Off-policy value function approximation

Konpat Preechakul
Chulalongkorn University
September 2019

Two sided problems

- Off-policy target value

$$G_t \rightarrow \rho G_t$$

- Off-policy target state distribution

$$p^b \rightarrow p^\pi$$

Target value problem

Target value problem

- If the target is on-policy, it needs correction
    - MC, N-step need importance sampling

- Some targets are off-policy, no need for correction
    - Expected SARSA
    - Deterministic policies
    - Q-learning
    - Tree-backup

A bird eye view of on/off-policy

Algorithm	V/Q value	Make it off-policy	Variance	Bias
Monte Carlo	V	IS	High	Low
	Q	IS	High	Low
One-step SARSA	V	IS	Lower	High
	Q	IS	Lower	High
One-step Expected SARSA	V	IS	Lower	High
	Q	Already	Low	High
One-step TD with Deterministic Policy (including Q-learning)	V	IS	Lower	High
	Q	Already	Low	High
N-step SARSA (including lambda)	V	IS	Medium	Medium
	Q	IS	Medium	Medium
Tree backup	V	Already	Low	High
	Q	Already	Low	High

Semi-gradient with correction

- Off-policy N-step

$$g_{t:t+n} = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots \gamma^n v(s_{t+n})$$

$$v(s_t) \leftarrow v(s_t) + \alpha \rho_{t:t+n-1} \left[G_{t:t+n} - v(s_t) \right]$$

- Off-policy N-step semi-gradient

$$\theta \leftarrow \theta + \alpha \rho_{t:t+n-1} \left(g_{t:t+n} - v_\theta(s_t) \right) \nabla_\theta v_\theta(s_t)$$

Q-learning with approximation

- No need for off-policy correction

$$q(s,a) \leftarrow$$
$$q(s,a) + \alpha\big[r(s,a) + \max_{a'} q(s',a') - q(s,a)\big]$$

for until q_θ is stable **do**
  take action according to $q_\theta(s,a)$
  collect (s,a,r,s')
  $\delta = r + \gamma \max_{a'} q_\theta(s',a') - q_\theta(s,a)$
  $\theta \leftarrow \theta + \alpha\delta\nabla_\theta q_\theta(s,a)$
end for

Summary

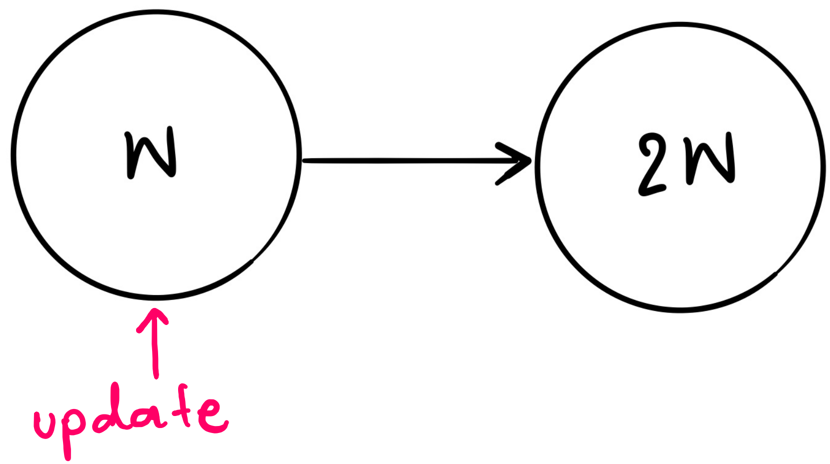
- Target value problem is straightforward
- Make sure that the target value is corrected
- Only **one part** of the problem

Target distribution problem

Target distribution problem

- Update distribution is “crucial” for convergence of semi-gradient
- Off-policy data = off-policy distribution
    - Convergence guarantee only on-policy distribution
- No convergence guarantee

Example of divergence



$w = 10$

$w \leftarrow w + \alpha(2w - w) \cdot 1$

$10 + 0.1(10) \cdot 1$

$w \leftarrow 11$

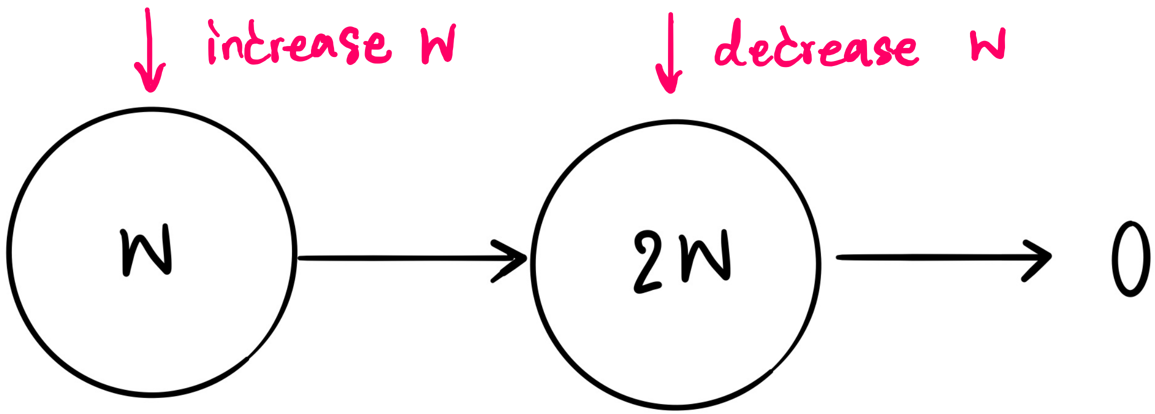
$w \leftarrow w + \alpha(2w - w) \cdot 1$

$11 + 0.1(22 - 11) \cdot 1$

$w \leftarrow 12.1$

frequent updates could diverge

On-policy is more reasonable



on-policy trajectory $\longrightarrow$

linear case : $2w \rightarrow 0$ in one-update ($\alpha = 1$)

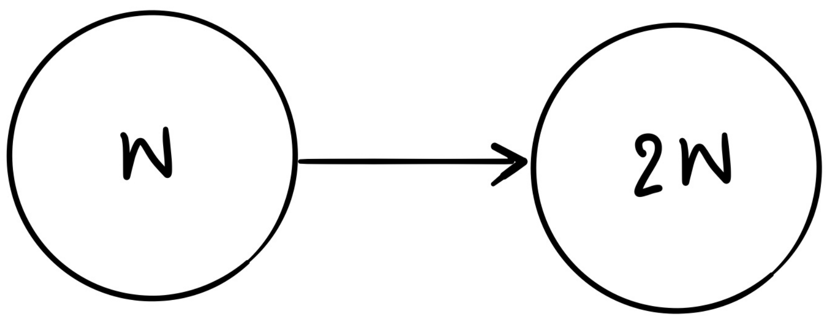
non-linear : not possible ! (needs multiple updates)

↳ on-policy + non-linear

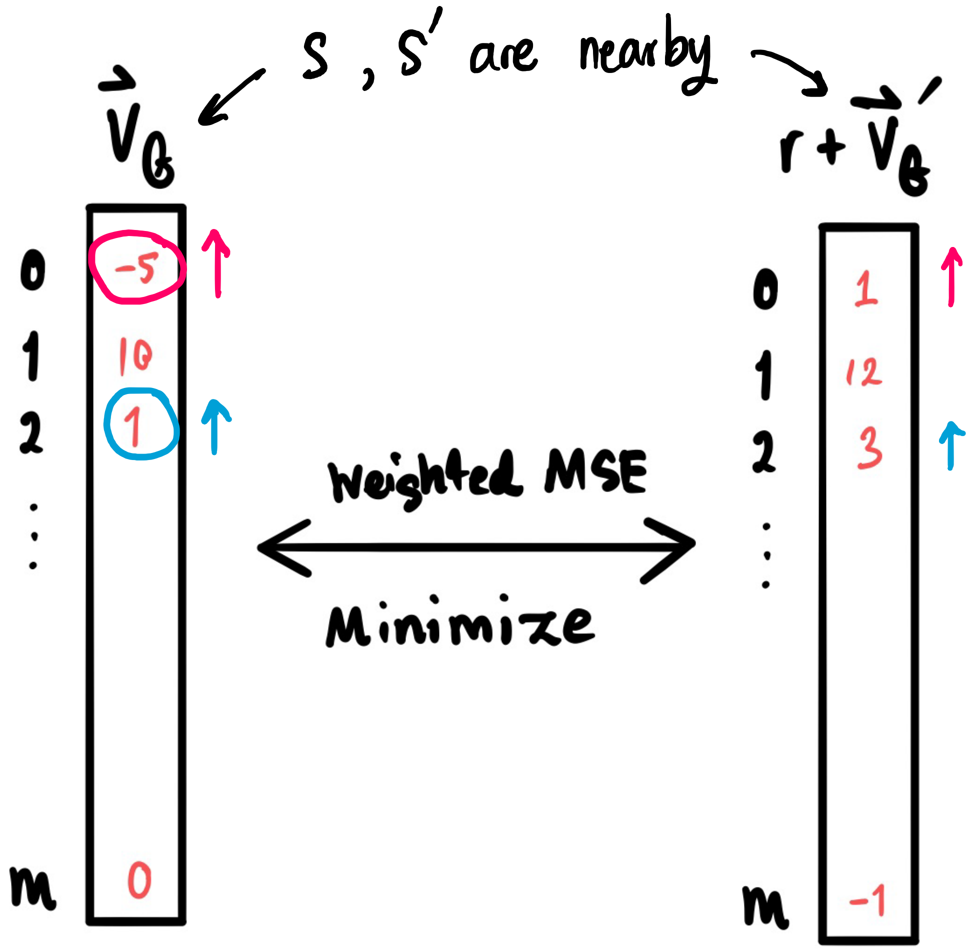
$\neq$ convergence

Intuitive divergence

- Problem of generalization of nearby states



Intuitive divergence



The deadly triad

- Off-policy
- Bootstrapping
- Approximation

Having the three at the same time causes "instability"

(with semi-gradient)

Double is not deadly

Off-policy + bootstrap:

Q-learning

Off-policy + approximation:

Off-policy MC with approx.

Bootstrap + approximation:

On-policy linear TD

All are stable.

When does divergence happen?

- When you use off-policy data
- You don't correct the target distribution
  - Even you have corrected the target value
- You use approximation
- **You use semi-gradient**
  - Imply bootstrapping

Bandages for semi-gradient

The deadly triad bandages

- Off-policy => **more on-policy**
- Bootstrapping => **less bootstrap**
- Approximation => **less approximate**

More on-policy

- On-policy state distribution correction
- Importance sampling

$$\theta \leftarrow \theta + \alpha \underbrace{\rho_{0:t-1}}_{\text{Distribution correction}} \underbrace{\rho_t}_{\text{Target correction}} \left(r_{t+1} + \gamma v_\theta(s_{t+1}) - v_\theta(s_t)\right) \nabla_\theta v_\theta(s_t)$$

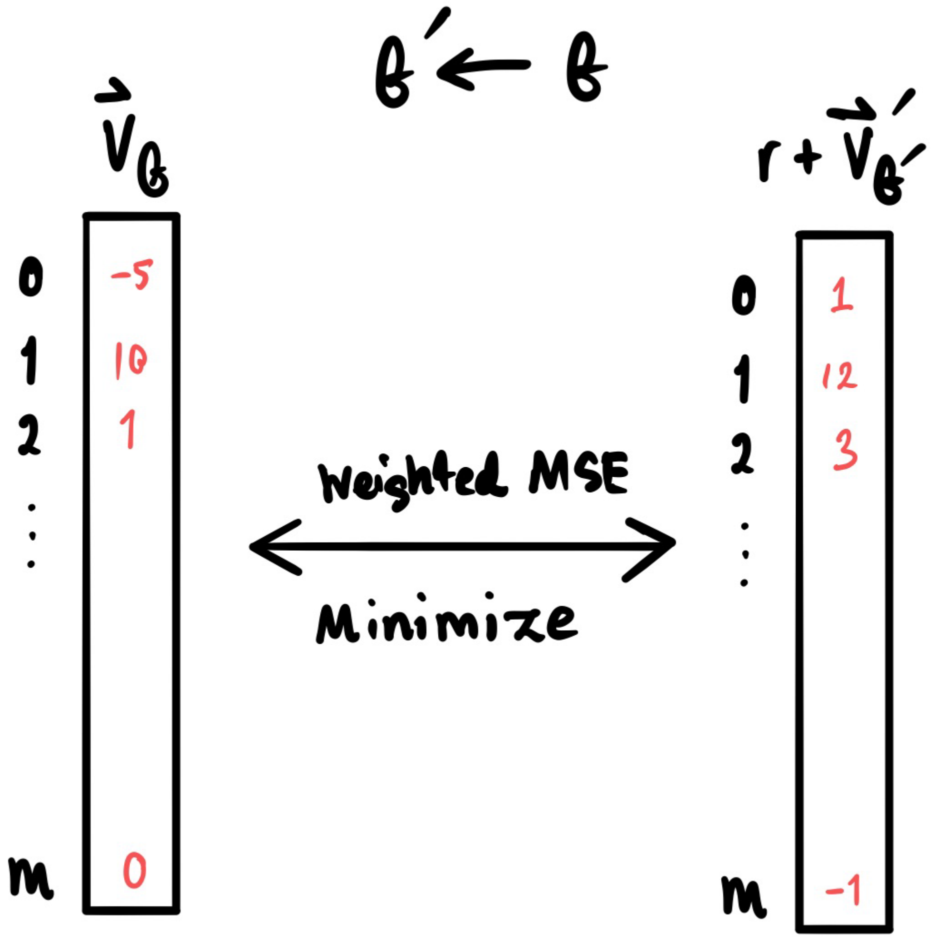
$$\rho_{t:T-1} = \prod_{i=t}^{T-1} \frac{\pi(a_i|s_i)}{b(a_i|s_i)}$$

- High variance

Less bootstrapping

- Smaller discount
  - Smaller bootstrap
- N-step return
  - Smaller bootstrap
- Target networks
  - Realize (more) the independence assumption
- Loss constraints
  - Reducing dependency

Target network visualized



Target networks

$\theta' \leftarrow \theta$

for until q_θ is stable **do**

  take action according to $q_\theta(s,a)$

  collect (s,a,r,s')

  $\delta = r + \gamma \max_{a'} q_{\theta'}(s',a') - q_\theta(s,a)$

  $\theta \leftarrow \theta + \alpha\delta\nabla_\theta q_\theta(s,a)$

  if every K steps **then**

    $\theta' \leftarrow \theta$ how long do you need to get to $r + V'_{\theta'}$

  end if

end for

Loss constraints

Reducing dependency between nearby states*

- Projected gradients

$V_\theta(s) \uparrow \qquad V_\theta(s') \uparrow$

$V_\theta(s) \uparrow \qquad V_\theta(s')$ unchanged

we use $g \approx \nabla V_\theta(s)$ s.t. $g \perp \nabla V_\theta(s')$

- Temporal consistency loss

$TC(\theta) = \| V_\theta(s') - V_{\theta'}(s') \|^2$

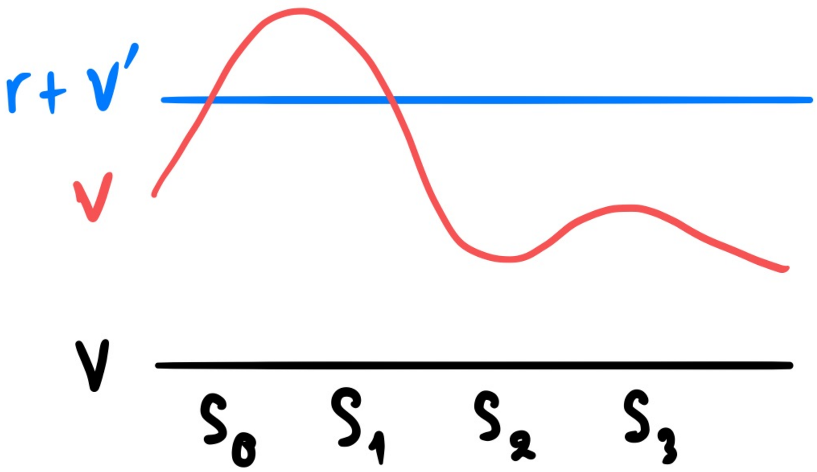
Pohlen, Tobias, Bilal Piot, Todd Hester, Mohammad Gheshlaghi Azar, Dan Horgan, David Budden, Gabriel Barth-Maron, et al. 2018. "Observe and Look Further: Achieving Consistent Performance on Atari." *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1805.11593.

Less approximate

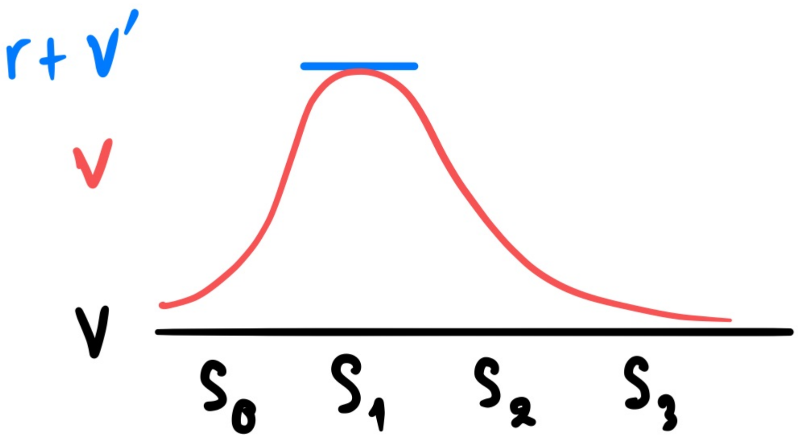
- Increase the capacity of the approximator
  - This reduces the generalization effect

- First-order tabular update approximation
  - Intuition, table is stable even with off-policy

First-order tabular update approximation



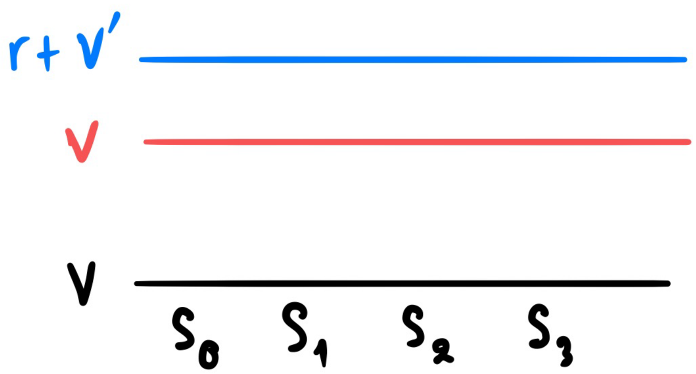
slope problem



generalization problem

First-order tabular update approximation

- Problems from “generalization”
- Problems from “slope”
  - SGD => steepest
  - Tabular => proportional
- If each update “knows” its “generalization”, it could correct itself!



Achiam, Joshua, Ethan Knight, and Pieter Abbeel. 2019. “Towards Characterizing Divergence in Deep Q-Learning.” *arXiv [cs.LG]*. arXiv. http://arxiv.org/abs/1903.08894.

Why do we love semi-gradient so much?

- It is fast
- It gives favorable fixed-point if trained successfully
  - TD fixed point
- It is probably the one which is shown to work on large problems
  - Atari
  - Go

Moving to true gradient

Semi to true gradient

- **Semi-gradient is the culprit**
- True gradient guarantees convergence
  - Merit from SGD
  - Even in non-linear case
  - But to where?
- **Loss functions:**
  - TD error
  - Bellman error
  - Projected Bellman error

TD Error (TDE)

$$\mathrm{TDE}(\theta) = \mathbb{E}\left[(R_{t+1} + \gamma v_\theta(S_{t+1}) - v_\theta(S_t))^2 | A_t \sim \pi\right]$$

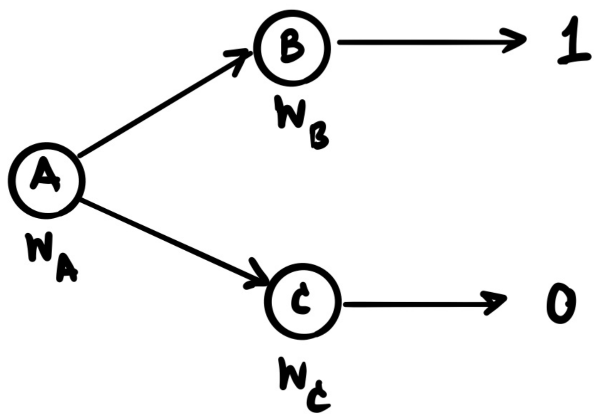
$$\mathrm{TDE}(\theta) = \mathbb{E}\left[\delta^2 | A_t \sim \pi\right]$$

True gradient:

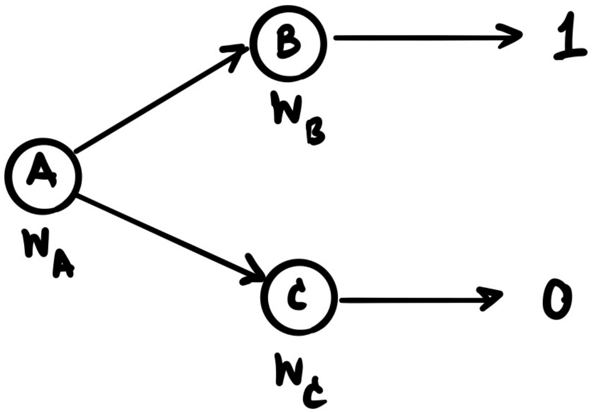
$$\theta \leftarrow \theta + \alpha\delta_t \left[\nabla_\theta v_\theta(s_t) - \gamma \nabla_\theta v_\theta(s_{t+1})\right]$$

How good is the fixed point?

A-split problem

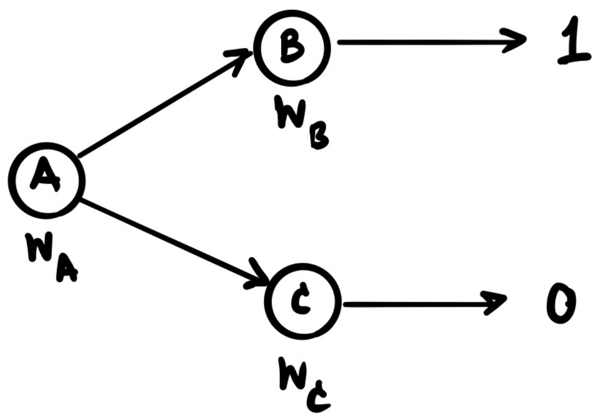


Semi-grad TD on A-split



$$\min_{v_\theta} \mathbb{E}_\pi[\delta^2]$$

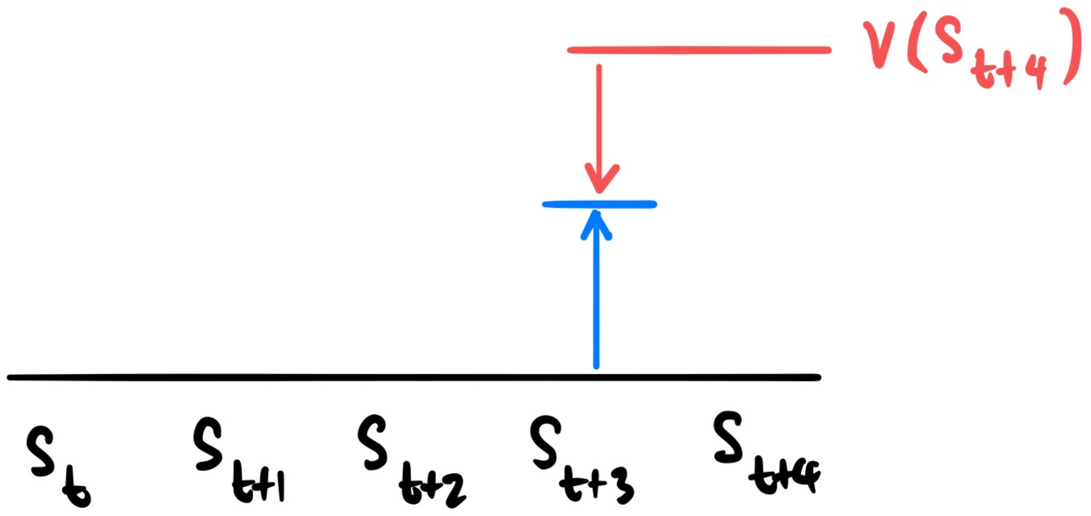
TDE on A-split

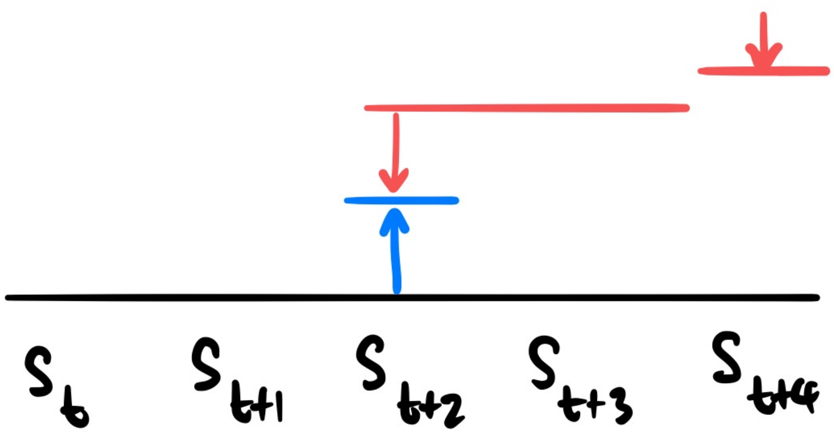


$$\min_\theta \mathbb{E}_\pi[\delta^2]$$ Tends to be large

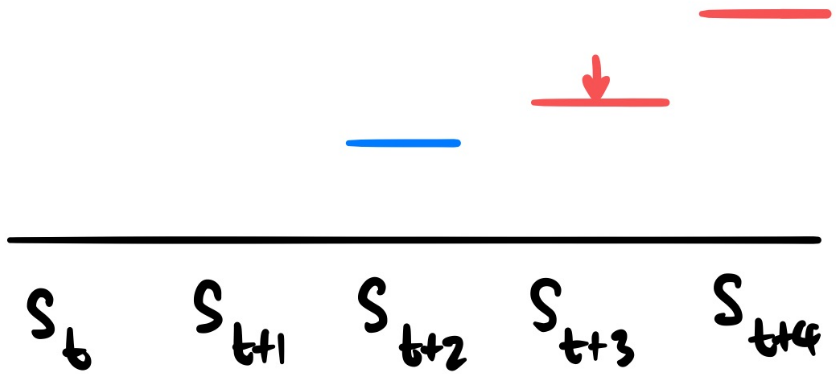
Temporal smoothing

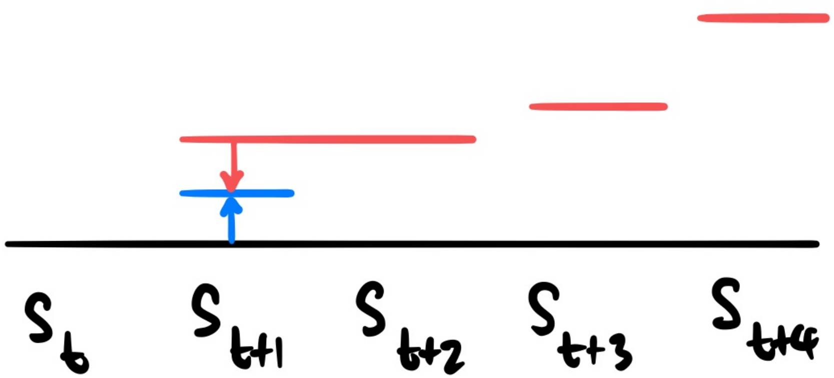
Update visualized true SGD TD





Update visualized true SGD TD





TDE seems bad, Alternatives?

Bellman error

- Bellman equation

$$v(s) = \sum_{a} \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma v(s')\right]$$

- Formulation for single state

$$\overline{\delta_\theta}(s) = \sum_{a} \pi(a|s) \sum_{s',r} p(s', r|s, a) \left[r + \gamma v_\theta(s')\right] - v_\theta(s)$$

$$= \mathbb{E}_\pi \left[\delta_s\right]$$

Bellman error

$$\overline{\delta_\theta}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a) \left[r + \gamma v_\theta(s')\right] - v_\theta(s)$$

$$= \mathbb{E}_\pi \left[\delta_s\right]$$

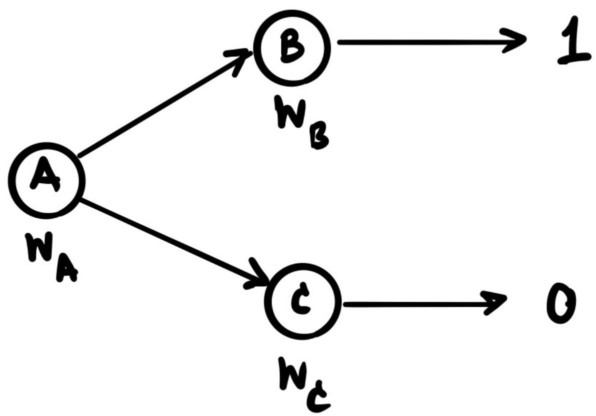
- Weighted squared error for all states

$$\|\overline{\delta_\theta}\|_\pi^2 = \mathrm{BE}(\theta) = \sum_s \mathbb{P}^\pi(s) \left[\mathbb{E}_\pi[\delta_s]^2\right]$$

vs. $E_\pi[\delta_s^2]$ TDE

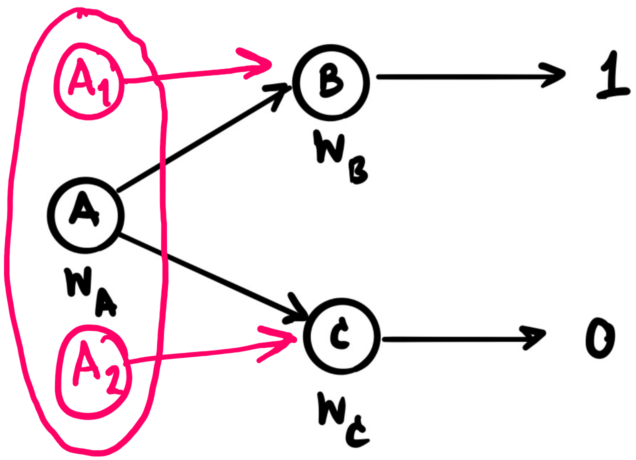
- Minimize the above, using SGD

BE on A-split



$$\min_\theta \mathbb{E}_\pi[\delta]^2$$ Tends to be smaller

BE on A-split (v2)



$$\min_\theta \mathbb{E}_\pi[\delta]^2$$ Tends to be smaller

Doesn't help anymore

Indistinguishable state features

Bellman error's gradient

$$\overline{\delta_\theta}(s) = \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_\theta(s')\right] - v_\theta(s)$$

$$\|\overline{\delta_\theta}\|_\pi^2 = \mathrm{BE}(\theta) = \sum_s \mathbb{P}^\pi(s)\left[\mathbb{E}_\pi[\delta_s]^2\right]$$

- Gradient

$$\nabla_\theta \mathrm{BE}(\theta) = \sum_s \mathbb{P}^\pi(s)\left[\mathbb{E}_\pi[\delta_s]\mathbb{E}_\pi[\gamma \nabla_\theta v_\theta(s') - \nabla_\theta v_\theta(s)]\right]$$

Double sampling problem

$$\|\overline{\delta_\theta}\|_\pi^2 = \mathrm{BE}(\theta) = \sum_s \mathbb{P}^\pi(s)\left[\mathbb{E}_\pi[\delta_s]^2\right]$$

$$\nabla_\theta \mathrm{BE}(\theta) = \sum_s \mathbb{P}^\pi(s)\left[\mathbb{E}_\pi[\delta_s]\mathbb{E}_\pi[\gamma \nabla_\theta v_\theta(s') - \nabla_\theta v_\theta(s)]\right]$$

Bellman error summary

- Impractical
- Might converge to undesirable fixed point
- A better loss function needed
  - Projected Bellman error
  - How to get its gradients (GTD2)

Read further

- Sutton 2018, Chapter 11
- Maei, Hamid Reza. 2011. “Gradient Temporal-Difference Learning Algorithms.” University of Alberta.
- Topics:
  - Projected Bellman error
  - GTD2

Levels of guarantees

- **Stability vs convergence**
  - Stability is easier to guarantee
- **On-policy vs off-policy**
  - On-policy is easier to guarantee
- **Linear vs non-linear**
  - Linear is easier to guarantee

Convergence of off-policy non-linear function approximation:

- Doesn’t imply a “good” fixed point

Summary of gradients

Semi gradients = converge only on linear with on-policy

- Semi-gradient TD
- You need luck and tricks

True gradients = converge even on non-linear, both on-policy and off-policy

- Value error
- TD error
- Bellman error

Assignment

- Q-learning with function approximation notebook (Github)