STATS 701 – Theory of Reinforcement Learning
Temporal Difference Methods

Ambuj Tewari

Associate Professor, Department of Statistics, University of Michigan
tewaria@umich.edu
https://ambujtewari.github.io/stats701-winter2021/

Slide Credits: Prof. M. Vidyasagar @ IIT Hyderabad, India

Winter 2021

Outline

1. Temporal Difference Learning
   - One-Step TD Learning
   - Multi-Step TD Learning
2. Temporal Difference Control
   - SARSA
   - Q-Learning

MDPs with Unknown Parameters

- Methods such as value iteration and policy iteration work when parameters of MDP are completely known.
- What if these are unknown but need to be inferred via simulation?
- That is the difference between MDPs and RL.

Outline

Outline

An Explicit Formula for the Value Function

Suppose $X_t = x_i$, the state of interest. An "explicit" formula for $V(x_i)$ is

$$V(x_i) = E\left[\sum_{\tau=0}^{\infty} \gamma^\tau R_{t+\tau+1} | X_t = x_i\right],$$

where

$$R_{t+\tau+1} = R(X_{t+\tau}, U_{t+\tau})$$

is the reward, but paid at time $t + \tau + 1$.

An Implicit Formula for the Value Function

Now the explicit formula can be expanded as

$$V(x_i) = R_{t+1} + E\left[\sum_{\tau=1}^{\infty} \gamma^{\tau} R_{t+\tau+1} | X_t = x_i\right].$$

This can now be rewritten as a recursion:

$$V(x_i) = R_{t+1} + \gamma E\left[V(X_{t+1}) | X_t = x_i\right]. \tag{1}$$

Monte Carlo Estimate Revisited

Given the sample path $\{(X_t, U_t, W_{t+1})\}_{t=0}^{T}$, the quantity

$$\hat{V}(x_i) = \sum_{\tau=0}^{T-t} \gamma^\tau W_{t+\tau+1}$$

provides an approximation to $V(x_i)$.

Disadvantage: We must wait for a complete episode to generate this estimate.

Is there an alternative?

The Temporal Difference

Write the implicit equation for V as

$$V(X_t) \sim R_{t+1} + \gamma V(X_{t+1}),$$

where the symbol $\sim$ denotes that the random variables on both sides of the formula are the same. So, if $\hat{\mathbf{v}} \in \mathbb{R}^n$ is a current guess for the value vector at time t, then we can take $\hat{V}(X_t)$ as a proxy for $V(X_t)$, W_{t+1} as a proxy for R_{t+1}, and $\hat{V}(X_{t+1})$ as a proxy for $V(X_{t+1})$, and define the temporal difference

$$\delta_{t+1} = W_{t+1} + \gamma \hat{V}_t(X_{t+1}) - \hat{V}_t(X_t).$$

Interpretation of the Temporal Difference

Recall

$$\delta_{t+1} = W_{t+1} + \gamma \hat{V}_t(X_{t+1}) - \hat{V}_t(X_t),$$

If $\hat{\mathbf{v}}_t = \mathbf{v}$, the true value vector, then

$$E[\delta_{t+1}|X_t = x_i] = E[W_{t+1} + \gamma \hat{V}(X_{t+1}) - \hat{V}(X_t)|X_t = x_i] = 0.$$

So δ_{t+1} gives a measure of how erroneous the estimate $\hat{V}(x_i)$ is. But it does not tell us how far off the estimates $\hat{V}(x_j), x_j \neq x_i$ are.

Temporal Difference Learning

Select a sequence of step sizes $\{\alpha_t\}_{t\geq 0}$ beforehand. Start off iterations with $\hat{\mathbf{v}} = \mathbf{0}$.

In terms of programming, TD update is very simple:

$$\hat{V}(X_t) = \hat{V}(X_t) + \alpha_t(W_{t+1} + \gamma \hat{V}(X_{t+1}) - \hat{V}_t(X_t))$$

For analysis, at time t, define the TD error:

$$\delta_{t+1} = W_{t+1} + \gamma \hat{V}_t(X_{t+1}) - \hat{V}_t(X_t),$$

and update $\hat{\mathbf{v}}_t$ as follows:

$$\hat{V}_{t+1}(x_j) = \begin{cases} \hat{V}_t(x_j) + \alpha_t \delta_{t+1} & \text{if } X_t = x_j, \\ \hat{V}_t(x_j), & \text{otherwise.} \end{cases}$$

Convergence of Temporal Difference Learning

See Section 2.1.1 of the book by Csaba Szepesvári.

Theorem

If all states are visited infinitely often and we select step size satisfying the Robbins-Monro conditions

$$\sum_{n=0}^{\infty} \alpha_n = \infty, \sum_{n=0}^{\infty} \alpha_n^2 < \infty.$$

Then the estimated value vector $\hat{\mathbf{v}}_t$ converges to the true value vector $\mathbf{v}$ almost surely.

Advantages of Temporal Difference Learning

- $\hat{V}$ is updated at each time step.
- Bootstrapping: Current estimate is based on previous estimate.
- Markov process need not have any absorbing states.
- If there are absorbing states, even partial episodes are useful.

Relationship to Monte Carlo Return

The Monte Carlo return over an episode can be expressed as a sum of temporal differences. Define

$$G_t := \sum_{\tau=0}^{T-t} \gamma^\tau W_{t+\tau+1}$$

to be the total return over an episode starting at time t and ending at T. Then G_t satisfies the recursion

$$G_t = W_{t+1} + \gamma G_{t+1}.$$

Relationship to Monte Carlo Return (Cont'd)

So we can write

$$\begin{aligned} G_t - \hat{V}(X_t) &= W_{t+1} + \gamma G_{t+1} - \hat{V}(X_t) \\ &= W_{t+1} + \gamma G_{t+1} - \hat{V}(X_t) - \gamma \hat{V}(X_{t+1}) + \gamma \hat{V}(X_{t+1}) \\ &= \delta_t + \gamma(G_{t+1} - \hat{V}(X_{t+1})). \end{aligned}$$

Now repeat until the end of the episode, when both G_{T+1} and $\hat{V}(X_{T+1})$ are zero (because X_T is an absorbing state). This leads to

$$G_t - \hat{V}(X_t) = \sum_{\tau=0}^{T-t} \gamma^\tau \delta_{t+\tau}.$$

Note: We assumed here that $\hat{V}$ is not updated during the episode

MC vs TD

Book by Csaba Szepesvári shows two examples in Section 2.1.2 one in which MC is slower to learn whereas in the other TD is slower (but only for a specific state)

When comparing TD and MC methods, Sutton & Barto (2nd ed) state: "At the current time this is an open question in the sense that no one has been able to prove mathematically that one method converges faster than the other. In fact, it is not even clear what is the most appropriate formal way to phrase this question!"

Outline

Multi-Step TD Learning: Philosophy

Define the return

$$G_t^{t+l} := \sum_{\tau=0}^{l-1} \gamma^t R_{t+\tau+1}.$$

Then with $l = 1$ we get

$$G_t^{t+l} = G_t = R_{t+1}.$$

Easy to verify that

$$V(X_t) \sim G_t^{t+l} + \gamma^l V(X_{t+l}).$$

(Analog of one-step recursion). Can we use this to define a "multi-step" temporal difference?

Multi-Step Temporal Difference

If $\hat{\mathbf{v}} = \mathbf{v}$, the true value vector, then

$$E[G_t^{t+l} + \gamma^l \hat{V}(X_{t+l}) - \hat{V}(X_t) | X_t = x_i] = 0.$$

Let us define

$$\delta_{t+1}^l := G_t^{t+l} + \gamma^l \hat{V}(X_{t+l}) - \hat{V}(X_t).$$

Then we can think of δ_{t+1}^l as an l-step temporal difference.

Multi-Step TD Updating Rule

$$\hat{V}_{t+1}(x_j) = \begin{cases} \hat{V}_t(x_j) + \alpha_t \delta'_{t+1} & \text{if } X_t = x_j, \\ \hat{V}_t(x_j), & \text{otherwise.} \end{cases}$$

Advantages?
Convergence Analysis?

Outline

Outline

SARSA: On-Policy Control

SARSA = State, Action, Reward, State, Action

- Once a policy π is fixed, the associated state sequence $\{X_t\}$ is a Markov process.
- So is the process $\{(X_t, U_t)\}$ (even if $\pi \in \Pi_p$).
- The associated action-value function $Q_\pi : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$ can be viewed as a "reward" on the joint Markov process$\{(X_t, U_t)\}$.
- It satisfies the recursion

$$\begin{aligned} Q_\pi(x_i, u_k) &= R(x_i, u_k) + \gamma \sum_{j=1}^{n} a_{ij}^{u_k} Q_\pi(x_j, \pi(x_j)) \text{ if } \pi \in \Pi_d \\ &= R(x_i, u_k) + \gamma E[Q_\pi(X_{t+1}, \pi(X_{t+1}))|(X_t, U_t) = (x_i, u_k)]. \end{aligned}$$

- It can be learned using a TD scheme just we can learn V_π.

Estimating Q_π for a Specific Policy

Given a sample path $\{(X_t, U_t, W_{t+1})\}_{t \geq 0}$.

- Start with some initial guess for $\hat{Q}_\pi$.
- If $\hat{Q}_\pi$ is exact, then

$$E[R_{t+1} + \gamma \hat{Q}_\pi(X_{t+1}, \pi(X_{t+1})) - \hat{Q}_\pi(X_t, U_t) | X_t, U_t] = 0.$$

- Update the guess according to

$$\hat{Q}_\pi(X_t, U_t) \leftarrow \hat{Q}_\pi(X_t, U_t) + \alpha_t [W_{t+1} + \gamma \hat{Q}_\pi(X_{t+1}, U_{t+1}) - \hat{Q}_\pi(X_t, U_t)].$$

  for the current pair (X_t, U_t); do not adjust $\hat{Q}_\pi$ for other pairs.
- Reduce α_t to zero.
- This converges to Q_π if every pair (x_i, u_k) is visited infinitely often. (This requires that $\pi(x_i | u_k) > 0$ for all (x_i, u_k).)

Converging to an Optimal Policy

We can in parallel update the policy π using an ϵ-greedy approach on the current policy, as before: Define

$$k^* = \arg\min_{u_k \in \mathcal{U}} \hat{Q}_\pi(x_i, u_k), \psi(x_i) = u_{k^*},$$

$$\phi(u_k|x_i) = \begin{cases} \frac{\epsilon}{|\mathcal{U}|} & k \neq k^* \\ \frac{\epsilon}{|\mathcal{U}|} + (1-\epsilon) & k = k^*. \end{cases}$$

All policies are functions of t.

Reduce ϵ to zero over time. Then perhaps $\phi \to \pi^*$ and $\hat{Q} \to Q^*$ as $t \to \infty$.

Outline

Reprise: The Action-Value Function

Recall the action-value function $Q : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$:

$$Q_\pi(x_i, u_k) := E_\pi \left[\sum_{t=0}^{\infty} \gamma^t R_\pi(X_t) | X_0 = x_i, U_0 = u_k \right].$$

Q satisfies the recursion

$$Q_\pi(x_i, u_k) = R(x_i, u_k) + \gamma \sum_{j=1}^{n} a_{ij}^{u_k} Q_\pi(x_j, \pi(x_j)).$$

The optimal function Q^* satisfies the recursion

$$Q^*(x_i, u_k) = R(x_i, u_k) + \gamma \sum_{j=1}^{n} a_{ij}^{u_k} \max_{w_l \in \mathcal{U}} Q^*(x_j, w_l).$$

Reprise: Action-Value to Optimal Value to Optimal Policy

Recall the Bellman equation:

$$V^*(x_i) = \max_{u \in \mathcal{U}} \left[R(x_i, u) + \gamma \sum_{j \in [n]} a_{ij}^u V^*(x_j) \right].$$

Therefore it is clear that

$$V^*(x_i) = \max_{u_k \in \mathcal{U}} Q^*(x_i, u_k).$$

This leads to a Bellman equation for Q^*:

$$Q^*(x_i, u_k) := R(x_i, u_k) + \gamma \sum_{j \in [n]} a_{ij}^{u_k} \max_{w_l \in \mathcal{U}} Q^*(x_j, w_l).$$

Once we know Q^*, it is easy to determine the optimal policy, via

$$\pi^*(x_i) = \arg \max_{u_k \in \mathcal{U}} Q^*(x_i, u_k).$$

Iterative Method to Estimate Q^*

Start with some arbitrary $\hat{Q} : \mathcal{X} \times \mathcal{U} \to \mathbb{R}$. Choose a sequence of positive step sizes $\{\alpha_t\}_{t \geq 0}$. As the time series $\{(X_t, U_t, W_{t+1})\}_{t \geq 0}$ is observed, update function $\hat{Q}$ as follows:

$$\hat{Q}(X_t, U_t) = \hat{Q}(X_t, U_t) + \alpha_t(W_{t+1} + \gamma \max_{u_k \in \mathcal{U}} \hat{Q}(X_{t+1}, u_k) - \hat{Q}(X_t, U_t))$$

Need to maintain sufficient exploration: e.g., choose U_t to be ϵ-greedy w.r.t. current $\hat{Q}$

Convergence Result

Note: The Reward R is permitted to be random.

Theorem

Suppose each state-action pair is visited infinitely often and the Robbins-Monro step size conditions

$$\sum_{n=0}^{\infty} \alpha_n = \infty, \sum_{n=0}^{\infty} \alpha_n^2 < \infty.$$

are satisfied. If there exists a finite constant R_M such that $|R(X_t, U_t)| \leq R_M$, then

$$Q_t(x_i, u_k) \to Q^*(x_i, u_k) \text{ as } t \to \infty, \ \forall x_i \in \mathcal{X}, u_k \in \mathcal{U}, \ \text{a.s.}$$

References

V. S. Borkar. Stochastic Approximation: A Dynamical Systems Viewpoint. Springer-Verlag, 2008

C. Szepesvári. Algorithms for Reinforcement Learning. Morgan and Claypool, 2010.

R. S. Sutton and A. G. Barto. Reinforcement Learning: An Introduction (Second Edition). MIT Press, 2018.

C. J. C. H. Watkins and P. Dayan. Q-learning. Machine Learning, 8(3-4):279–292, 1992.

J. Tsitsiklis. Asynchronous stochastic approximation and Q-learning, 1994.

T. Jaakkola, M. I. Jordan, and S. P. Singh. On the convergence of stochastic iterative dynamic programming algorithms. Neural Computation, 6, 1994.

V. S. Borkar and S. P. Meyn. The O.D.E. method for convergence of stochstic approximation and reinforcement learning. Siam J. Control, 38 (2):447–69, 2000.