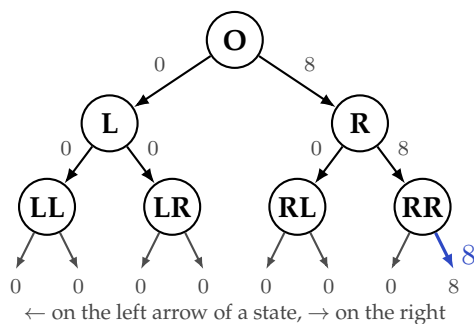


Lecture 14: Function Approximation II: Control and the Deadly Triad

How to use these notes. Board content in board order: figures, equations, tables, one-line statements. Today's task is *control when the table is too big*: the learner chooses the actions, keeps d numbers for all the pairs, and bootstraps; the questions are where the update stops and when it does not stop at all. Policy evaluation with features enters as a stated fact (one gray box); everything else is defined where it is first used, so the note stands on its own. Reading: [Sutton and Barto \(2018, Sections 10.1 and 11.1–11.3\)](#); optional, [Ernst et al. \(2005\)](#) for fitted Q-iteration. Throughout the course, the notes draw on the public materials of Stanford CS 234 ([Brunskill, 2026](#)), UIUC CS 443 ([Jiang, 2026](#)), and Cornell CS 4789 ([Sun, 2025](#)); some examples are adapted from them.

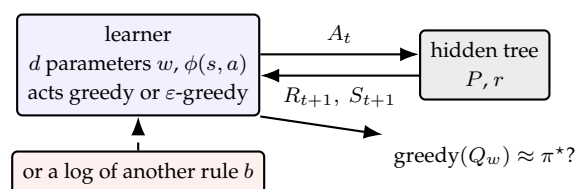
Control With Features: the Learner Chooses

The prize tree, seven states in three layers (O; L, R; LL, LR, RL, RR), the choices now the learner's: at every state left or right, the move is sure, and the last step right from RR pays 8; $\gamma = 1$. Wanted: the best rule, from d numbers for fourteen pairs.



The control tree. On each arrow, in gray, $Q^*(s, a)$, hidden from the learner: 8 along the prize path, 0 off it.

$V^* = (8; 0, 8; 0, 0, 0, 8)$; π^* : right, right, right.



Control with features: the learner acts, or reads a log; the tree answers; d numbers stand for every pair; P and r are never seen.

When the learner chooses the actions and keeps d numbers for the pairs: how does the bootstrapped update move them, where does it stop, and when does it fail to stop at all?

Policy evaluation with features (stated) (Sutton and Barto, 2018, Sections 9.3–9.4).

$$V_w(s) = w^\top \phi(s); \quad w \leftarrow w + \alpha (\text{target} - V_w(S_t)) \nabla_w V_w(S_t)$$

the template; target G_t : gradient Monte Carlo; target $R_{t+1} + \gamma V_w(S_{t+1})$: semi-gradient TD(0)

a fair coin, one number for the bottom layer: $V_{w_{\text{LS}}} = (1; 0, 2; 1, 1, 1, 1)$

the prize tree: where the returns stop, the best fit; $\overline{VE} = \sum_s \mu(s)(V^\pi - V_w)^2 = 3$

$$V_{w_{\text{TD}}} = (1; 1, 1; 1, 1, 1, 1)$$

where TD(0) stops, its fixed point, $\overline{VE}$ 4: the bootstrapped target reads the shared number

$S_t \sim \mu$ of π itself (on-policy), V_w linear $\Rightarrow w_t \rightarrow w_{\text{TD}}$

the theorem, two conditions; control removes the first: the data come from another rule

Today's targets.

1. Control: $Q_w(s, a) = w^\top \phi(s, a)$; semi-gradient Q-learning is the template with the max; two episodes by hand: a left turn at L priced 2 by the prize behind RR.
2. Fitted Q-iteration: freeze the targets, regress, repeat; three rounds on six transitions: the fit ties O (4 against 4), the table finds the prize path; in proportion, a tie at every choice.
3. The deadly triad: features, bootstrapping, off-policy; two states by hand, $w \rightarrow 1.4w$; Baird's seven states by computer, $12 \rightarrow 677$, in a reading box.

The data, the template, and what is wanted.

$$S_0 = O, A_0, R_1, S_1, A_1, R_2, S_2, A_2, R_3, \text{end}; \quad A_t \sim b(\cdot \mid S_t)$$

one episode; b is the behavior, the rule that drew the actions: the learner's own, or a log's

$$Q^*(s, a) = r(s, a) + \gamma \sum_{s'} P(s' \mid s, a) \max_{a'} Q^*(s', a'), \quad \pi^*(s) = \operatorname{argmax}_a Q^*(s, a)$$

what is wanted (stated); on the tree Q^* is written on the arrows

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha (\text{target} - Q(S_t, A_t)), \quad \text{target} = R_{t+1} + \gamma \max_a Q(S_{t+1}, a)$$

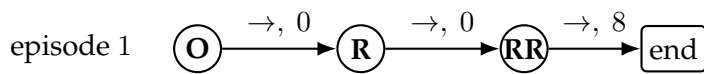
Q-learning on a table (stated): the optimality backup sampled; SARSA: $R_{t+1} + \gamma Q(S_{t+1}, A_{t+1})$

the hand runs: $b =$ the coin, left or right w.p. $\frac{1}{2}$; $\mu(s, a) = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ by layer

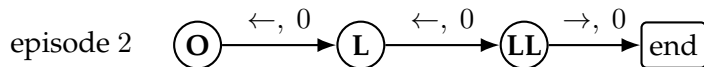
two episodes, six transitions, below; $\mu(s, a)$: the probability of taking the pair; each layer sums to 1

today: d numbers for 14 pairs; known: transitions, γ, ϕ ; hidden: P, r, Q^*

the table keeps 14; the learner keeps d and reads its rule off them



above each arrow: the action, then the reward



Two episodes of the coin, used by hand throughout: the prize, then a left turn and nothing. Six transitions $(S_t, A_t, R_{t+1}, S_{t+1})$: the record of the hand runs.

Q with features (Sutton and Barto, 2018, Section 10.1).

$$Q_w(s, a) := w^\top \phi(s, a), \quad \nabla_w Q_w(s, a) = \phi(s, a)$$

the fit on pairs and its gradient; any differentiable Q_w , a network included, supplies both

$$\phi(s, a) = e_{(s,a)} : \quad Q_w(s, a) = w_{(s,a)}$$

one-hot: the table, 14 numbers on the tree

$$\phi(s, a) = e_{(g(s),a)} : \quad Q_w(s, a) = w_{g(s),a}$$

a group of states shares one number per action; the gradient is the cell's indicator

the layer class: one cell $w_{g,a}$ per group $g \in \{O, L, R, 2\}$ and action a ; $d = 8$

O, L, R alone, the bottom layer shares its two cells ($w_{2\leftarrow}, w_{2\rightarrow}$): the grid below

$$\text{greedy w.r.t. } Q_w : \quad \arg\max_a Q_w(s, a)$$

$|A|$ evaluations of the fit, no model: a lookup, as with a table

Example: is Q^* in the class?

layer class: $Q^*(LL, \rightarrow) = 0$ and $Q^*(RR, \rightarrow) = 8$ must share $w_{2\rightarrow}$ no

the greedy policy of Q^* is in it (right everywhere); Q^* itself is not

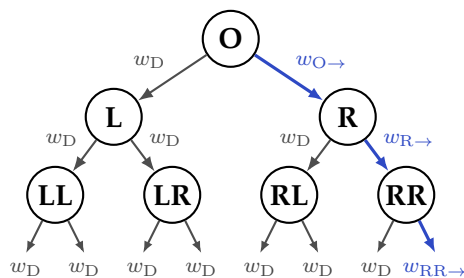
the dead-end class on pairs: $(O, \rightarrow), (R, \rightarrow), (RR, \rightarrow)$ alone, w_D for the other eleven
four numbers: the three steps of the prize path, and one for everything off it

Q^* at $w = (8, 8, 8, 0)$ yes

so the stop of any method is the truth there, as on a table

the layer class	$\leftarrow$	$\rightarrow$
O	$w_{O\leftarrow}$	$w_{O\rightarrow}$
L	$w_{L\leftarrow}$	$w_{L\rightarrow}$
R	$w_{R\leftarrow}$	$w_{R\rightarrow}$
LL, LR, RL, RR	$w_{2\leftarrow}$	$w_{2\rightarrow}$

The grid of the layer class: eight cells; the bottom row (blue) serves four states. One-hot would have seven rows.



The dead-end class on pairs: the three steps of the prize path alone (blue), one number w_D for the eleven arrows that leave it; Q^* is in it.

Semi-gradient Q-learning (Watkins, 1989), (Sutton and Barto, 2018, Sections 10.1 and 11.1).

- Input: transitions $(S_t, A_t, R_{t+1}, S_{t+1})$ drawn by any behavior b ; the features ϕ ; a step size α . Output: w , read as the policy $\arg\max_a Q_w(s, a)$.
- Start: $w_0 = 0$; $Q_w(\text{end}, \cdot) = 0$.
- Each step t :

$$y_t = R_{t+1} + \gamma \max_a Q_{w_t}(S_{t+1}, a), \quad \delta_t = y_t - Q_{w_t}(S_t, A_t) \quad \text{target and TD error}$$

$$\ell_t(w) = \frac{1}{2} (Q_w(S_t, A_t) - y_t)^2 \quad \text{hold } y_t \text{ fixed when differentiating}$$

$$w_{t+1} = w_t - \alpha \nabla_w \ell_t(w_t) = w_t + \alpha \delta_t \nabla_w Q_w(S_t, A_t) \Big|_{w=w_t} \quad \text{chain rule}$$

$$w_{t+1} = w_t + \alpha \delta_t \phi(S_t, A_t); \quad \text{a group: } w_{g,a} \leftarrow w_{g,a} + \alpha \delta_t$$

with features; by hand one cell moves

- SARSA with features: the target $R_{t+1} + \gamma Q_w(S_{t+1}, A_{t+1})$, A_{t+1} the behavior's next action (on-policy); Expected SARSA: $\sum_a \pi(a | S_{t+1}) Q_w(S_{t+1}, a)$.
- One-hot: Q-learning on the table. Memory d ; $|A|$ evaluations of the fit a step.

Off-policy by construction: b decides which pair is updated, the max prices the greedy rule. *Semi*: differentiate the prediction while holding y_t fixed.

Example: semi-gradient Q-learning by hand. $\alpha = \frac{1}{2}$, $\gamma = 1$, $w = 0$, the layer class, the six transitions; $\delta_t = R_{t+1} + \max_a Q_w(S_{t+1}, a) - Q_w(S_t, A_t)$.

$$t = 0, (O, \rightarrow, 0, R) : \quad \delta_0 = 0 + \max(w_{R\leftarrow}, w_{R\rightarrow}) - w_{O\rightarrow} = 0$$

nothing moves: every cell reads 0

$$t = 1, (R, \rightarrow, 0, RR) : \quad \delta_1 = 0 + \max(w_{2\leftarrow}, w_{2\rightarrow}) - w_{R\rightarrow} = 0$$

$$t = 2, (RR, \rightarrow, 8, \text{end}) : \quad \delta_2 = 8 - w_{2\rightarrow} = 8, \quad w_{2\rightarrow} \leftarrow 0 + \frac{1}{2} \cdot 8 = 4$$

right reads 4 at all four bottom states; the table: at RR only

$$t = 3, (O, \leftarrow, 0, L) : \quad \delta_3 = 0 + \max(w_{L\leftarrow}, w_{L\rightarrow}) - w_{O\leftarrow} = 0$$

$$t = 4, (L, \leftarrow, 0, LL) : \quad \delta_4 = 0 + \max(w_{2\leftarrow}, w_{2\rightarrow}) - w_{L\leftarrow} = \max(0, 4) - 0 = 4$$

the max reads the shared 4: a left turn priced by the prize behind RR

$$w_{L\leftarrow} \leftarrow 0 + \frac{1}{2} \cdot 4 = 2 \quad \text{the table's target read 0; its } (L, \leftarrow) \text{ stays 0}$$

$$t = 5, (LL, \rightarrow, 0, \text{end}) : \quad \delta_5 = 0 - w_{2\rightarrow} = -4, \quad w_{2\rightarrow} \leftarrow 4 + \frac{1}{2}(-4) = 2$$

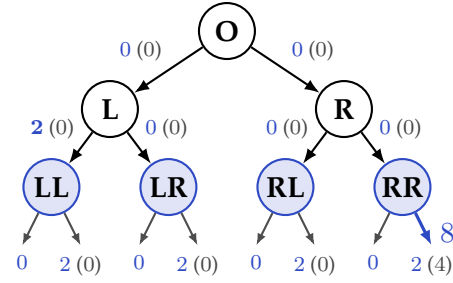
LL pulls the shared cell down, RR with it

after two episodes: $Q_w = (0, 0; 2, 0; 0, 0; 0, 2)$; the table: $Q(RR, \rightarrow) = 4$ only

the fit prefers left at L, right at the bottom

after two episodes	←	→
O	0	0
L	2	0
R	0	0
LL, LR, RL, RR	0	2

The grid after two episodes: two cells moved, (L, ←) and the shared (2, →).



On the arrows: the fit in blue, the table in gray. One visit to RR priced “right” at every bottom state; the max then priced a left turn at L by it.

- The max in the target reads shared numbers: a left turn at L was priced by the prize behind RR, a state the left turn cannot reach.
- Where the numbers are pulled in the long run: the batch version, next, which also replaces the noise of one pass by a regression.

The Batch Version: Fitted Q-Iteration

Fitted Q-iteration (FQI) (Ernst et al., 2005; Riedmiller, 2005); fitted value iteration, Gordon (1995).

- Input: a record $\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^N$ from any behavior; a class $\{Q_w\}$; rounds K . Start: $w_0 = 0$.
- Round $k = 0, 1, \dots, K - 1$:

$$y_i = r_i + \gamma \max_a Q_{w_k}(s'_i, a) \quad \text{the targets, frozen for the round: } w_k, \text{ not } w$$

$$w_{k+1} = \underset{w}{\operatorname{argmin}} \sum_{i=1}^N (Q_w(s_i, a_i) - y_i)^2$$

fit the same squared loss over the record; a group's cell: the mean target

- Output: the greedy policy of Q_{w_K} . Any class fits a round: a network, by SGD on the squared loss, the targets frozen throughout.

One-hot: the mean target at a pair is $\hat{r}(s, a) + \gamma \sum_{s'} \hat{P}(s' | s, a) \max_a Q_{w_k}(s', a)$, value iteration in the record's own model (stated): where replaying the record with Q-learning and a shrinking step heads.

Example: three rounds of FQI on the six transitions. The layer class, $\gamma = 1$, $w_0 = 0$; a cell is the mean of the targets of its transitions.

round 1 : $y_i = r_i = 0, 0, 8; 0, 0, 0 \Rightarrow w_{2 \rightarrow} = \frac{8+0}{2} = 4$, every other cell 0

(RR, $\rightarrow$) and (LL, $\rightarrow$) share the cell: 8 and 0 averaged

round 2 : (R, $\rightarrow$) $\rightarrow$ RR, (L, $\leftarrow$) $\rightarrow$ LL : $y = \max(w_{2 \leftarrow}, w_{2 \rightarrow}) = 4 \Rightarrow w_{R \rightarrow} = w_{L \leftarrow} = 4$

both read the shared 4 through the max

(O, $\rightarrow$) $\rightarrow$ R, (O, $\leftarrow$) $\rightarrow$ L : $y = \max(0, 0) = 0$

the targets read the frozen cells of round 1: still 0 at L and R

round 3 : (O, $\rightarrow$) $\rightarrow$ R : $y = \max(0, 4) = 4$; (O, $\leftarrow$) $\rightarrow$ L : $y = \max(4, 0) = 4$

the cells of round 2 at R and at L, read through the max

$\Rightarrow w_{O \leftarrow} = w_{O \rightarrow} = 4$

settled: round 4 repeats round 3; the rounds walk back from the end

	round 1		round 2		round 3		the table, round 3	
	$\leftarrow$	$\rightarrow$	$\leftarrow$	$\rightarrow$	$\leftarrow$	$\rightarrow$	$\leftarrow$	$\rightarrow$
O	0	0	0	0	4	4	0	8
L	0	0	4	0	4	0	0	0
R	0	0	0	4	0	4	0	8
bottom	0	4	0	4	0	4	$Q(\text{RR}, \rightarrow) = 8$, the rest 0	

The grid after each round (bold: the cells that moved), and the table on the same six: $Q(\text{RR}, \rightarrow) = 8$ after round 1, $Q(\text{R}, \rightarrow) = 8$ after round 2, $Q(\text{O}, \rightarrow) = 8$ after round 3, the rest 0. The fit ties O; the table finds the prize path.

Example: what the two greedy rules earn from O.

the fit: $Q_w(\text{O}, \leftarrow) = Q_w(\text{O}, \rightarrow) = 4$, a coin at the tie: $\frac{1}{2} \cdot 0 + \frac{1}{2} \cdot 8 = 4$; the table: 8

the fit priced (L, $\leftarrow$) at 4 through the shared cell: O cannot tell L from R

Example: replace the six transitions by the coin's true visit proportions. The layer class, bottom-up; $\mu(s, a) = \frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ by layer.

$$(2, \rightarrow) : \frac{1}{8} [3(0 - w_{2\rightarrow}) + (8 - w_{2\rightarrow})] = 0 \Rightarrow w_{2\rightarrow} = 2; \quad (2, \leftarrow) : w_{2\leftarrow} = 0$$

a last step pays its reward: 0, 0, 0, 8 right, 0 left

$$(L, \cdot), (R, \cdot) : \frac{1}{4} [\gamma \max(w_{2\leftarrow}, w_{2\rightarrow}) - w_{sa}] = 0 \Rightarrow 2 \text{ at all four}$$

every middle arrow lands on a bottom state reading $\max(0, 2) = 2$

$$(O, \cdot) : \frac{1}{2} [\gamma \max(2, 2) - w_{Oa}] = 0 \Rightarrow w_{O\leftarrow} = w_{O\rightarrow} = 2$$

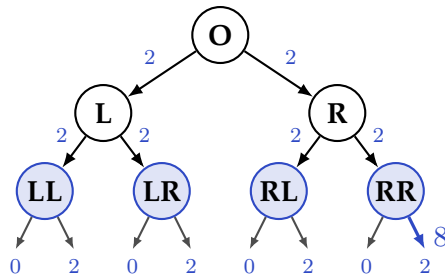
2 at every arrow above the bottom: a tie at every choice

greedy, a coin at each tie: $8 \cdot \frac{1}{4} = 2$ against $V^*(O) = 8$

Q^* : (0, 8) at O and R, (0, 0) at L: the fit cannot tell L from R

the dead-end class on pairs: $(8, 8, 8, 0) = Q^*$ in three rounds

eight numbers tie, four find the prize: the class decides, not d



ties: $2 = 2$ at O, at L, at R
the prize is found one time in four

The stop of the layer class under the coin's weighting: right at the bottom, no preference above. The shared bottom cell erased where the prize is.

- By computer, 200 runs of 2,000 episodes of the coin, $\alpha = 0.1$, from 0: semi-gradient Q-learning in the layer class ends at (2.0, 2.0; 1.9, 2.0; 2.0, 2.0; 0, 2.0) on average; the table at Q^* .
- On a tree the rounds walk back from the end and stop at the depth; on a chain with loops they iterate, and need not settle: next.

Where It Stops, and Why It May Not: the Deadly Triad

Derivation: one fitted backup, its fixed point, and convergence (Tsitsiklis and Van Roy, 1997), (Sutton and Barto, 2018, Sections 9.4 and 11.3). Use a linear class $\mathcal{F} = \{Q_w\}$ and the true transition proportions. $\|U\|_\mu^2 := \sum_{s,a} \mu(s,a) U(s,a)^2$; ΠU is the least-squares fit to a table U within this class.

$$y_i = r_i + \gamma \max_a Q_{w_k}(s'_i, a), \quad \mathbb{E}[y_i \mid s_i, a_i] =: (\mathcal{T}Q_{w_k})(s_i, a_i)$$

freeze the old Q ; its mean target at each pair defines the optimality backup $\mathcal{T}Q_{w_k}$

$$Q_{w_{k+1}} = \operatorname{argmin}_{Q \in \mathcal{F}} \|Q - \mathcal{T}Q_{w_k}\|_\mu^2 = \Pi \mathcal{T}Q_{w_k}$$

fit the mean target; frozen target noise adds a constant to the expected squared loss

$$Q_{w_{k+1}} = Q_{w_k} \iff Q_w = \Pi \mathcal{T}Q_w$$

the fit returns the same Q ; also the zero mean update of semi-gradient Q -learning (stated)

$$\|\Pi U - \Pi U'\|_\mu \leq \|U - U'\|_\mu \quad \text{linear projection cannot enlarge distances}$$

$$\|\mathcal{T}^\pi V - \mathcal{T}^\pi V'\|_\mu \leq \gamma \|V - V'\|_\mu \quad \text{for } \mu \text{ the weighting of } \pi \text{ itself, } \gamma < 1 \text{ (stated)}$$

on-policy prediction: combining the two bounds makes $\Pi \mathcal{T}^\pi$ contract; linear TD(0) converges

off-policy, by another weighting or by the max: no such line; $\Pi \mathcal{T}$ can stretch

fitting each round does not guarantee convergence across rounds: the examples below $\square$

Example: two states by hand (Sutton and Barto, 2018, Section 11.2), (Tsitsiklis and Van Roy, 1996). A: $\phi = 1$, B: $\phi = 2$, so $V_w = (w, 2w)$; A steps to B, B to B; reward 0; $\gamma = 0.9$; the truth 0 is in the class.

$$(A, 0, B) : \quad \delta = 0 + \gamma \cdot 2w - w = 0.8w, \quad w \leftarrow w + \frac{1}{2} \cdot 0.8w \cdot 1 = 1.4w$$

A pushes w away from 0: from 1, 1.4, 1.96, 2.74, 3.84

$$(B, 0, B) : \quad \delta = \gamma \cdot 2w - 2w = -0.2w, \quad w \leftarrow w + \frac{1}{2}(-0.2w) \cdot 2 = 0.8w$$

B pulls w toward 0; sampling weights decide the balance

$$\text{follow the target chain, after reaching B: } w \leftarrow 0.8w \Rightarrow w \rightarrow 0$$

repeated updates at B shrink w

$$\text{restart at A for every sample: } w \leftarrow 1.4w \Rightarrow w \rightarrow \infty \quad (w_0 > 0)$$

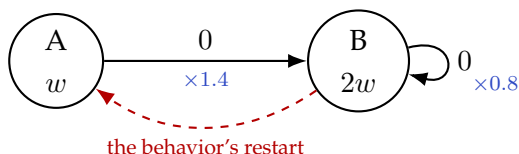
replay only A: B never supplies a shrinking update

$$\text{both states equally: } \mathbb{E}[\delta \phi(S)] = \frac{1}{2}(0.8 - 0.4)w = 0.2w > 0 \quad (w > 0)$$

A outweighs B once A is visited more than one time in three

$$\text{batch, equal weights: } w' = \operatorname{argmin}_w [(w - 2\gamma w_k)^2 + (2w - 2\gamma w_k)^2] = \frac{6}{5}\gamma w_k = 1.08w_k$$

$\Pi \mathcal{T}$ stretches by 1.08 (iff $\gamma > \frac{5}{6}$); under B alone, γw_k



the weighting of π : B only $\rightarrow$ shrinks

A one time in two $\rightarrow$ grows

$\gamma = 0.9$, $\alpha = \frac{1}{2}$, truth 0

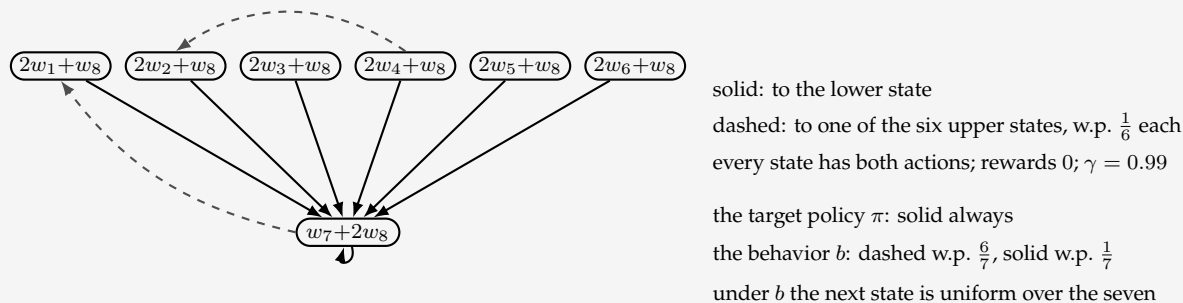
Two states, one parameter: the same update, two weightings, two fates. The dashed restart is what a behavior policy, or an iteration over all states, adds.

The deadly triad (Sutton and Barto, 2018, Section 11.3); Baird (1995); Boyan and Moore (1995). Trouble needs all three: *features* (the step moves other pairs), *bootstrapping* (the fit is in the target), and *off-policy* (the pairs updated arrive with another rule's weighting, not the target policy's).

Method	features	bootstrap	off-policy	Settles?
TD(0), Q-learning on the table		✓	✓	yes: the table, any b with coverage (stated)
gradient Monte Carlo with features	✓		✓	yes: SGD on $\overline{VE}$ under the weighting given
semi-gradient TD(0), on-policy, linear	✓	✓		yes: w_{TD} , the theorem (stated)
semi-gradient TD(0), off-policy	✓	✓	✓	can run away: two states; Baird (reading)
semi-gradient Q-learning, FQI	✓	✓	✓	off-policy by the max; can run away (stated); no theorem

Drop any one and the trouble goes. The max makes Q-learning off-policy by itself: under the learner's own ϵ -greedy rule the pairs arrive at that rule's rate and the target prices the greedy one (rarely trouble in practice, stated).

Reading (optional): Baird's counterexample, by computer (Baird, 1995), (Sutton and Barto, 2018, Section 11.2). Nothing below depends on this box. The two-state example scaled up: seven states, a known model, the truth in the class, and the off-policy weighting alone sends the fit away.



Baird's counterexample: seven states, two actions, eight parameters; the expression in a state is $V_w(s)$.

Under π every move lands on the lower state.

The setting. $V^\pi = 0$ everywhere; $w = 0$ is in the class, and the eight feature vectors are linearly independent: a favorable case. Start $w_0 = (1, 1, 1, 1, 1, 1, 10, 1)$, so $V_{w_0} = 3$ at the upper states and 12 at the lower; $\alpha = 0.01$. Three learners:

$$(i) \quad w \leftarrow w + \alpha \rho_t \delta_t \nabla_w V_w(S_t), \quad \rho_t = \frac{\pi(A_t | S_t)}{b(A_t | S_t)} = 7 \text{ (solid)}, 0 \text{ (dashed)}$$

the ratio corrects action choices; states still arrive uniformly under b

$$(ii) \quad w \leftarrow w + \frac{\alpha}{7} \sum_s (\gamma V_w(\text{lower}) - V_w(s)) \nabla_w V_w(s)$$

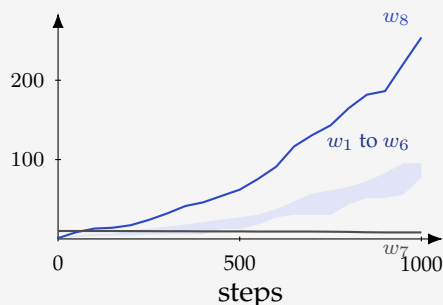
compare with (i): its exact mean update, the same state weights, no sampling noise

$$(iii) \quad w \leftarrow w + \alpha (\gamma V_w(\text{lower}) - V_w(\text{lower})) \nabla_w V_w(\text{lower})$$

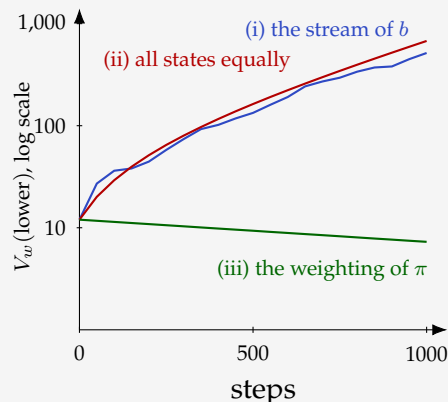
compare with (ii): change only state weights, to those of π (the lower state alone)

$V_w(\text{lower state})$ after step	0	100	300	500	1,000
(i) semi-gradient TD(0) on the stream of b , with the ratio	12	36.2	73.9	133.6	516.0
(ii) the expected step, all seven states equally	12	29.2	79.9	163.1	676.9
(iii) the expected step under the weighting of π (the lower state)	12	11.4	10.3	9.3	7.3

Truth 0. Seed 639 for the stream. The two off-policy learners grow without bound; the on-policy one shrinks the lower state's value to 0 (0.08 by step 10,000).



The eight parameters under (i), the stream of b (the band: the smallest and the largest of $w_1, \dots, w_6$).



The lower state's value under the three learners.

Nothing is random in the red curve; the weighting alone does it.

- Nothing is random or hidden in (ii): the model is known, the truth is in the class, and the iteration still runs away.
- Under the weighting of π the step lands where μ looks: the lower state goes to 0; the six upper states, never visited, end at -1.8 , invisible to μ .
- The mean step is linear, $-Aw$ with $A = \sum_s \mu(s) \phi(s) (\phi(s) - \gamma \phi(\text{lower}))^\top$; under the uniform weighting the symmetric part of A has a negative eigenvalue, -1.02 : a direction it stretches (stated).

The Map

approximate the value	approximate the policy	approximate the model
V_w, Q_w : semi-gradient TD(0), Q-learning, FQI; DQN	π_θ : policy gradient, actor-critic (a critic is V_w)	$\hat{P}, \hat{r}$: certainty equivalence; planning and search in it
fit: a backup or a return	fit: the objective $J(\pi_\theta)$ directly	fit: the transitions
stop: a projected fixed point	stop: a stationary point	stop: the record's own model

The map: three things a learner with d parameters can approximate. Today's methods sit in the left box, with DQN, the same methods with a network, named beside them; the other two boxes are named by what they are.

Summary

Function approximation for control, on one board

$$\delta_t = R_{t+1} + \gamma \max_a Q_w(S_{t+1}, a) - Q_w(S_t, A_t), \quad w \leftarrow w + \alpha \delta_t \nabla_w Q_w$$

semi-gradient Q-learning; two episodes: a left turn at L priced 2 by the prize behind RR

$$y_i = r_i + \gamma \max_a Q_w(s'_i, a), \quad w_{k+1} = \operatorname{argmin}_w \sum_i (Q_w(s_i, a_i) - y_i)^2$$

FQI: three rounds on six transitions, the fit ties O; one-hot: value iteration in the record's model

stop: $Q_w = \Pi \mathcal{T} Q_w$; eight numbers: 2 at every choice, ties; four numbers: Q^*
the class decides, not d

triad: features, bootstrapping, off-policy; two states $w \rightarrow 1.4w$; Baird $12 \rightarrow 677$
 Π never stretches; $\mathcal{T}^\pi$ shrinks under its own weighting, else $\Pi \mathcal{T}$ may stretch; Baird in the reading box

References

- Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 30–37. Morgan Kaufmann, 1995.
- Justin A. Boyan and Andrew W. Moore. Generalization in reinforcement learning: Safely approximating the value function. In *Advances in Neural Information Processing Systems 7*, pages 369–376, 1995.
- Emma Brunskill. CS 234: Reinforcement learning. Stanford University course materials, 2026. URL <https://web.stanford.edu/class/cs234/>. Winter 2026 offering.
- Damien Ernst, Pierre Geurts, and Louis Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556, 2005.
- Geoffrey J. Gordon. Stable function approximation in dynamic programming. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 261–268. Morgan Kaufmann, 1995.
- Nan Jiang. CS 443: Reinforcement learning. University of Illinois Urbana-Champaign course materials, 2026. URL <https://nanjiang.cs.illinois.edu/cs443/>. Spring 2026 offering.
- Martin Riedmiller. Neural fitted Q iteration – first experiences with a data efficient neural reinforcement learning method. In *Machine Learning: ECML 2005, Lecture Notes in Computer Science*, volume 3720, pages 317–328. Springer, 2005.
- Wen Sun. CS 4/5789: Introduction to reinforcement learning. Cornell University course

materials, 2025. URL https://wensun.github.io/CS4789_spring_2025.html. Spring 2025 offering.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 2 edition, 2018. URL <http://incompleteideas.net/book/the-book-2nd.html>.

John N. Tsitsiklis and Benjamin Van Roy. Feature-based methods for large scale dynamic programming. *Machine Learning*, 22(1–3):59–94, 1996.

John N. Tsitsiklis and Benjamin Van Roy. An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, 42(5):674–690, 1997.

Christopher J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King’s College, University of Cambridge, 1989.