

# Foundations of Reinforcement Learning

Linear function approximation

Yuejie Chi

Department of Electrical and Computer Engineering

**Carnegie Mellon University**

Spring 2023

# Outline

---

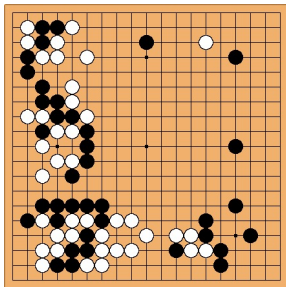
Linear function approximation

Convergence of TD(0) with linear function approximation

Extensions

# Humungous state space

---



$$S \approx 2 \cdot 10^{170}$$

Exploiting **low-complexity model** is essential for efficient RL!

- Save computation/space/data
- Generalize across state or states and actions

# Function approximation

---

## Function approximation

The *object of interest* possesses some *low-dimensional representation*.

- value function
- Q-function
- transition kernel
- reward
- policy
- Parametric:
  - Linear combinations of features
  - Neural networks
- Nonparametric:
  - Decision trees
  - Nearest neighbors

Today we will focus on *differentiable* function approximation and apply first-order methods to *incrementally* update the underlying parameters for *on-policy* evaluation.

# **Policy evaluation with linear function approximation**

# Linear V/Q function approximation

---

The **value/Q** function is a *linear* combination of features:

$$V(s; w) = \phi(s)^\top w,$$

$$Q(s, a; w) = \psi(s, a)^\top v,$$

where

- $\phi(s)$  maps the state space  $\mathcal{S}$  to  $\mathbb{R}^{d_1}$ ;
- $\psi(s, a)$  maps the state-action space  $\mathcal{S} \times \mathcal{A}$  to  $\mathbb{R}^{d_2}$ ;
- $w, v$  are the *low-dimensional embeddings* we wish to learn.

Typically, the number of features is much smaller than the dimension, i.e.

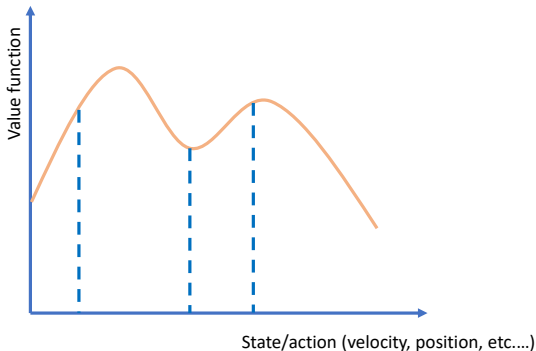
$$d_1 \ll |\mathcal{S}|, \quad d_2 \ll |\mathcal{S}||\mathcal{A}|.$$

We assume the feature maps are **known**.

# Feature selection

---

When the states/actions are numbers, function approximation for value functions bears familiarity with that of **interpolation and regression** in supervised learning.



**Example features:** polynomials, Fourier basis, radial basis functions,...

# Incremental update with **true** value

---

**Objective function:**

$$J(w) = \frac{1}{2} \mathbb{E}_{s \sim d^\pi} \left[ \underbrace{(V^\pi(s) - V(s; w))^2}_{=: J(s; w)} \right] = \frac{1}{2} \mathbb{E}_{s \sim d^\pi} \left[ (V^\pi(s) - \phi(s)^\top w)^2 \right],$$

which is *quadratic* w.r.t.  $w$ .

- Given access to  $V^\pi(s)$ , the stochastic gradient is evaluated as

$$\nabla_w J(s; w) = - \underbrace{(V^\pi(s) - \phi(s)^\top w)}_{\text{approx. error}} \phi(s).$$

- Update the weight  $w$  via

$$w \leftarrow w - \alpha \nabla_w J(s; w) = w + \alpha (V^\pi(s) - \phi(s)^\top w) \phi(s),$$

where  $\alpha$  is the learning rate.



# Incremental update with **target** value

---

However, in reality, we do not have access to  $V^\pi(s)$  (otherwise we won't need to learn!).

**Instead:** replace  $V^\pi(s)$  by its **target**  $V_{\text{target}}^\pi(s)$  from MC or TD.

$$J(w) = \frac{1}{2} \mathbb{E}_{s \sim d^\pi} \left[ \underbrace{\left( V_{\text{target}}^\pi(s) - \phi(s)^\top w \right)^2}_{=: J(s;w)} \right],$$

Update the weight  $w$  via

$$w \leftarrow w - \alpha \nabla_w J(s; w) = w + \alpha \left( V_{\text{target}}^\pi(s) - \phi(s)^\top w \right) \phi(s),$$

where  $\alpha$  is the learning rate.

# Examples of different targets

---

- In MC, use the return  $G_t$

$$w \leftarrow w + \alpha (G_t - \phi(s)^\top w) \phi(s)$$

- In TD(0), use the TD target  $r_t + \gamma V(s_{t+1}, w) = r_t + \gamma \phi(s_{t+1})^\top w$

$$w \leftarrow w + \alpha (r_t + \gamma \phi(s_{t+1})^\top w - \phi(s_t)^\top w) \phi(s_t)$$

- In TD( $\lambda$ ), use the  $\lambda$ -return  $G_t^\lambda$

$$w \leftarrow w + \alpha (G_t^\lambda - \phi(s)^\top w) \phi(s)$$

These are “**semi-gradient**” methods, since we only consider the gradient of the function approximator, not the target.

# Convergence of TD(0) with linear function approximation

# TD(0) with linear function approximation

---

Suppose we collect a trajectory following policy  $\pi$ :

$$s_0, r_0, s_1, r_1, s_2, r_2, \dots$$

**TD(0) on a single trajectory:**

$$w_{t+1} \leftarrow w_t + \alpha_t (r_t + \gamma \phi(s_{t+1})^\top w_t - \phi(s_t)^\top w_t) \phi(s_t)$$

## Question

Does TD(0) converge on a single trajectory, and if so, what does it converge to? How does the choice of feature vectors impact performance?

# Matrix representation

---

**Feature matrix and reward vector:** suppose  $\mathcal{S}$  is finite with size  $S$ ,

$$\Phi = [\phi(1), \phi(2), \dots, \phi(S)]^\top = \begin{bmatrix} \phi(1)^\top \\ \phi(2)^\top \\ \vdots \\ \phi(S)^\top \end{bmatrix} \in \mathbb{R}^{S \times d}, \quad r = \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(S) \end{bmatrix} \in \mathbb{R}^S,$$

where we assume  $\Phi$  is of **full column rank**.

**Value function approximation:** the value function is approximated as

$$V_w = \Phi w = [\phi(1), \phi(2), \dots, \phi(S)]^\top w \in \text{span}(\Phi).$$

**Assumptions:** the feature maps are bounded:

$$\|\phi(s)\|_2 \leq 1 \quad \forall s \in \mathcal{S}$$

# The TD(0) update rule in matrix form

---

The update rule of TD(0) can be written as

$$w_{t+1} = w_t - \alpha_t(A_t w_t - b_t)$$

where

$$A_t = \phi(s_t)(\phi(s_t) - \gamma\phi(s_{t+1}))^\top \in \mathbb{R}^{d \times d},$$

$$b_t = \phi(s_t)r_t \in \mathbb{R}^d.$$

## Question

What is the fixed point of TD(0)?

**Intuition:** If we let  $w_{t+1} = w_t$ , then TD(0) should approximately solve the “average” version of this equation:

$$A_t w_t \approx b_t.$$

# Stochastic approximation view of TD(0)

---

**State stationary distribution  $\mu$  of the Markov chain:**

$$D_\mu = \begin{bmatrix} \mu(1) & & & \\ & \mu(2) & & \\ & & \ddots & \\ & & & \mu(S) \end{bmatrix},$$

and  $\mu(s) > 0$  for all  $s \in \mathcal{S}$ .

**Population version:** averaging  $A_t$  and  $b_t$  over  $\mu$ :

$$\mathbb{E}_{s_t \sim \mu}[A_t] = \Phi^\top D_\mu (I - \gamma P^\pi) \Phi := A \in \mathbb{R}^{d \times d}$$

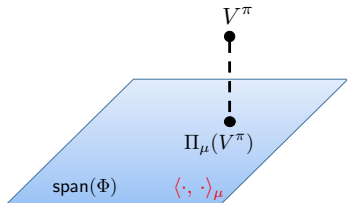
$$\mathbb{E}_{s_t \sim \mu}[b_t] = \Phi^\top D_\mu r := b \in \mathbb{R}^d$$

TD(0) applies stochastic approximation to solve the linear system of equations:

$$Aw = b.$$

— *but what is this, really?*

# Best linear function approximation to $V^\pi$ ?



A natural projection criteria is

$$\begin{aligned}\Pi_\mu(V^\pi) &= \arg \min_{z=\Phi w} \sum_{s \in \mathcal{S}} \mu(s) (V^\pi(s) - \phi(s)^\top w)^2 \\ &= \arg \min_{z=\Phi w} \|V^\pi - \Phi w\|_\mu^2,\end{aligned}$$

where we weigh the importance of different states by  $\mu$ .

- The solution is

$$\Pi_\mu(V^\pi) = \Phi \underbrace{(\Phi^\top D_\mu \Phi)^{-1}}_{=:\Sigma} \Phi^\top D_\mu V^\pi,$$

where  $\Sigma$  is the **covariance** w.r.t. the features weighted by  $\mu$ :

$$\Sigma = \Phi^\top D_\mu \Phi = \mathbb{E}_{s \sim \mu} [\phi(s)\phi(s)^\top].$$



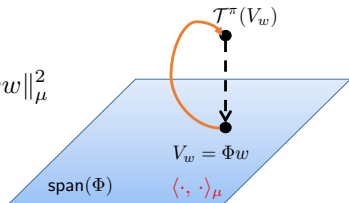
# Projected Bellman equation

In TD(0), the target is the one-step look-ahead of  $V_w = \Phi w$ :

$$\mathcal{T}^\pi(\Phi w) = r + \gamma P^\pi \Phi w$$

Project this back to  $\text{span}(\Phi)$ :

$$\min_w \|\mathcal{T}^\pi(\Phi w) - \Phi w\|_\mu^2$$



## Projected Bellman equation

$$\Phi w = \Pi_\mu \mathcal{T}(\Phi w)$$

where  $\Pi_\mu(v) = \operatorname{argmin}_{z \in \Phi w} \|z - v\|_\mu^2$ .

# Fixed-point of projected Bellman equation

---

The fixed-point of projected Bellman equation satisfies:

$$w = (\Phi^\top D_\mu \Phi)^{-1} \Phi^\top D_\mu (r + \gamma P^\pi \Phi w),$$

$$\Leftrightarrow$$

$$(\Phi^\top D_\mu \Phi) w = \Phi^\top D_\mu (r + \gamma P^\pi \Phi w)$$

$$\Leftrightarrow$$

$$\underbrace{\Phi^\top D_\mu (I - \gamma P^\pi) \Phi}_{=: A} w = \underbrace{\Phi^\top D_\mu r}_{=: b}$$

— TD(0) applies stochastic approximation to solve this!

# Asymptotic convergence

## Theorem 1 ([Tsitsiklis and Van Roy, 1997])

*TD converges to the fixed point  $w^*$  of the projected Bellman equation*

$$\Phi w = \Pi_{\mu} \mathcal{T}(\Phi w)$$

where  $\Pi_{\mu}(v) = \operatorname{argmin}_{z \in \Phi w} \|z - v\|_{\mu}^2$ , as long

$$\sum_{t=0}^{\infty} \alpha_t = \infty \quad \text{and} \quad \sum_{t=0}^{\infty} \alpha_t^2 < \infty.$$

*In addition,*

$$\underbrace{\|V_{w^*} - V^{\pi}\|_{\mu}}_{\text{TD error}} \leq \frac{1}{(1 - \gamma)} \underbrace{\|\Pi_{\mu} V^{\pi} - V^{\pi}\|_{\mu}}_{\text{approx. error}}.$$

- asymptotic convergence
- approximation error

# Proof

---

$$\underbrace{\|V_{w^*} - V^\pi\|_\mu}_{\text{TD error}} \leq \frac{1}{(1-\gamma)} \underbrace{\|\Pi_\mu V^\pi - V^\pi\|_\mu}_{\text{approx. error}}.$$

**Proof:**

$$\begin{aligned} & \|\Phi w^* - V^\pi\|_\mu \\ & \leq \|\Phi w^* - \Pi_\mu V^\pi\|_\mu + \|\Pi_\mu V^\pi - V^\pi\|_\mu && \text{(triangle inequality)} \\ & \leq \|\Pi_\mu \mathcal{T}(\Phi w^*) - \Pi_\mu V^\pi\|_\mu + \|\Pi_\mu V^\pi - V^\pi\|_\mu && \text{(fixed point)} \\ & \leq \|\mathcal{T}(\Phi w^*) - V^\pi\|_\mu + \|\Pi_\mu V^\pi - V^\pi\|_\mu && \text{(nonexpansiveness of } \Pi_\mu) \\ & \leq \|\mathcal{T}(\Phi w^*) - \mathcal{T}V^\pi\|_\mu + \|\Pi_\mu V^\pi - V^\pi\|_\mu && \text{(Bellman equation)} \\ & \leq \gamma \|\Phi w^* - V^\pi\|_\mu + \|\Pi_\mu V^\pi - V^\pi\|_\mu && \text{(Bellman contraction)} \end{aligned}$$

# Finite-time convergence of TD(0) with LFA

Polyak-Ruppert averaging: 
$$\bar{w}_T = \frac{1}{T} \sum_{i=1}^T w_i$$

## Theorem 2

Under i.i.d. data, consider any  $0 \leq \delta \leq 1$ , and  $0 < \epsilon \leq \max\{1, \|w^*\|_\Sigma\}$ . There exists some universal constant  $C > 0$  such that

$$\|\bar{w}_T - w^*\|_\Sigma \leq \epsilon$$

with probability at least  $1 - \delta$ , provided that the sample size exceeds

$$T \geq C \frac{(\max_s \phi(s)^\top \Sigma^{-1} \phi(s))(1 + \|w^*\|_\Sigma^2) \log(d/\delta)}{(1 - \gamma)^2 \epsilon^2}.$$

**Interpretation:** when  $\|w^*\|_\Sigma^2 \geq 1$ ,  $\epsilon$ -accuracy as soon as

$$T \gtrsim \frac{\kappa \|w^*\|_2^2}{(1 - \gamma)^2 \epsilon^2}, \quad \text{where } \kappa = \frac{\lambda_{\max}(\Sigma)}{\lambda_{\min}(\Sigma)}.$$

# Extensions

# Applying TD(0) to on-policy control

---

## SARSA with linear function approximation:

- Approximate the *on-policy* Q-function with

$$Q(s, a; w) = \psi(s, a)^\top v,$$

- **Policy evaluation:** apply TD(0) to update the weight

$$v_{t+1} \leftarrow v_t + \alpha (r_t + \gamma \psi(s_{t+1}, a_{t+1})^\top v_t - \psi(s_t, a_t)^\top v_t) \psi(s_t, a_t)$$

- **Policy improvement:**  $\epsilon$ -greedy policy improvement.

# Least-squares TD (LSTD)

---

[Bradtke and Barto, 1996]: Given a collection of training data

$$(s_t, V^\pi(s_t)), \quad t = 0, \dots, T - 1$$

We can also instead minimize the **batch** loss:

$$J(w) = \sum_{t=0}^{T-1} (V^\pi(s_t) - \phi(s_t)^\top w)^2$$

with the hope this leads to estimates with lower variance.

**Step 1:** Setting  $\nabla_w J(w) = 0$ , we have

$$\sum_{t=0}^{T-1} (V^\pi(s_t) - \phi(s_t)^\top w) \phi(s_t) = 0$$



# Solution to LSTD

---

**Step 2:** Replace  $V^\pi(s_t)$  by the target  $r_t + \gamma\phi(s_{t+1})^\top w$  to obtain

$$\sum_{t=0}^{T-1} (r_t + \gamma\phi(s_{t+1})^\top w - \phi(s_t)^\top w) \phi(s_t) = 0.$$

$$\implies w = \left( \sum_{t=0}^{T-1} \phi(s_t)(\phi(s_t) - \gamma\phi(s_{t+1}))^\top \right)^{-1} \left( \sum_{t=0}^{T-1} r_t \phi(s_t) \right).$$

- Recall

$$A_t = \phi(s_t)(\phi(s_t) - \gamma\phi(s_{t+1}))^\top \in \mathbb{R}^{d \times d}, \quad b_t = \phi(s_t)r_t \in \mathbb{R}^d.$$

- Aggregate the stochastic equations

$$\hat{A} = \sum_{t=0}^{T-1} A_t = \sum_{t=0}^{T-1} \phi(s_t)(\phi(s_t) - \gamma\phi(s_{t+1}))^\top$$

$$\hat{b} = \sum_{t=0}^{T-1} b_t = \sum_{t=0}^{T-1} \phi(s_t)r(s_t) \quad \implies \quad w = \hat{A}^{-1}\hat{b}$$

# LSTD vs TD

---

$$\hat{A} = \sum_{t=0}^{T-1} A_t = \sum_{t=0}^{T-1} \phi(s_t)(\phi(s_t) - \gamma\phi(s_{t+1}))^\top$$

$$\hat{b} = \sum_{t=0}^{T-1} b_t = \sum_{t=0}^{T-1} \phi(s_t)r_t \quad \implies \quad w = \hat{A}^{-1}\hat{b}$$

- Can be solved by direct calculation, or TD with experience replay.
- More sample efficient than TD.
- Computationally more expensive than TD, but can be made relatively efficient via recursive calculation by applying rank-one updates.

# Beyond linear function approximation

---

## Objective function:

$$J(w) = \frac{1}{2} \mathbb{E}_{s \sim d^\pi} \left[ \underbrace{(V^\pi(s) - V(s; w))^2}_{=: J(s; w)} \right],$$

where  $V(s; w)$  is a differentiable function approximator.

- Given access to  $V^\pi(s)$ , the stochastic gradient is evaluated as

$$\nabla_w J(s; w) = \underbrace{(V^\pi(s) - V(s; w))}_{\text{approx. error}} \nabla_w V(s; w).$$

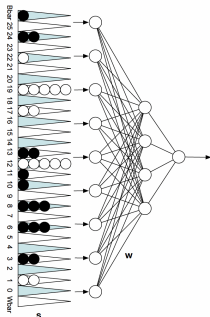
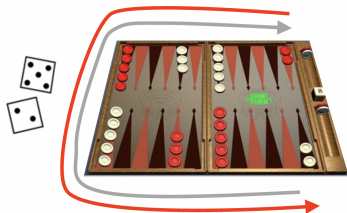
- Update the weight  $w$  via

$$w \leftarrow w - \alpha \nabla_w J(s; w) = w + \alpha (V^\pi(s) - V(s; w)) \nabla_w V(s; w),$$

where  $\alpha$  is the learning rate.

# TD-Gammon

— [Tesauro, 1995]



- Value network: three-layer neural network
- Self-play: millions of games played against itself
- Beat the best human player of backgammon at the time

# References I

---



Bradtke, S. J. and Barto, A. G. (1996).

Linear least-squares algorithms for temporal difference learning.  
*Machine learning*, 22(1-3):33–57.



Tesauro, G. (1995).

Temporal difference learning and td-gammon.  
*Communications of the ACM*, 38(3):58–68.



Tsitsiklis, J. and Van Roy, B. (1997).

An analysis of temporal-difference learning with function approximation.  
*IEEE Transactions on Automatic Control*, 42(5):674–690.