times \mathbb{R}^k \text{ i.i.d. from } P,$$

   where $R_t = (R_t(1), \ldots, R_t(k)) \in \mathbb{R}^k$.

2. For $t = 1, \ldots, T$,
   1. Our algorithm **selects action** $A_t \in \mathcal{A} = \{1, \ldots, k\}$ based on X_t and history

$$\mathcal{D}_t = \Big((X_1, A_1, R_1(A_1)), \ldots, (X_{t-1}, A_{t-1}, R_{t-1}(A_{t-1})) \Big).$$

   2. Our algorithm **receives reward** $R_t(A_t)$.

- We **never observe** $R_t(a)$ for $a \neq A_t$.

Generalizing from contextual bandits

- Contextual bandits: contexts $X_1, \ldots, X_T$ are i.i.d.
- What about playing a video game, driving a car, moving a robot arm?
- Next context depends on previous context and action selected.
- We now want to allow dependence between consecutive X_i's.
- This is the **main difference** between reinforcement learning and contextual bandits.

Markov decision processes (MDPs)

"MDPs are a mathematically idealized form of the reinforcement learning problem for which precise theoretical statements can be made." [SB18, p. 47]

Markov decision processes

- Learner / decision maker is called the **agent**
- Agent interacts with the **environment**
- Each round $t = 0, 1, 2, 3, \ldots,$
  - agent receives a **state** $X_t \in \mathcal{X}$.
  - agent selects an action $A_t \in \mathcal{A}$
  - agent receives a reward $R_t \in \mathbb{R}$
- We get a **trajectory**: $X_0, A_0, R_0, X_1, A_1, R_1, X_2, A_2, R_2, X_3, \ldots$

MDPs, continued

- The **dynamics** of the MDP are given by

$$\mathbb{P}\left(X_{t+1}=x', R_t=r \mid X_t=x, A_t=a\right)=p(x', r \mid x, a),$$

  for any $x', x \in \mathcal{X}$, $r \in \mathbb{R}$, $a \in \mathcal{A}$.
- Gives distribution of reward and next state given previous state and action.
- Note: For simplicity, below we assume that rewards and states are discrete
  - The final algorithms will not require this. (Still need finite action space.)

Key points

1. The reward and the next state are **generated jointly**.
   - Why? e.g. allows next state to contain information about reward
2. Note that the transition probabilities have no explicit dependence on time.
   - Though we can always include time into the state x.

Episodic Learning

Episodic learning

- Often problem breaks up into "**episodes**" or "**trials**".
- For an episode there is a final time step T
  - need not be the same in every episode
  - it's typically random.
- Sometimes the task just continues, without natural breaks.
- These are called **continuing tasks**.
- In episodic learning, we typically update our policy after every episode.
- In continuing tasks, we have to update as we go
- We'll consider the episodic case, but things are similar for continuing case.

Notation

- We **could** denote the trajectories for each episode as

$$
\begin{array}{ll}
\text{Episode 1:} & X_{1,0}, A_{1,0}, R_{1,0}, X_{1,1}, A_{1,1}, R_{1,1}, X_{1,2}, A_{1,2}, R_{1,2}, X_{1,3} \\
\text{Episode 2:} & X_{2,0}, A_{2,0}, R_{2,0}, X_{2,1}, A_{2,1}, R_{2,1}, X_{2,2}, A_{2,2}, R_{2,2}, X_{2,3}, A_{2,3}, R_{2,3}, X_{2,4} \\
\text{Episode 3:} & X_{3,0}, A_{3,0}, R_{3,0}, X_{3,1}, A_{3,1}, R_{3,1}, X_{3,2} \\
\vdots & \vdots
\end{array}
$$

- However, we'll find we only need to refer to one episode at a time.
- We either take expectations w.r.t. a single episode.
- Or we're computing an update step based on a single episode.
- So we'll leave off the epsiode subscript, and just use a subscript for round.

- I think of each episode as the analogue of a single round of a contextual bandit. In fact, if each episode ends after round 1, it's exactly the contextual bandit setting (assuming we set things up as described in a previous note, where round 0 starts in a fixed start state, but the state distribution in round 1 is the same as the context distribution in the contextual bandit). So an episode is kind of an expanded version of a contextual bandit round.

Start and terminal states

- For simplicity (and w.l.o.g.),
  assume we always start in a special **start state** $x_0 \in \mathcal{X}$.
- We'll also assume we have a **terminal state** $x_{\text{stop}} \in \mathcal{X}$.
- The terminal state is an "absorbing" state: once we arrive, we never leave.
- We get no reward in the terminal state.
- Formally, this means:

$$p(x', r \mid x_{\text{stop}}, a) = \mathbb{1}\left[x' = x_{\text{stop}}\right] \mathbb{1}\left[r = 0\right].$$

- So we can either say that the final time step of a trajectory is T, or that

$$X_{T+1} = X_{T+2} = \cdots = x_{\text{stop}}$$
$$R_{T+1} = R_{T+2} = \cdots = 0$$

- **We'll assume** that $\mathbb{P}(T < \infty) = \mathbb{P}(X_t = x_{\text{stop}}, \text{some } t) = 1$.

- How can we say that starting in start state x_0 is not a loss in generality? Suppose we want to start in a random state given by $p_0(x)$. Then we can define $p(x_1, r_0 \mid x_0, a_0) = p_0(x_1)\mathbb{1}[r_0 = 0]$. In words, no matter what action is taken in round 0, the state distribution in round 1 is $p_0(x)$, as desired, and the reward received in round 0 is 0. That way the MDP is equivalent to the MDP that starts at round 1 with initial state distribution $p_0(x)$.

- Note that with our stop state convention, we can write the total reward received in an episode in two ways:

$$\sum_{t=0}^{T} R_t = \sum_{t=0}^{\infty} R_t$$

Policies and Value Functions

Policies

- A policy for an MDP at round t
  - gives a conditional distribution over action A_t
  - conditioned on the state X_t.
- In this module, we consider policies parameterized by θ: $\pi_\theta(a \mid x)$, for $\theta \in \mathbb{R}^d$.
- At round t, action $A_t \in \mathcal{A} = \{1, \ldots, k\}$ is chosen according to

$$\mathbb{P}(A_t = a \mid X_t = x) = \pi_\theta(a \mid x).$$

- Our policy parameter θ will be **fixed** for each episode.
- However, our policy can still "learn", in a certain sense, within an episode.
- Unlike contextual bandit setting, in each round of an episode,
  - the state X_t can summarize the history of play since the beginning of the episode.

The state-value function

- In contextual bandits, the **value** of a policy is the expected reward.
- In MDPs, we define a couple different value functions for a policy.

Definition (State-value function)

The **state-value function** for policy π, denoted $v_\pi(x)$ is the expected reward starting in state x and following π thereafter:

$$v_\pi(x) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} R_k \mid X_0 = x\right] \quad \forall x \in \mathcal{X}.$$

- With the convention that $X_0 = x_0$, the value of a policy is $v_\pi(x_0)$.

The action-value function

Definition (Action-value function)

The **action-value function** for policy π, denoted $q_\pi(x, a)$ is the expected reward starting in state x, taking action a, and following π thereafter:

$$q_\pi(x, a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} R_k \mid X_0 = x, A_0 = a\right] \quad \forall x \in \mathcal{X}, a \in \mathcal{A}.$$

- Since the dynamics are time-indepenent, it would be equivalent to make the definition

$$q_\pi(x, a) = \mathbb{E}_\pi\left[\sum_{k=0}^{\infty} R_{k+t} \mid X_t = x, A_t = a\right],$$

and similarly for the definition of the state-value function.

The value functions

- Exercise: Write $v_\pi(x)$ in terms of $q_\pi(x,a)$. (Let $G = \sum_{t=0}^{\infty} R_t$.):

$$
\begin{aligned}
v_\pi(x) &= \mathbb{E}_\pi\left[G \mid X_0 = x\right] \\
&= \mathbb{E}_\pi\left[\mathbb{E}_\pi\left[G \mid A_0, X_0 = x\right] \mid X_0 = x\right] \\
&= \sum_a \pi(a \mid x)\mathbb{E}_\pi\left[G \mid A_0 = a, X_0 = x\right] \\
&= \sum_a \pi(a \mid x) q_\pi(x,a)
\end{aligned}
$$

- Concept checks: In this inner expectation: $\mathbb{E}_\pi[G \mid A_0, X_0 = x]$, why did we indicate a dependency on π in the expectation?
  - Answer: Although the reward R_0 has nothing to do with the policy distribution, since we're conditioning on A_0 and X_0, all subsequent rewards will be affected by the policy distribuiton.

Intuition builder / lemma for later

Show: $q_\pi(x,a) = \mathbb{E}_\pi\left[R_t \mid (X_t, A_t) = (x,a)\right] + \sum_{x'} p(x' \mid x,a) v_\pi(x')$.
Proof: Then

$$
\begin{aligned}
q_\pi(x,a) &= \mathbb{E}_\pi\left[R_0 + \sum_{k=1}^{\infty} R_k | (X_0, A_0) = (x,a)\right] \\
&= \mathbb{E}_\pi\left[\mathbb{E}_\pi\left[R_0 + \sum_{k=1}^{\infty} R_k | X_1, R_0, (X_0, A_0) = (x,a)\right] \mid (X_0, A_0) = (x,a)\right] \\
&= \mathbb{E}_\pi\left[R_0 + \mathbb{E}_\pi\left[\sum_{k=1}^{\infty} R_k | X_1\right] \mid (X_0, A_0) = (x,a)\right] \\
&= \mathbb{E}_\pi\left[R_0 \mid (X_0, A_0) = (x,a)\right] + \mathbb{E}\left[v_\pi(X_1) \mid (X_0, A_0) = (x,a)\right] \\
&= \mathbb{E}_\pi\left[R_0 \mid (X_0, A_0) = (x,a)\right] + \sum_{x'} p(x' \mid x,a) v_\pi(x')
\end{aligned}
$$

REINFORCE

Policy gradient for contextual bandits

- We took a "policy gradient" approach to contextual bandits.
- The idea was to find the policy $\pi_\theta(a \mid x)$ that optimized

$$J(\theta) = \mathbb{E}_\theta\left[R(A)\right].$$

- We found that

$$R_t(A_t)\nabla_\theta \log \pi_{\theta_t}(A_t \mid X_t)$$

  was an unbiased estimate of $\nabla J(\theta)$.
- We uses that to form an SGD-style optimization algorithm:

$$\theta_{t+1} \leftarrow \theta_t + \eta R_t(A_t)\nabla_\theta \log \pi_{\theta_t}(A_t \mid X_t)$$

Policy gradient for MDPs

- What if we think about each action in an episode as a separate round of a contextual bandit?
- Then our update would be

$$\theta_{t+1} \leftarrow \theta_t + \eta R_t \nabla_\theta \log \pi_{\theta_t}(A_t \mid X_t).$$

- The problem: actions may lead to delayed payoffs.
- Extreme case: All intermediate rewards are 0 -- we only get a single episode-level reward at the end.
- Another approach: use the total episode reward for each round of an episode:

$$\theta_{t+1} \leftarrow \theta_t + \eta \left[\sum_{i=1}^{\infty} R_t \right] \nabla_\theta \log \pi_{\theta_t}(A_t \mid X_t).$$

- This could work...

Rewards-to-go

- But one thing doesn't seem quite right with

$$\theta_{t+1} \leftarrow \theta_t + \eta \left[\sum_{i=1}^{\infty} R_t \right] \nabla_\theta \log \pi_{\theta_t}(A_t \mid X_t).$$

- Action A_t can be penalized by poor rewards received at time $t-1$.
- Seems to make more sense to only include rewards received after A_t:

$$\theta_{t+1} \leftarrow \theta_t + \eta \left[\sum_{i=t}^{\infty} R_t \right] \nabla_\theta \log \pi_{\theta_t}(A_t \mid X_t).$$

- This is the basic REINFORCE update, which we will derive in the next section.

Proof of the Policy Gradient Theorem

The objective

- Consider policy space $\pi_\theta(a \mid x)$.
- We'd like to find θ maximizing

$$
\begin{aligned}
J(\theta) &= \mathbb{E}\left[\sum_{i=0}^{\infty} R_i \mid X_0 = x_0\right] \\
&= v_{\pi_\theta}(x_0).
\end{aligned}
$$

- Since we're only dealing with policies π_θ, we'll write

$$
v_\theta(x) := v_{\pi_\theta}(x) \qquad q_\theta(x,a) := q_{\pi_\theta}(x,a)
$$

Policy gradient theorem proof piece

- Recall that $q_\theta(x,a) = \mathbb{E}_\theta\left[R_t \mid (X_t, A_t) = (x,a)\right] + \sum_{x'} p(x' \mid x,a) v_\theta(x')$.
- So $\nabla_\theta q_\theta(x,a) = \sum_{x'} p(x' \mid x,a) \nabla_\theta v_\theta(x')$.
- Then

$$
\begin{aligned}
\nabla_\theta v_\theta(x) &= \nabla_\theta \sum_a \pi_\theta(a \mid x) q_\theta(x,a) \\
&= \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x) + \pi_\theta(a \mid x) \nabla_\theta q_\theta(x,a)\right] \\
&= \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x) + \pi_\theta(a \mid x) \sum_{x'} p(x' \mid x,a) \nabla_\theta v_\theta(x')\right]
\end{aligned}
$$

- Note that this is a recurrence relation! ($\nabla_\theta v_\theta(\cdot)$ shows up on the LHS and RHS).

Cleaning up the recurrence

- Let $\mathbb{P}_\theta(x \to x', k)$ be the prob of being in state x' in k steps:
  - conditioned on starting in state x (under policy π_θ).

$$\mathbb{P}_\theta(x \to x', k) := \mathbb{P}_\theta\left(X_k = x' \mid X_0 = x\right)$$

- Let $\phi(x) = \sum_a [q_\theta(x,a)\nabla_\theta \pi_\theta(a \mid x)]$. Then

$$\begin{aligned}
\nabla_\theta v_\theta(x) &= \sum_a \left[q_\theta(x,a)\nabla_\theta \pi_\theta(a \mid x) + \pi_\theta(a \mid x) \sum_{x'} p(x' \mid x,a) \nabla_\theta v_\theta(x') \right] \\
&= \phi(x) + \sum_a \pi_\theta(a \mid x) \sum_{x'} p(x' \mid x,a) \nabla_\theta v_\theta(x') \\
&= \phi(x) + \sum_{x'} \left[\sum_a p(x' \mid x,a) \pi_\theta(a \mid x) \right] \nabla_\theta v_\theta(x') \\
&= \phi(x) + \sum_{x'} \mathbb{P}_\theta(x \to x', 1) \nabla_\theta v_\theta(x')
\end{aligned}$$

Unrolling the recurrence

$$
\begin{aligned}
&\nabla_\theta v_\theta(x)\\
&= \phi(x)+\sum_{x'}\mathbb{P}_\theta(x\to x',1)\nabla_\theta v_\theta(x')\\
&= \phi(x)+\sum_{x'}\mathbb{P}_\theta(x\to x',1)\left[\phi(x')+\sum_{x''}\mathbb{P}_\theta(x'\to x'',1)\nabla_\theta v_\theta(x'')\right]\\
&= \phi(x)+\sum_{x'}\mathbb{P}_\theta(x\to x',1)\phi(x')+\sum_{x''}\left[\sum_{x'}\mathbb{P}_\theta(x\to x',1)\mathbb{P}_\theta(x'\to x'',1)\right]\nabla_\theta v_\theta(x'')\\
&= \phi(x)+\sum_{x'}\mathbb{P}_\theta(x\to x',1)\phi(x')+\sum_{x''}\mathbb{P}_\theta(x\to x'',2)\nabla_\theta v_\theta(x'')\\
&= \phi(x)+\sum_{x'}\mathbb{P}_\theta(x\to x',1)\phi(x')+\sum_{x''}\mathbb{P}_\theta(x\to x'',2)\nabla_\theta v_\theta(x'')+\cdots
\end{aligned}
$$

Putting it together

$$
\begin{aligned}
\nabla_\theta v_\theta(x) &= \phi(x) + \sum_{x'} \mathbb{P}_\theta(x \to x', 1)\phi(x') + \sum_{x''} \mathbb{P}_\theta(x \to x'', 2)\phi(x'') \\
&\quad + \sum_{x'''} \mathbb{P}_\theta(x \to x''', 3)\phi(x''') + \sum_{x''''} \mathbb{P}_\theta(x \to x'''', 4)\nabla_\theta v_\theta(x'''') + \cdots \\
&= \sum_{k=0}^{\infty} \sum_{x'} \mathbb{P}_\theta\left(x \to x', k\right)\phi(x')
\end{aligned}
$$

- In the last step, we use the facts that
  - for large enough k, $\mathbb{P}_\theta(x \to x', k) = 0$ for $x' \neq x_{\text{stop}}$ (by assumption), and
  - $\nabla_\theta v_\theta(x_{\text{stop}}) = 0$, since $v_\theta(x_{\text{stop}}) \equiv 0$ for all θ (by assumption).

Back to the objective

- We now bring in the start state:

$$
\begin{aligned}
\nabla J(\theta) &= \nabla_\theta v_\theta(x_0) \\
&= \sum_x \left[\sum_{k=0}^{\infty} \mathbb{P}_\theta\left(x_0 \to x, k\right) \right] \phi(x) \\
&= \sum_x \left[\sum_{k=0}^{\infty} \mathbb{E}_\theta\left[X_k = x \mid X_0 = x_0\right] \right] \phi(x) \\
&== \sum_x \left[\mathbb{E}_\theta \left[\sum_{k=0}^{\infty} \mathbb{1}\left[X_k = x'\right] \mid X_0 = x \right] \right] \phi(x) \\
&= \sum_x \eta(x) \phi(x),
\end{aligned}
$$

where $\eta(x) := \mathbb{E}_\theta\left[\sum_{k=0}^{\infty} \mathbb{1}\left[X_k = x\right] \mid X_0 = x_0\right]$, which is the expected number of visits to state x in an episode, when we start in state $X_0 = x_0$ and select actions according to π_θ.

Conclusion

- Let $\mathcal{X}' = \mathcal{X} - \{x_{\text{stop}}\}$.
- Then $\sum_{x' \in \mathcal{X}'} \eta(x')$ is the expected number of visits to any state.
- In other words, it's the expected number of rounds in an episode.
- We can write

$$
\begin{aligned}
\nabla J(\theta) &= \sum_x \eta(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right] \\
&= \left[\frac{\sum_{x' \in \mathcal{X}'} \eta(x')}{\sum_{x' \in \mathcal{X}'} \eta(x')}\right] \sum_x \eta(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right] \\
&= \left[\sum_{x'} \eta(x')\right] \sum_x \frac{\eta(x)}{\sum_{x' \in \mathcal{X}'} \eta(x')} \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right] \\
&= \left[\sum_{x'} \eta(x')\right] \sum_x \mu(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right],
\end{aligned}
$$

where $\mu(x) := \eta(x) / \sum_{x' \in \mathcal{X}'} \eta(x')$.

Policy gradient theorem for MDPs

- Summarizing our results:

$$
\begin{aligned}
\nabla J(\theta) &= \sum_x \eta(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x) \right] \\
&= \left[\sum_{x'} \eta(x') \right] \sum_x \mu(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x) \right],
\end{aligned}
$$

where

$$
\begin{aligned}
\eta(x) &= \mathbb{E}_\theta \left[\sum_{k=0}^{\infty} \mathbb{1}\left[X_k = x \right] \mid X_0 = x_0 \right] \\
\mu(x) &= \eta(x) / \sum_{x' \in \mathcal{X}'} \eta(x').
\end{aligned}
$$

Monte carlo estimates

Dropping the scalar factor

- We have
$$\nabla J(\theta) = \left[\sum_{x'} \eta(x')\right] \sum_{x} \mu(x) \sum_{a} \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right].$$
- If we only care about the step direction, and not the magnitude, then we can drop the scalar factor $\left[\sum_{x'} \eta(x')\right]$.
- This seems reasonable in a batch gradient setting, where we're doing some kind of line search for the step size.
- However, note that $\eta(x)$ has a dependence on θ, so it will be changing during our optimization.
- Not obvious that the factor can be safely "absorbed into the step size" (as stated in [SB18, p. 326]).

Monte Carlo approximations

- We have
$$\nabla J(\theta) \propto \sum_x \mu(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right].$$
- The idea of Monte Carlo policy gradient methods is that
$$\sum_x \mu(x) \sum_a \left[q_\theta(x,a) \nabla_\theta \pi_\theta(a \mid x)\right] \quad \approx \quad \mathbb{E}_{X_t \sim \theta} \sum_a \left[q_\theta(X_t,a) \nabla_\theta \pi_\theta(a \mid X_t)\right].$$
- What exactly is that expectation? To be from $\mu(x)$...
- it's like putting all the rounds from all the episodes into a bag, and sampling uniformly.
- Unfortunately... this isn't really what we're doing in REINFOCE... shrug.

All-actions method

- So a one-sample Monte Carlo approximation is

$$\sum_{a} \left[q_\theta(X_t, a) \nabla_\theta \pi_\theta(a \mid X_t) \right].$$

- We can plug-in an action-value estimate $\hat{q}_\theta(x, a)$, fit to historical data.
- Assuming we can sum over all actions, we can then compute

$$\sum_{a} \left[\hat{q}_\theta(X_t, a) \nabla_\theta \pi_\theta(a \mid X_t) \right]$$

  as our estimate to the gradient step direction.
- This is called an **all-actions** method.

REINFORCE

- To get REINFORCE, we use our "clever trick" with logs:

$$
\begin{aligned}
\sum_a \left[q_\theta(X_t, a)\nabla_\theta \pi_\theta(a \mid X_t)\right] &= \sum_a \left[q_\theta(X_t, a)\pi_\theta(a \mid X_t)\nabla_\theta \log \pi_\theta(a \mid X_t)\right] \\
&= \mathbb{E}_{A_t \sim \pi(a \mid X_t)}\left[q_\theta(X_t, A_t)\nabla_\theta \log \pi_\theta(A_t \mid X_t)\right] \\
&= \mathbb{E}_{A_t \sim \pi(a \mid X_t)}\left[\mathbb{E}_\theta\left[\sum_{k=t}^{\infty} R_k \mid X_t, A_t\right]\nabla_\theta \log \pi_\theta(A_t \mid X_t)\right] \\
&= \mathbb{E}_{A_t \sim \pi(a \mid X_t)}\left[\mathbb{E}_\theta\left[\nabla_\theta \log \pi_\theta(A_t \mid X_t)\sum_{k=t}^{\infty} R_k \mid X_t, A_t\right]\right] \\
&= \mathbb{E}_\theta\left[\nabla_\theta \log \pi_\theta(A_t \mid X_t)\sum_{k=t}^{\infty} R_k\right].
\end{aligned}
$$

- If we choose a random round from all of our episodes, and take X_t, A_t, and the sum of all subsequent rewards $\sum_{k=t}^{\infty} R_k$, then we can get an unbiased estimate for the last expression from

References

Resources

- The development of Markov decision processes (MDPs) is based on [SB18, Ch 3] Terminology was based on [CFV17].
- The proof for the policy gradient theorem is based on [SMSM00], which is essentially the same as the proof in [SB18, p. 325].
- The presentation of the recurrence part of the policy gradient theorem proof is based on Lilian Weng's blog, which is a great source for additional detail and discussion [Wen18]

References I

[CFV17] Victor Chernozhukov and Iván Fernández-Val, *Treatment effects*, Econometrics—MIT Course 14.382, Cambridge MA, 2017, MIT OpenCourseWare.

[SB18] Richard S. Sutton and Andrew G. Barto, *Reinforcement learning: An introduction*, A Bradford Book, Cambridge, MA, USA, 2018.

[SMSM00] Richard S Sutton, David McAllester, Satinder Singh, and Yishay Mansour, *Policy gradient methods for reinforcement learning with function approximation*, Advances in Neural Information Processing Systems (S. Solla, T. Leen, and K. Müller, eds.), vol. 12, MIT Press, 2000.

[Wen18] Lilian Weng, *Policy gradient algorithms*, Apr 2018, `https://lilianweng.github.io/lil-log/2018/04/08/policy-gradient-algorithms.html#proof-of-policy-gradient-theorem`.