Learning from a Stream of Data: Value Function Learning

(CSC2547: Introduction to Reinforcement Learning)

Amir-massoud Farahmand

University of Toronto & Vector Institute

Table of Contents

RL Setting and the Stream of Data

- Planning setting: the model ($\mathcal{P}$ and $\mathcal{R}$) is known.
  - VI, PI, LP
- RL setting: no access to the model; instead, we observe data of agent interacting with its environment.
  - Stream of Data

    $$X_1, A_1, R_1, X_2, A_2, R_2,$$

    with $A_t \sim \pi(\cdot|X_t)$, $X_{t+1} \sim \mathcal{P}(\cdot|X_t, A_t)$ and $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$.
- Questions:
  - How can we learn a value of policy π?
  - How can we learn V^* or Q^* (and consequently, the optimal policy π^*)?
- We (often) assume exact representation of the value function. Only feasible for finite MDPs.

Estimation of the Mean of a Random Variable

Let us start from a simple problem of estimating the mean of a random variable, given samples from it. To be concrete, assume that we are given n real-valued r.v.

$$Z_1, \dots, Z_t,$$

all drawn i.i.d. from a distribution ν.
Q: How can we estimate the expectation $m = \mathbb{E}\left[Z\right]$ with $Z \sim \nu$?

Sample Average Estimator

Use the sample (or empirical) average:

$$m_t \triangleq \frac{1}{t} \sum_{i=1}^{t} Z_i.$$

We know that under mild conditions, by the Law of Large Numbers (LLN), $m_t \to m$, almost surely.

How to Get an Online Estimator?

- The naive implementation of m_t requires storing all $Z_1, \ldots, Z_t$.
- This is infeasible when t is large.
- But we can do it online too:

$$\begin{aligned} m_{t+1} &= \frac{1}{t+1} \sum_{i=1}^{t+1} Z_i = \frac{1}{t+1} \left[\sum_{i=1}^{t} Z_i + Z_{t+1} \right] \\ &= \frac{1}{t+1} \left[t m_t + Z_{t+1} \right] \\ &= \left(1 - \frac{1}{t+1} \right) m_t + \frac{1}{t+1} Z_{t+1}. \end{aligned}$$

How to Get an Online Estimator?

Let us define $\alpha_t = \frac{1}{t+1}$. We can write

$$m_{t+1} = (1 - \alpha_t) m_t + \alpha_t Z_t.$$

The variable α_t is called the learning rate or step size.
With this choice of α_t, the estimate m_t converges to m as $t \to \infty$.
This online procedure is an example of the family of stochastic approximation (SA) methods.

Stochastic Approximation

$$\theta_{t+1} = (1 - \alpha_t)\theta_t + \alpha_t Z_t. \tag{1}$$

- Note that θ_t is a random variable.
- Various choices of α_t.
  - If $\alpha_t = \frac{1}{t+1}$, we get the sample mean estimator.
  - Fixed $\alpha_t = \alpha$.
  - $\alpha_t = \frac{c}{t^p+1}$
- Let us study the fixed $\alpha_t = \alpha$ closer.

$$\theta_{t+1} = (1 - \alpha)\theta_t + \alpha Z_t.$$

Stochastic Approximation: Fixed α

$$\theta_{t+1} = (1-\alpha)\theta_t + \alpha Z_t.$$

Studying its expectation and variance as a function of time t.
Take expectation of both sides to get

$$\begin{aligned}
\mathbb{E}\left[\theta_{t+1}\right] &= \mathbb{E}\left[(1-\alpha)\theta_t + \alpha Z_t\right] \\
&= (1-\alpha)\mathbb{E}\left[\theta_t\right] + \alpha\mathbb{E}\left[Z_t\right] \\
&= (1-\alpha)\mathbb{E}\left[\theta_t\right] + \alpha m.
\end{aligned}$$

Denote $\mathbb{E}\left[\theta_t\right]$ by $\bar{\theta}_t$ (which is not a r.v. anymore), and write the equation above as

$$\bar{\theta}_{t+1} = (1-\alpha)\bar{\theta}_t + \alpha m.$$

Stochastic Approximation: Fixed α

$$\bar{\theta}_{t+1} = (1-\alpha)\bar{\theta}_t + \alpha m.$$

We would like to study the behaviour of $\bar{\theta}_t$ as t increases. Assuming that $\theta_0 = 0$ (so $\bar{\theta}_0 = 0$) and $0 \leq \alpha < 1$, we get that

$$\begin{aligned}
\bar{\theta}_t &= \alpha m, \\
\bar{\theta}_2 &= (1-\alpha)\alpha m + \alpha m, \\
\bar{\theta}_3 &= (1-\alpha)^2\alpha m + (1-\alpha)\alpha m + \alpha m, \\
&\vdots \\
\bar{\theta}_t &= \alpha \sum_{i=0}^{t-1} (1-\alpha)^i m = \frac{\alpha m(1-(1-\alpha)^t)}{1-(1-\alpha)} = m\left[1-(1-\alpha)^t\right].
\end{aligned}$$

Stochastic Approximation: Fixed α

$$\bar{\theta}_t = m\left[1 - (1-\alpha)^t\right] \Longrightarrow \lim_{t\to\infty} \bar{\theta}_t = m.$$

- θ_t converges to m in expectation.
- Reassuring, but is not enough.
- It is imaginable that θ_t converges in expectation, but has a large deviation around its mean.

Let us compute its variance too.

Stochastic Approximation: Fixed α

Because of independent of Z_t:

$$\mathrm{Var}\left[\theta_{t+1}\right] = \mathrm{Var}\left[(1-\alpha)\theta_t + \alpha Z_t\right] = (1-\alpha)^2 \mathrm{Var}\left[\theta_t\right] + \alpha^2 \mathrm{Var}\left[Z_t\right].$$

As a quick calculation, we have that
$\mathrm{Var}\left[\theta_{t+1}\right] \geq \alpha^2 \mathrm{Var}\left[Z_t\right] = \alpha^2\sigma^2$.
We can show that

$$\lim_{t\to\infty} \mathrm{Var}\left[\theta_t\right] = \frac{\alpha\sigma^2}{2-\alpha}.$$

- For a constant α, the variance of θ_t is not going to converge to zero.
- θ_t fluctuates around its mean (in different runs of the data stream; though a similar conclusion would hold within the same sequence (θ_t) too).

Stochastic Approximation

- In order to make θ_t converge in a sense stronger than expectation, we need $\alpha_t \to 0$ with some schedule.
- $\alpha_t = \frac{1}{t+1}$ works, but is not the only acceptable one.
- But any sequence α_t going to zero is not working either.
  - It should not converge to zero too fast, as it would not allow enough adaptation. Or too slow!
- The SA Conditions:

$$\sum_{t=0}^{\infty} \alpha_t = \infty,$$
$$\sum_{t=0}^{\infty} \alpha_t^2 < \infty.$$

Online Learning of the Reward Function

Recall the immediate reward problem:

- At episode t, the agent starts at state $X_t \sim \rho \in \mathcal{M}(\mathcal{X})$.
- It chooses action $A_t \sim \pi(\cdot|X_t)$.
- It receives a reward of $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$.
- The agent then starts a new independent episode $t + 1$, and the process repeats.

The goal is to learn how to act optimally.

- When the reward function $r : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ was known, the optimal policy would be

$$\pi^*(x) \leftarrow \operatorname*{argmax}_{a \in \mathcal{A}} r(x, a).$$

- What if when we do not know the reward function?

Online Learning of the Reward Function

- Use SA to estimate $r(x, a)$.
- An extension of how we estimated the mean of a single variable $Z \sim \nu$ to many variables (one for each state-action pairs $(x, a) \in \mathcal{X} \times \mathcal{A}$).
- Denote $\hat{r}_t : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ as our estimate of r at time t.
- Let us denote the state-action-indexed sequence $\alpha_t(x, a)$ as the step size for (x, a).
- At time/episode t, the state-action pair (X_t, A_t) is selected. We update $\hat{r}_t(X_t, A_t)$ as

  $$\hat{r}_{t+1}(X_t, A_t) \leftarrow (1 - \alpha_t(X_t, A_t))\hat{r}_t(X_t, A_t) + \alpha_t(X_t, A_t)R_t, \tag{2}$$

  and do not change our estimate $\hat{r}_{t+1}(x, a)$ from what we had $\hat{r}_t(x, a)$ for all $(x, a) \neq (X_t, A_t)$.

Online Learning of the Reward Function

$$\hat{r}_{t+1}(X_t, A_t) \leftarrow (1 - \alpha_t(X_t, A_t))\hat{r}_t(X_t, A_t) + \alpha_t(X_t, A_t)R_t.$$

The SA conditions should be satisfied for each state-action pair, i.e., for any $(x, a) \in \mathcal{X} \times \mathcal{A}$, we need to have

$$\sum_{t=0}^{\infty} \alpha_t(x, a) = \infty,$$
$$\sum_{t=0}^{\infty} \alpha_t^2(x, a) < \infty.$$

Selecting $\alpha_t(x, a)$

To define $\alpha_t(x, a)$, use a counter on how many times (x, a) has been picked up to time t. We define

$$n_t(x, a) \triangleq |\{\, i : (X_i, A_i) = (x, a), i = 1, \ldots, t \,\}| \,.$$

We can then choose

$$\alpha_t(x, a) = \frac{1}{n_t(x, a)}.$$

This leads to $\hat{r}_t(x, a)$ being a sample mean of all rewards encountered at (x, a).
Q: What happens if

- the sampling distribution $X_t \sim \rho$ never chooses a particular state x_0?
- the policy $\pi(\cdot|x_0)$ never chooses a particular action a_0 at a certain state x_0?

From Reward Estimation to Action Selection

By selecting

$$a \leftarrow \pi_g(x; r) = \operatorname*{argmax}_{a \in \mathcal{A}} r(x, a),$$

we would choose the optimal action at state x. In lieu of r, we can use $\hat{r}_t : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$, estimated using the SA (2), and choose the action $A_t = \pi_g(X_t; \hat{r}_t)$ at state X_t.
This would be the greedy policy w.r.t. $\hat{r}_t$.

Problem with the Greedy Policy

- If $\hat{r}_t$ is an inaccurate estimate of r, the agent may choose a suboptimal action.
- It is also possible that it gets stuck in choosing that action forever, without any chance to improve its estimate (this is not OK).

Consider a problem where we only have one state x_1 with two actions a_1 and a_2. The reward function is

$$\begin{aligned} r(x_1, a_1) &= 1, \\ r(x_1, a_2) &= 2. \end{aligned}$$

Suppose that the reward is deterministic. Suppose that the initial estimate of the reward $\hat{r}(x_1, \cdot) = 0$.

Problem with the Greedy Policy

Assume that in the first episode $t = 1$, the agent happen to choose a_1. So its estimates would be

$$\hat{r}_2(x_1, a_1) = (1 - \alpha_1) \times 0 + \alpha_1 \times 1 > 0$$
$$\hat{r}_2(x_1, a_2) = \hat{r}_1(x_1, a_2) = 0.$$

- The next time the agent encounters x_1, the selected action would be a_1 again, and $\hat{r}_3(x_1, a_1)$ remains positive.
- Since a_2 is not selected, the value of $\hat{r}_3(x_1, a_2)$ remains zero.
- As long as the agent follows the greedy policy, it always chooses action a_1 and never chooses action a_2.
- The estimate $\hat{r}_t(x_1, a_1)$ becomes ever more accurate, but $\hat{r}_2(x_1, a_2)$ remains inaccurate.
- This is problematic as the optimal action here is a_2!

Solution: ε-Greedy

Solution: Force the agent regularly picks actions other than the one suggested by the greedy policy.
For $\varepsilon \geq 0$ and a function $\hat{r}$, we define π_ε as

$$\pi_\varepsilon(x; \hat{r}) = \begin{cases} \pi_g(x; \hat{r}) & \text{w.p. } 1 - \varepsilon, \\ \text{Uniform}(\mathcal{A}) & \text{w.p. } \varepsilon. \end{cases}$$

- The uniform choice of action in the ε-greedy helps the agent explore all actions, even if the action is seemingly suboptimal.
- The greedy part of its action select mechanism exploits the current knowledge about the reward function, and chooses the action that has the highest estimated reward.

Exploration-Exploitation Tradeoff

- Exploiting our knowledge is a reasonable choice when our knowledge about the world is accurate.
- When we have uncertainty about the world, we should not be overconfident of our knowledge and exploit it all the time, but instead explore other available actions, which might happen to be better.
- The tradeoff between exploration and exploitation is a major topic in RL and is an area of active research.

Boltzmann distribution for Exploration-Exploitation Tradeoff

Another heuristic: select actions according to the Boltzmann (or Gibbs or softmax) distribution. Given a parameter $\tau > 0$, and the reward function $\hat{r}$, the probability of selecting action a at state x is

$$\pi(a|x;\hat{r}) = \frac{\exp(\frac{\hat{r}(x,a)}{\tau})}{\sum_{a' \in \mathcal{A}} \exp(\frac{\hat{r}(x,a')}{\tau})}.$$

More weight to actions with higher estimated value (i.e., reward).

- When $\tau \to 0$, the behaviour of this distribution would be the same as the greedy policy.
- When $\tau \to \infty$, the probability of all actions would be the same (uniform distribution).

Monte Carlo Estimation for Policy Evaluation

The reward learning problem is a special case of value function learning problem when the episode ends in one time step.
Goal: Methods to learn (or estimate) the value function V^{π} and Q^{π} of a policy.

Monte Carlo Estimation for Policy Evaluation

Recall that

$$V^{\pi}(x) = \mathbb{E}\left[G_t^{\pi} | X_t = x\right],$$

with $G_t^{\pi} \triangleq \sum_{k \geq t} \gamma^{k-t} R_k$.
So G_t^{π} (conditioned on starting from $X_t = x$) plays the same rule as the r.v. Z in estimating $m = \mathbb{E}\left[Z\right]$.
Obtaining a sample from return G^{π} is easy, at least conceptually:
If the agent starts at state x, and follows π, we can draw one sample of r.v. G^{π} by computing the cumulative average of rewards collected during the episode.
Each trajectory is sometimes called a rollout.

Monte Carlo Estimation for Policy Evaluation

If we repeat this process from the same state, we get another draw of r.v. G^π.
Let us call the value of these samples $G^{\pi(1)}(x), G^{\pi(2)}(x), \dots, G^{\pi(n)}(x)$. We can get an estimate $\hat{V}(x)$ of $V^\pi(x)$ by taking the sample average:

$$\hat{V}^\pi(x) = \frac{1}{n} \sum_{i=1}^{n} G^{\pi(i)}(x).$$

We can also use a SA procedure too.

Monte Carlo Estimation (PE) (Initial-State Only)

```
Require: Step size schedule $(\alpha_t(x))_{t \geq 1}$ for all $x \in \mathcal{X}$.
1: Initialize $\hat{V}_1^\pi : \mathcal{X} \to \mathbb{R}$ arbitrary, e.g., $\hat{V}_1^\pi = 0$.
2: for each episode $t$ do
3:     Initialize $X_1^{(t)} \sim \rho$
4:     for each step $t$ of episode do
5:         Follow $\pi$ to obtain $X_1^{(t)}, A_1^{(t)}, R_1^{(t)}, X_2^{(t)}, A_2^{(t)}, R_2^{(t)}, \ldots$.
6:     end for
7:     Compute $G_1^{\pi(t)} = \sum_{k \geq 1} \gamma^{k-1} R_k^{(t)}$.
8:     Update
           $\hat{V}_{t+1}^\pi(X_1^{(t)}) \leftarrow \left(1 - \alpha_t(X_1^{(t)})\right) \hat{V}_t^\pi(X_1^{(t)}) + \alpha_t(X_1^{(t)}) G_1^{\pi(t)}$.
9: end for
```

First-Visit and Every-Visit Monte Carlo Estimators

The previous procedure might be a bit wasteful of our data. How can we improve it?

Temporal Difference Learning for Policy Evaluation

- MC allows us to estimate $V^\pi(x)$ by using returns $G^\pi(x)$.
- MC does not benefit from the recursive property of the value function.
- MC is agnostic to the MDP structure.
  - Advantageous: If the problem is not an MDP.
  - Disadvantageous: If the problem is an MDP.
- We have seen methods benefitting from the structure of the MDP in the previous lecture. Can we use similar methods, even if we do not know $\mathcal{P}$ and $\mathcal{R}$?

Temporal Difference Learning for Policy Evaluation

Let us focus on VI for PE: At state x, the procedure is

$$V_{k+1}(x) \leftarrow r^{\pi}(x) + \gamma \int \mathcal{P}(\mathrm{d}x'|x,a)\pi(\mathrm{d}a|x)V_k(x').$$

If we do not know r^{π} and $\mathcal{P}$, we cannot compute this.
Suppose that we have n samples $A_i \sim \pi(\cdot|x)$, $X'_i \sim \mathcal{P}(\cdot|x, A_i)$, and $R_i \sim \mathcal{R}(\cdot|x, A_i)$.
Using these samples and V_k, we compute

$$Y_i = R_i + \gamma V_k(X'_i).$$

Now notice that

$$\mathbb{E}\left[R_i | X = x\right] = r^{\pi}(x),$$

and

$$\mathbb{E}\left[V_k(X'_i) | X = x\right] = \int \mathcal{P}(\mathrm{d}x'|x,a)\pi(\mathrm{d}a|x)V_k(x').$$

Temporal Difference Learning for Policy Evaluation

So the r.v. Y_i satisfies

$$\mathbb{E}\left[Y_i | X = x\right] = \mathbb{E}\left[R_i + \gamma V_k(X_i') | X = x\right] = (T^\pi V_k)(x).$$

This means that Y_i is an unbiased sample from the effect of T^π on V_k, evaluated at x.
We can use a sample mean to estimate $(T^\pi V_k)(x)$. Or we can devise a SA procedure.

Empirical Bellman Operator

The empirical Bellman operator:

$$(\hat{T}^{\pi} V_k)(x) \triangleq R(x) + \gamma V_k(X'(x)),$$

It provides an unbiased estimate of $(T^{\pi} V_k)(x)$:

$$\mathbb{E}\left[(\hat{T}^{\pi} V_k)(x) | X = x\right] = (T^{\pi} V_k)(x).$$

The empirical version of the VI:

$$V_{k+1} \leftarrow \hat{T}^{\pi} V_k = T^{\pi} V_k + \underbrace{\left(\hat{T}^{\pi} V_k - T^{\pi} V_k\right)}_{\triangleq \varepsilon_k}.$$

- A deterministic part
- A stochastic part

Temporal Difference Learning (Synchronous)

```
Require: Policy $\pi$, step size schedule $(\alpha_k)_{k \geq 1}$.
1: Initialize $V_1 : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ arbitrary, e.g., $V_1(x) = 0$.
2: for iteration $k = 1, 2, \ldots$ do
3:     for each state $x \in \mathcal{X}$ do
4:         Let $A \sim \pi(\cdot|x)$
5:         $X'(x) \sim \mathcal{P}(\cdot|X, A)$ and $R(x) \sim \mathcal{R}(\cdot|x, A)$
6:         Let $(\hat{T}^\pi V_k)(x) \triangleq R(x) + \gamma V_k(X'(x))$
7:     end for
8:     Update
           $V_{k+1} \leftarrow (1 - \alpha_k) V_k + \alpha_k \hat{T}^\pi V_k$
9: end for
```

Temporal Difference Learning: From Synchronous to Asynchronous

We do not need to update all states at the same time.

Temporal Difference Learning

```
Require: Policy $\pi$, step size schedule $(\alpha_t)_{t \geq 1}$.
1: Initialize $V_1 : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ arbitrary, e.g., $V_1(x) = 0$.
2: Initialize $X_1 \sim \rho$
3: for each step $t = 1, 2, \ldots$ do
4:     Let $A_t \sim \pi(\cdot|x)$
5:     Take action $A_t$, observe $X_{t+1} \sim \mathcal{P}(\cdot|X_t, A_t)$ and
   $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$
6:     Update
       $$V_{t+1}(x) \leftarrow \begin{cases} V_t(x) + \alpha_t(x)[R_t + \gamma V_t(X_{t+1}) - V_t(X_t)] & x = X_t \\ V_t(x) & x \neq X_t \end{cases}$$
7: end for
```

Temporal Difference Learning for Policy Evaluation

The update rule could be written in perhaps a simpler, but less precise, form of

$$V(X_t) \leftarrow V(X_t) + \alpha_t(X_t)[R_t + \gamma V(X_{t+1}) - V(X_t)],$$

without showing any explicit dependence of V on time index t.

Temporal Difference Error

The term

$$\delta_t \triangleq R_t + \gamma V(X_{t+1}) - V(X_t)$$

is called *temporal difference (TD) error*.
This is a noisy measure of how close we are to V^π.
To see this more clearly, let us define the dependence on the TD error on its components more explicitly: Given a transition (X, A, R, X') and a value function V, we define

$$\delta(X, R, X'; V) \triangleq R + \gamma V(X') - V(X).$$

We have

$$\mathbb{E}\left[\delta(X, R, X'; V) | X = x\right] = (T^\pi V)(x) - V(x) = \mathsf{BR}(V)(x).$$

So in expectation, the TD error is equal to the Bellman residual of V, evaluated at state x.
Recall that the Bellman residual is zero when $V = V^\pi$.

TD Learning for Action-Value Function

We can use a similar procedure to estimate the action-value function.
To evaluate π, we need to have an estimate of $(T^\pi Q)(x, a)$ for all $(x, a) \in \mathcal{X} \times \mathcal{A}$.
Suppose that $(X_t, A_t) \sim \mu$ and $X'_t \sim \mathcal{P}(\cdot|X_t, A_t)$ and $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$.
The update rule would be

$$Q_{t+1}(X_t, A_t) \leftarrow Q_t(X_t, A_t) + \alpha_t(X_t, A_t) \left[R_t + \gamma Q_t(X'_t, \pi(X'_t)) - Q_t(X_t, A_t) \right],$$

and

$$Q_{t+1}(x, a) \leftarrow Q_t(x, a)$$

for all other $(x, a) \neq (X_t, A_t)$.
It is easy to see that

$$\mathbb{E}\left[R_t + \gamma Q_t(X'_t, \pi(X'_t)) | X = x, A = a \right] = (T^\pi Q)(x, a).$$

On-policy and Off-policy Sampling Scenarios

$$Q_{t+1}(X_t, A_t) \leftarrow Q_t(X_t, A_t) + \alpha_t(X_t, A_t) \left[R_t + \gamma Q_t(X_t', \pi(X_t')) - Q_t(X_t, A_t)\right].$$

Observation:

- π appears only in $Q_t(X_t', \pi(X_t'))$ term.
- The action A_t does not need to be selected by π itself.

This entails that the agent can generate the stream of data $X_1, A_1, R_1, X_2, A_2, R_2, \dots$ by following a policy π_b that is different from the policy that we want to evaluate π.

On-policy and Off-policy Sampling Scenarios

- When $\pi_b = \pi$, we are in the on-policy sampling scenario, in which the agent is evaluating the same policy that it is following.
- When $\pi_b \neq \pi$, we are in the off-policy sampling scenario, in which the agent is evaluating a policy that is different from the one it is following.

Monte Carlo Estimation for Control

- We can use similar methods for solving the control problem, i.e., finding the optimal value function and the optimal policy.
- The general idea is to use some version of PI.
- If we run many rollouts from each state-action pair (x, a), we can define $\hat{Q}_t^\pi$ that converges to Q^π.
- If we wait for an infinite time, $\hat{Q}_\infty^\pi = \lim_{t\to\infty} \hat{Q}_t^\pi = Q^\pi$. We can then choose $\pi' \leftarrow \pi_g(\hat{Q}_\infty^\pi)$.
- This PI can be described by the following sequence of π and Q^π:

$$\pi_0 \xrightarrow{\mathsf{E}} Q^{\pi_0} \xrightarrow{\mathsf{I}} \pi_1 \xrightarrow{\mathsf{E}} Q^{\pi_1} \xrightarrow{\mathsf{I}} \cdots .$$

Monte Carlo Estimation for Control

- We do not need to have a very accurate estimation of Q^{π_k} before performing the policy improvement step.
- We can perform MC for a finite number of rollouts from each state, and then perform the improvement step.

Monte Carlo Control (Initial-State Only)

```
Require: Initial policy $\pi_1$, step size schedule $(\alpha_k)_{k \geq 1}$.
 1: Initialize $Q_1 : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ arbitrary, e.g., $Q_1 = 0$.
 2: for each iteration $k = 1, 2, \ldots$ do
 3:     for all $(x, a) \in \mathcal{X} \times \mathcal{A}$ do
 4:         Initialize $X_1 = x$ and $A_1 = a$.
 5:         Generate an episode from $X_1$ by choosing $A_1$, and then
            following $\pi_k$ to obtain $X_1, A_1, R_1, X_2, A_2, R_2, \ldots$.
 6:         Compute $G_1^{\pi_k}(X_1, A_1) = \sum_{t \geq 1} \gamma^{t-1} R_t$.
 7:         Update

    $\hat{Q}_{k+1}^{\pi}(X_1, A_1) \leftarrow (1 - \alpha_k(X_1, A_1))\,\hat{Q}_k^{\pi}(X_1, A_1) + \alpha_k(X_1, A_1) G_1^{\pi_k}(X_1, A_1)$

 8:     end for
 9:     Improve policy: $\pi_{k+1} \leftarrow \pi_g(Q_{k+1})$.
10: end for
```

Monte Carlo Control (Initial-State Only)

Proposition (Convergence of MC for Control – Proposition 5 of Tsitsiklis 2002)

The sequence Q_k generated by the previous algorithm with the learning rate (α_k) satisfying the SA conditions (3) converges to Q^* almost surely.

$$\sum_{t=0}^{\infty} \alpha_t(x) = \infty, \qquad \sum_{t=0}^{\infty} \alpha_t^2(x) < \infty. \tag{3}$$

Temporal Difference Learning for Control: Q-Learning

We can use TD-like methods for the problem of control too. Consider any $Q \in \mathcal{B}(\mathcal{X} \times \mathcal{A})$. Let $X' \sim \mathcal{P}(\cdot|X, A)$ and $R \sim \mathcal{R}(\cdot|X, A)$ and define

$$Y = R + \gamma \max_{a' \in \mathcal{A}} Q(X', a').$$

We have

$$\begin{aligned}\mathbb{E}\left[Y|X = x, A = a\right] &= r(x, a) + \gamma \int \mathcal{P}(\mathrm{d}x'|x, a) \max_{a' \in \mathcal{A}} Q(x', a') \\ &= (T^* Q)(x, a).\end{aligned}$$

So Y is an unbiased noisy version of $(T^* Q)(x, a)$.
The *empirical Bellman optimality operator* is

$$(\hat{T}^* Q)(x, a) \triangleq R + \gamma \max_{a' \in \mathcal{A}} Q(X', a').$$

Q-Learning Algorithm

We can use SA to update the estimate of Q^*:

$$\begin{aligned} Q_{t+1}(X_t, A_t) \leftarrow & (1 - \alpha_t(X_t, A_t))Q_t(X_t, A_t) + \\ & \alpha_t(X_t, A_t) \left[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(X_{t+1}, a') \right] \end{aligned} \tag{4}$$

for the observed (X_t, A_t) and

$$Q_{t+1}(x, a) \leftarrow Q_t(x, a)$$

for all other states $(x, a) \neq (X_t, A_t)$.

Q-Learning Algorithm

```
Require: Step size schedule $(\alpha_k)_{k\geq 1}$.
Require: Policy mechanism $\pi$.
1: Initialize $Q : \mathcal{X} \times \mathcal{A} \to \mathbb{R}$ arbitrary, e.g., $Q = 0$.
2: Initialize $X_1 \sim \rho$
3: for each step $t$ do
4:     $A_t \sim \pi(\cdot|X_t)$,
5:     Take action $A_t$, observe $X_{t+1}$ and $R_t$
6:     Update:
       $Q_{t+1}(X_t, A_t) \leftarrow Q_t(X_t, A_t) + \alpha_t(X_t, A_t) \left[ R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(X_{t+1}, a') - Q_t(X_t, A_t) \right].$
7: end for
```

Q-Learning Algorithm

Q: What is the policy that the Q-Learning algorithm is evaluating?

SARSA Algorithm

Follow a PI-like procedure: Estimate Q^π for a given π, and perform policy improvement to obtain a new π.

- Usual PI: Wait long enough until the TD method produces a $Q \to Q^\pi$; then improve.
- Generalized policy iteration (or optimistic policy iteration): improve the policy before Q converges to Q^π

SARSA Algorithm

The SARSA algorithm:

- At state X_t
- Choose $A_t = \pi_t(X_t)$
- Receives $X_{t+1} \sim \mathcal{P}(\cdot|X_t, A_t)$ and $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$
- At the time step $t+1$, choose $A_{t+1} = \pi_t(X_{t+1})$
- Update rule:

$$\begin{aligned} Q_{t+1}(X_t, A_t) \leftarrow & (1 - \alpha_t(X_t, A_t)) Q_t(X_t, A_t) + \\ & \alpha_t(X_t, A_t) \left[R_t + \gamma Q_t(X_{t+1}, A_{t+1}) \right]. \end{aligned}$$

π_t: close to a greedy policy $\pi_g(Q_t)$, but with some amount of exploration, e.g., the ε-greedy policy.
The greedy part performs the policy improvement, while the occasional random choice of actions allows the agent to have some exploration.

Q-Learning vs SARSA

Comparing the update rules:

- Q-Learning: $\max_{a' \in \mathcal{A}} Q_t(X_{t+1}, a')$
- SARSA: $Q_t(X_{t+1}, A_{t+1}) = Q_t(X_{t+1}, \pi_t(X_{t+1}))$.

Comparing the evaluated policy:

- Q-Learning: the greedy policy $\pi_g(Q_t)$ (off-policy)
- SARSA: π_t, i.e., the same policy that selects actions (on-policy)

Stochastic Approximation: A Second Look

Suppose that we want to find the fixed-point of an operator L:

$$L\theta = \theta,$$

for $\theta \in \mathbb{R}^d$, and $L : \mathbb{R}^d \to \mathbb{R}^d$.
Consider the iterative update

$$\theta_{t+1} \leftarrow (1 - \alpha)\theta_t + \alpha L\theta_t.$$

If L is c-Lipschitz with $c < 1$ and α is small enough, this would converge.

Stochastic Approximation: A Second Look

If we do not have access to $L\theta_t$, but only its noise contaminated $L\theta_t + \eta_t$ with $\eta_t \in \mathbb{R}^d$ being a zero-mean noise, we perform

$$\theta_{t+1} \leftarrow (1 - \alpha_t)\theta_t + \alpha_t(L\theta_t + \eta_t).$$

Similar to (1), with the difference that the latter concerns the estimation of a mean given an unbiased noisy value of the mean, while here we are dealing with a noisy evaluation of an operator L being applied to θ_t.
Recall that α_t cannot be a fixed number, or the variance of the estimate would not go to zero.
We need the usual SA conditions on step sizes.

Stochastic Approximation: A General Model

Assume that at time t, the i-th component of θ_t is updated as

$$\theta_{t+1}(i) \leftarrow (1 - \alpha_t(i))\theta_t(i) + \alpha_t(i)\left[(L\theta_t)(i) + \eta_t(i)\right], \qquad (5)$$

with the understanding that $\alpha_t(j) = 0$ for $j \neq i$ (components that are not updated).
Next: We provide a result showing the convergence of θ_t to θ^*, the fixed point of L.
This requires some assumptions!

Assumptions on Noise

The history of the algorithm up to time t by F_t:

$$F_t = \{\theta_0, \theta_1, \ldots, \theta_t\} \cup \{\eta_0, \eta_1, \ldots, \eta_{t-1}\} \cup \{\alpha_0, \alpha_1, \ldots, \alpha_t\}.$$

Assumption A1

(a) For every i and t, we have $\mathbb{E}\left[\eta_t(i) | F_t\right] = 0$.

(b) Given any norm $\|\cdot\|$ on $\mathbb{R}^d$, there exist constants c_1, c_2 such that for all i and t, we have

$$\mathbb{E}\left[|\eta_t(i)|^2 | F_t\right] \leq c_1 + c_2 \|\theta_t\|^2 .$$

Convergence Result

Theorem (Convergence of the Stochastic Approximation – Proposition 4.4 of Bertsekas and Tsitsiklis 1996)

Let (θ_t) be the sequence generated by (5). Assume that

1. *(Step Size) The step sizes $\alpha_t(i)$ (for $i = 1, \ldots, d$) are non-negative and satisfy*
$$\sum_{t=0}^{\infty} \alpha_t(i) = \infty, \qquad \sum_{t=0}^{\infty} \alpha_t^2(i) < \infty.$$
2. *(Noise) The noise $\eta_t(i)$ satisfies Assumption A1.*
3. *The mapping L is a contraction w.r.t. $\|\cdot\|_\infty$ with a fixed point of θ^*.*

Then θ_t converges to θ^* almost surely.

Convergence of Q-Learning

The Q-Learning update rule (4) has the same form as the SA update rule (5):

- θ is $Q \in \mathbb{R}^{\mathcal{X} \times \mathcal{A}}$
- the operator L is the Bellman optimality operator T^*
- the index i in the SA update is the selected (X_t, A_t)
- the noise term $\eta_t(i)$ is the difference between $(T^* Q_t)(X_t, A_t)$ and the sample-based version $R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(X_{t+1}, a')$.

Convergence of Q-Learning

Theorem

Suppose that for all $(x, a) \in \mathcal{X} \times \mathcal{A}$*, the step sizes* $\alpha_t(x, a)$ *satisfy*

$$\sum_{t=0}^{\infty} \alpha_t(x, a) = \infty, \qquad \sum_{t=0}^{\infty} \alpha_t^2(x, a) < \infty.$$

Furthermore, assume that the reward is of bounded variance.
Then, Q_t *converges to* Q^* *almost surely.*

Convergence of Q-Learning (Proof)

Suppose that at time t, the agent is at state X_t, takes action A_t, gets to $X'_t \sim \mathcal{P}(\cdot|X_t, A_t)$ and $R_t \sim \mathcal{R}(\cdot|X_t, A_t)$.
The update rule of the Q-Learning algorithm can be written as

$$\begin{aligned} Q_{t+1}(X_t, A_t) \leftarrow & (1 - \alpha_t(X_t, A_t))Q_t(X_t, A_t) + \\ & \alpha_t(X_t, A_t) \left[(T^* Q_t)(X_t, A_t) + \eta_t(X_t, A_t) \right], \end{aligned}$$

with

$$\eta_t(X_t, A_t) = (R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(X'_t, a')) - (T^* Q_t)(X_t, A_t),$$

and

$$Q_{t+1}(x, a) \leftarrow Q_t(x, a) \qquad (x, a) \notin (X_t, A_t).$$

Convergence of Q-Learning (Proof)

- T^* is a γ-contraction mapping, so condition (3) of the theorem is satisfied.
- Condition (1) is assumed too.
- It remains to verify the conditions (2) on noise η_t, which are conditions (a) and (b) of Assumption A1.

Convergence of Q-Learning (Proof)

Let F_t be the history of algorithm up to and including when the step size $\alpha_t(X_t, A_t)$ is chosen, but just before X'_t and R_t are revealed. We have:

$$\mathbb{E}\left[\eta_t(X_t, A_t) | F_t\right] = \mathbb{E}\left[R_t + \gamma \max_{a' \in \mathcal{A}} Q_t(X'_t, a') \mid F_t\right] - (T^* Q_t)(X_t, A_t) = 0.$$

This verifies condition (a): zero-mean noise.

Convergence of Q-Learning (Proof)

To verify (b), we provide an upper bound on $\mathbb{E}\left[\eta_t^2(X_t, A_t) | F_t\right]$:

$$\mathbb{E}\left[\eta_t^2(X_t, A_t) \mid F_t\right] =$$

$$\mathbb{E}\Bigg[\Big|(R_t - r(X_t, A_t)) +$$

$$\gamma\left(\max_{a' \in \mathcal{A}} Q_t(X_t', a') - \int \mathcal{P}(\mathrm{d}x' | X_t, A_t) \max_{a' \in \mathcal{A}} Q_t(x', a')\right)\Big|^2 \mid F_t\Bigg]$$

$$\leq 2\mathrm{Var}\left[R_t \mid X_t, A_t\right] + 2\gamma^2 \mathrm{Var}\left[\max_{a' \in \mathcal{A}} Q_t(X', a') \mid X_t, A_t\right].$$

Convergence of Q-Learning (Proof)

We have

$$
\begin{aligned}
\mathrm{Var}\left[\max_{a' \in \mathcal{A}} Q_t(X', a') \mid X_t, A_t\right] &\le \mathbb{E}\left[\left|\max_{a' \in \mathcal{A}} Q_t(X', a')\right|^2 \mid X_t, A_t\right] \\
&\le \max_{x,a} |Q_t(x, a)|^2 \\
&\le \sum_{x,a} |Q_t(x, a)|^2 = \|Q_t\|_2^2 .
\end{aligned}
$$

Convergence of Q-Learning (Proof)

Denote the maximum variance of the reward distribution over the state-action space $\max_{(x,a)\in\mathcal{X}\times\mathcal{A}} \mathrm{Var}\left[R(x,a)\right]$ by σ_R^2, which is assumed to be bounded.
We have

$$\mathbb{E}\left[\eta_t^2(X_t, A_t) \mid F_t\right] \leq 2(\sigma_R^2 + \gamma^2 \left\|Q_t\right\|_2^2).$$

Therefore, we can choose $c_1 = 2\sigma_R^2$ and $c_2 = 2\gamma^2$ in condition b.
All conditions of Theorem 2 are satisfied, so Q_t converges to Q^* (a.s.).

Remarks

- The step size condition is state-action dependent.
- If there is a state-action pair that is not selected at all or only a finite number of times, the condition cannot be satisfied.
- We need each state-action pair to be visited infinitely often.

Remarks

- The state-action-dependence of the step size might be different from how the Q-Learning algorithm is sometimes presented, in which a single learning rate α_t is used for all state-action pairs.
- A single learning rate suffices if the agent happens to visit all $(x, a) \in \mathcal{X} \times \mathcal{A}$ frequent enough, for example every $M < \infty$ steps.
- This is only an asymptotic guarantee. It does not show anything about the convergence rate, i.e., how fast Q_t converges to Q^*.

Summary

- From Planning (known model) to Learning (unknown model)
- Stochastic Approximation for online estimation of a noisy quantity
- Methods for estimation of value function
  - Monte Carlo
  - Temporal Difference Learning
- Established convergence of Q-Learning

References

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

John N. Tsitsiklis. On the convergence of optimistic policy iteration. *Journal of Machine Learning Research (JMLR)*, 3(1): 59–72, 2002.