Off-policy Methods with Approximation

presented by Jason TOKO

Outline

- 1、Off-policy semi-gradient Methods
- 2、Off-policy Divergence and the Deadly Triad
- 3、Linear Value-function Geometry
- 4、SGD in the Bellman Error
- 5、Learnability of MSVE and MSBE
- 6、Gradient-TD Methods
- 7、Emphatic-TD Methods

Off-policy semi-gradient Methods

Off-policy semi-gradient Methods

- The **challenge** of off-policy learning can be divided into two parts:
- ①the target of the learning update
  - Importance sampling
  - Tree backup
- ②the distribution of the updates
  - True gradient methods(*)
  - Importance sampling

Off-policy semi-gradient Methods

- Let start with the target of the learning update~
- (per-step importance sampling ratio: $\rho_t \doteq \rho_{t:t} = \dfrac{\pi(A_t|S_t)}{b(A_t|S_t)}$.)
- (policy TD(0)

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \rho_t \delta_t \nabla \hat{v}(S_t, \mathbf{w}_t)$$

$$\delta_t \doteq R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t), \text{ or} \qquad \text{(episodic)}$$

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t). \qquad \text{(continuing)}$$

- (ted Sarsa

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \delta_t \nabla \hat{q}(S_t, A_t, \mathbf{w}_t)$$

$$\delta_t \doteq R_{t+1} + \gamma \sum_a \pi(a|S_{t+1}) \hat{q}(S_{t+1}, a, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t), \text{ or} \qquad \text{(episodic)}$$

$$\delta_t \doteq R_{t+1} - \bar{R}_t + \sum_a \pi(a|S_{t+1}) \hat{q}(S_{t+1}, a, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t). \qquad \text{(continuing)}$$

Off-policy semi-gradient Methods

- ③semi-gradient n-step Sarsa

$$\mathbf{w}_{t+n} \doteq \mathbf{w}_{t+n-1} + \alpha \rho_{t+1} \cdots \rho_{t+n-1} \left[G_{t:t+n} - \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1}) \right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1})$$

$$G_{t:t+n} \doteq R_{t+1} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}), \text{ or } \qquad \text{(episodic)}$$

$$G_{t:t+n} \doteq R_{t+1} - \bar{R}_t + \cdots + R_{t+n} - \bar{R}_{t+n-1} + \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}), \quad \text{(continuing)}$$

- ④ n-step tree-backup algorithm（without importance sampling）

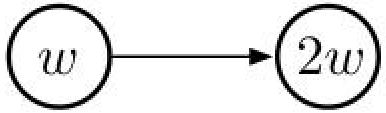
$$\mathbf{w}_{t+n} \doteq \mathbf{w}_{t+n-1} + \alpha \left[G_{t:t+n} - \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1}) \right] \nabla \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1})$$

$$G_{t:t+n} \doteq \hat{q}(S_t, A_t, \mathbf{w}_{t-1}) + \sum_{k=t}^{t+n-1} \delta_k \prod_{i=t+1}^{k} \gamma \pi(A_i|S_i),$$

Off-policy Divergence and the Deadly Triad

Off-policy Divergence

- However, off-policy learning with semi-gradient methods may be unstable and diverge.
- e.g.



- TD error:

$$\delta_t = R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t) = 0 + \gamma 2 w_t - w_t = (2\gamma - 1) w_t.$$

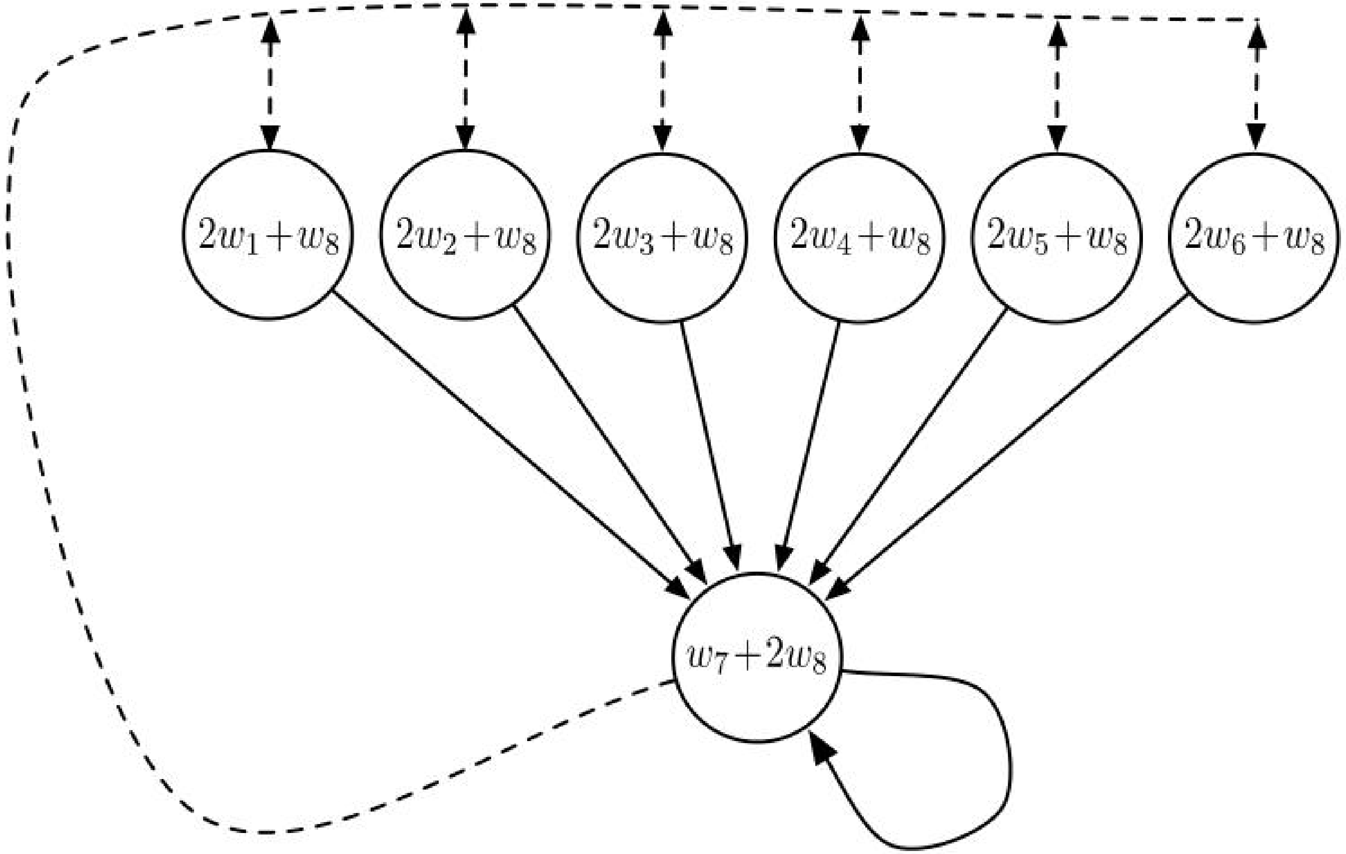
- If we use off-policy semi-gradient TD(0) update:

$$w_{t+1} = w_t + \alpha \rho_t \delta_t \nabla \hat{v}(S_t, w_t) = w_t + \alpha \cdot 1 \cdot (2\gamma - 1) w_t \cdot 1 = \big(1 + \alpha(2\gamma - 1)\big) w_t.$$

- Thus, if γ >0.5, then w will diverge.

Off-policy Divergence

- *Baird's counterexample:*



$\pi(\text{solid}|\cdot) = 1$

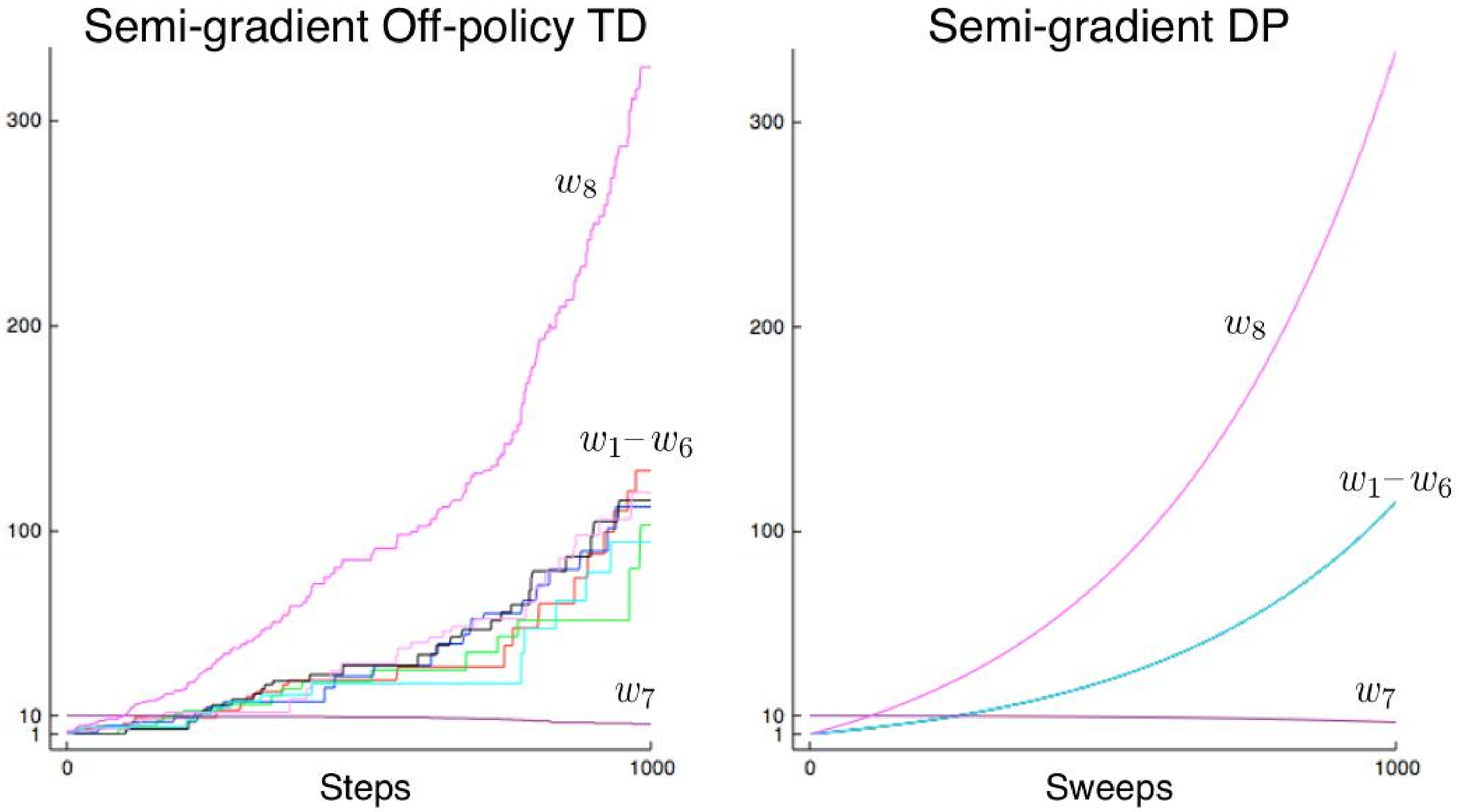
$\mu(\text{dashed}|\cdot) = 6/7$

$\mu(\text{solid}|\cdot) = 1/7$

$\gamma = 0.99$

Off-policy Divergence

- Result of semi-gradient TD(0) and semi-gradient DP:



They both

Off-policy Divergence

- In contrast, if we use asynchronous DP that the backups follow **on-policy distribution**, then it would converge to a solution with bounded error~
- Thus, **on-policy distribution** really matters!!!

The Deadly Triad

- The 3 factors that lead to the danger of instability and divergence are really terrible~~
- We usually call them: The deadly triad! (素质三连？？)
- ① **Function approximation**
- ② **Bootstrapping**
- ③ **Off-policy training**

Just like : $2KNO_3+3C+S==K_2S+N_2\uparrow+3CO_2\uparrow$

The Deadly Triad

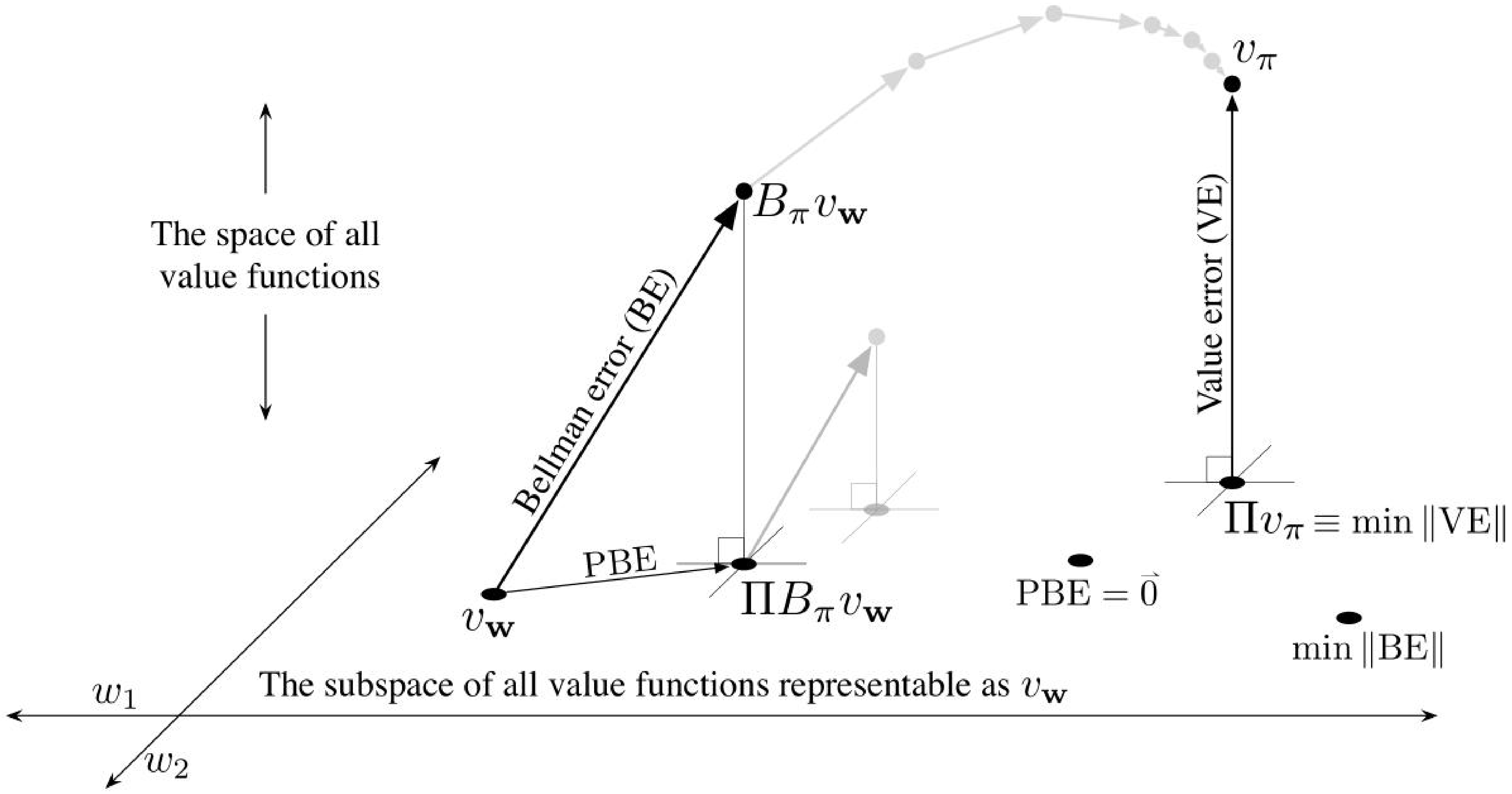
- Therefore, to avoid the danger, we may choose to give up one of them~
- **Function approximation:** it is the one that mostly cannot be given up because we need its **great expressive power.**
- **Bootstrapping:** doing without it is possible, but at the cost of **computational efficiency** and **data efficiency**
- **Off-policy training:** it is not a necessity, however it is essential to other anticipated use cases e.g. **parallel learning**

Linear Value-function Geometry

Linear Value-function Geometry

- *Let us talk about something relaxing~*
- *Geometry……orz*

Linear Value-function Geometry



Linear Value-function Geometry

- Something we should know……
  - ① The space of the value functions~
  - ② The distance between value function using the norm:

$$\|v\|_\mu^2 \doteq \sum_{s\in\mathcal{S}} \mu(s)v(s)^2.$$

  - ③ The projection operator Π :

$$\Pi v \doteq v_{\mathbf{w}} \quad \text{where} \quad \mathbf{w} = \arg\min_{\mathbf{w}} \|v - v_{\mathbf{w}}\|_\mu^2 .$$

matrix form: $\Pi := X(X^{\mathrm{T}}DX)^{-1}X^{\mathrm{T}}D$

  - ④ Bellman error(BE):

$$\bar{\delta}_{\mathbf{w}}(s) \doteq \left(\sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v_{\mathbf{w}}(s')\right]\right) - v_{\mathbf{w}}(s)$$

Linear Value-function Geometry

- Something we should know……
  - ⑤ A new objective function--**Mean Squared Bellman E**

    $$\mathrm{MSBE}(\mathbf{w}) = \left\|\bar{\delta}_{\mathbf{w}}\right\|_{\mu}^{2}$$

  - ⑥ $(B_\pi v)(s) \doteq \sum_a \pi(a|s) \sum_{s',r} p(s',r|s,a)\left[r + \gamma v(s')\right], \quad \forall s \in \mathcal{S}, \forall v : \mathcal{S} \to \mathbb{R}.$
  - ⑦ Projected Bellman error (PBE) vector: $\Pi\bar{\delta}_{v_w}$
  - ⑧ Another objective--**Mean Square Projected Bellman Error** (M

    $$\mathrm{MSPBE}(\mathbf{w}) = \left\|\Pi\bar{\delta}_{\mathbf{w}}\right\|_{\mu}^{2}$$

Linear Value-function Geometry

- Conclusion:
  - We can have many other objective function such as **MSBE** and **MSPBE**, in addition to MSVE that we have used before.
  - We will obtain different algorithms to minimize these objective.(we mainly talk about them later)
  - Of course, all of them have the different optimal parameters (**w**) and different value functions (**v**)

SGD in the Bellman Error

SGD in the Bellman Error

- Frankly speaking, **true SGD methods** (such as Monte Carlo methods) can converge very robustly,
- under both **on-policy** and **off-policy** training,
- as well as for general **non-linear** (differentiable) function approximators

- Thus, developing a true SGD method is really a valid way to eliminate instability and divergence~
- Here we would find true SGD methods for BE

SGD in the Bellman Error

- Firstly, we consider a **naive** objective: **Mean Squared TD Error** (MSTDE):

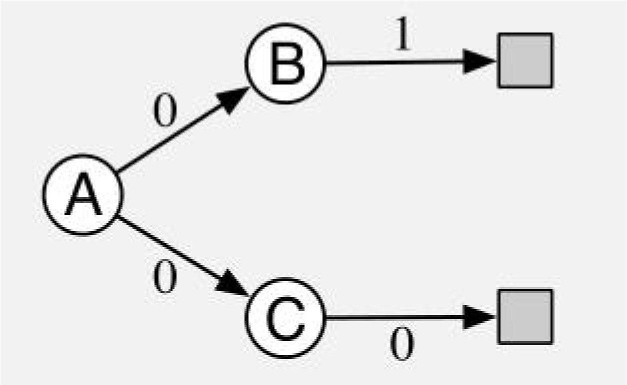
$$\begin{aligned}\mathrm{MSTDE}(\mathbf{w}) &= \sum_{s\in\mathcal{S}} \mu(s)\mathbb{E}\big[\delta_t^2 \mid S_t\!=\!s, A_t\!\sim\!\pi\big] \\ &= \sum_{s\in\mathcal{S}} \mu(s)\mathbb{E}\big[\rho_t\delta_t^2 \mid S_t\!=\!s, A_t\!\sim\!b\big] \\ &= \mathbb{E}_b\big[\rho_t\delta_t^2\big]\,.\end{aligned}$$

- And we can get a **naive residual-gradient** algorithm:

$$\begin{aligned}\mathbf{w}_{t+1} &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla(\rho_t\delta_t^2) \\ &= \mathbf{w}_t - \alpha\rho_t\delta_t\nabla\delta_t \\ &= \mathbf{w}_t + \alpha\rho_t\delta_t\big(\nabla\hat{v}(S_t,\mathbf{w}_t) - \gamma\nabla\hat{v}(S_t,\mathbf{w}_t)\big)\end{aligned}$$

SGD in the Bellman Error

- Although this algorithm converges robustly, it does not always converge to a desirable place:
- e.g.



It's so naive!!

- For the solution(v(A)=0.5, v(B)=0.75, v(C)=0.25), MSTDE=1/16
- But for true value(v(A)=0.5, v(B)=1, v(C)=0), MSTDE=1/8

SGD in the Bellman Error

- Since it is so naive, we should find a better algorithm to minimize MSBE:

$$
\begin{aligned}
\mathbf{w}_{t+1} &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla(\mathbb{E}_\pi[\delta_t]^2) \qquad \text{(expectation here implicitly conditional on } S_t) \\
&= \mathbf{w}_t - \frac{1}{2}\alpha\nabla(\mathbb{E}_b[\rho_t\delta_t]^2) \\
&= \mathbf{w}_t - \alpha\mathbb{E}_b[\rho_t\delta_t]\,\nabla\mathbb{E}_b[\rho_t\delta_t] \\
&= \mathbf{w}_t - \alpha\mathbb{E}_b\big[\rho_t(R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}) - \hat{v}(S_t,\mathbf{w}))\big]\,\mathbb{E}_b[\rho_t\nabla\delta_t] \\
&= \mathbf{w}_t + \alpha\Big[\mathbb{E}_b\big[\rho_t(R_{t+1} + \gamma\hat{v}(S_{t+1},\mathbf{w}))\big] - \hat{v}(S_t,\mathbf{w})\Big]\Big[\nabla\hat{v}(S_t,\mathbf{w}) - \gamma\mathbb{E}_b\big[\rho_t\nabla\hat{v}(S_{t+1},\mathbf{w})\big]\Big]
\end{aligned}
$$

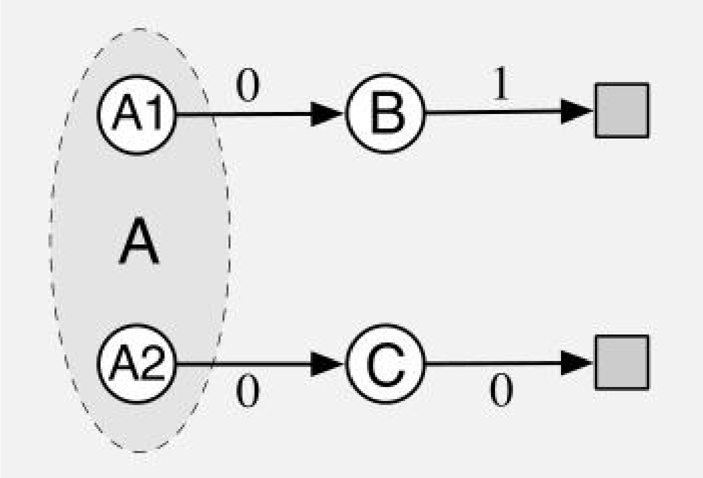
- called **residual gradient algorithm**

SGD in the Bellman Error

- The residual gradient algorithm can be used in 2 ways:
  - ① in the case of deterministic environments
  - ② obtain two independent samples of the next state from the current state(only can be done in the simulated environment)
- In either of these cases it is guaranteed to converge to a minimum of the MSBE, however……

SGD in the Bellman Error

- e.g.



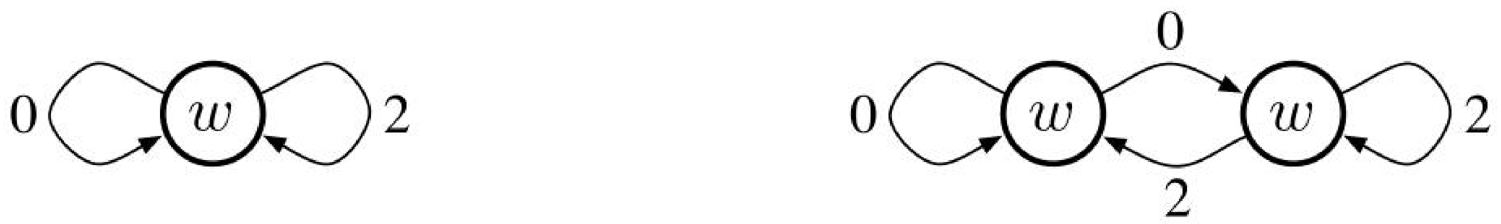
what a pity!

- For the solution(v(A)=0.5, v(B)=0.75, v(C)=0.25), MSBE=1/16
- But for true value(v(A)=0.5, v(B)=1, v(C)=0), MSBE=1/8
- On a **deterministic** problem, the Bellman errors and TD errors are all the same, so the MSBE is always the same as the MSTDE!!
- Thus, minimizing the MSBE may not be a desirable goal~

Learnability of MSVE and MSBE

Learnability of MSVE and MSBE

- To understand better, we now turn to learn about a new concept: Learnbaility
- Let's begin with an example:



- We cannot tell which the MRP produce the data, it is not learnable ------> MSVE is not learnable!!!

Learnability of MSVE and MSBE

- *Luckily, they still have the same optimal parameter* ***w****
- *In fact, optimal parameter(w*) of MSVE is* ***learnable:***

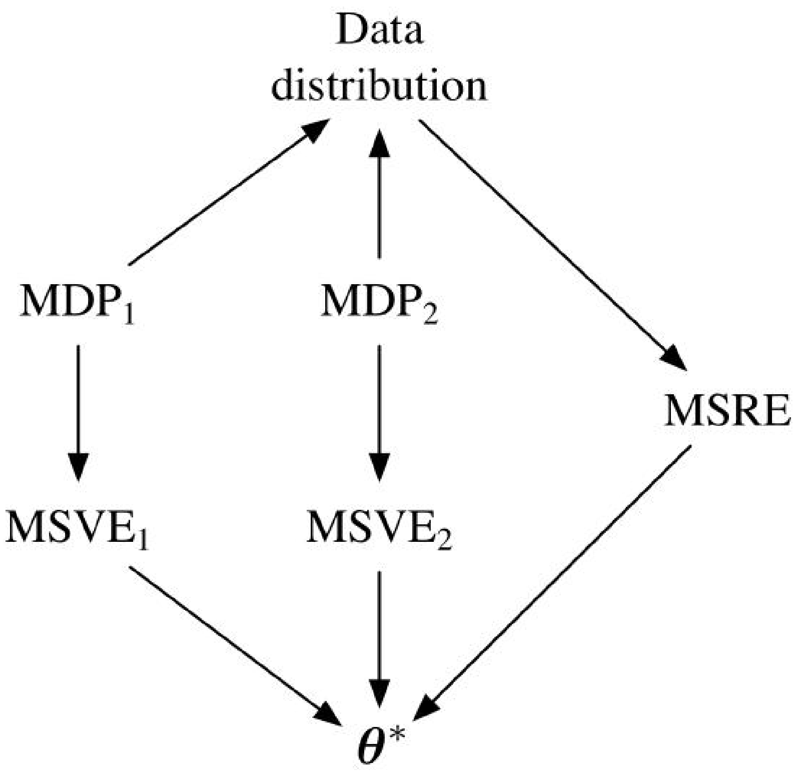
$$\begin{aligned}\mathrm{MSRE}(\mathbf{w}) &= \mathbb{E}\Big[\big(G_t - \hat{v}(S_t,\mathbf{w})\big)^2\Big] \\ &= \mathrm{MSVE}(\mathbf{w}) + \mathbb{E}\Big[\big(G_t - v_\pi(S_t)\big)^2\Big]\end{aligned}$$

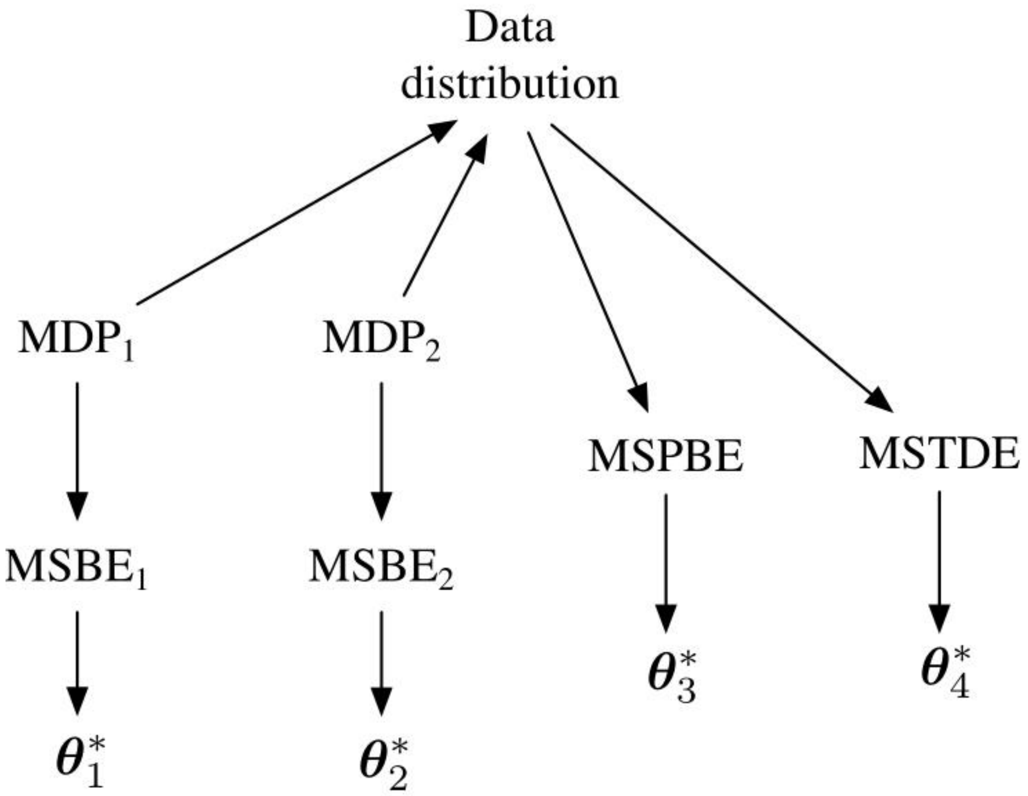
Learnability of MSVE and MSBE

- The MSBE can be computed from knowledge of the MDP but is not learnable from data.
- Not like MSVE, different MDP can have different minimum parameters------>MSBE's minimum solution is not learnable.

Learnability of MSVE and MSBE

- Conclusion:





Gradient-TD Methods

Gradient-TD Methods

- To compute the TD fixedpoint $\mathbf{w}_{TD}$, we have many methods:
- ① Linear semi-gradient TD(0)
  - Not true SGD, diverge under off-policy training......
- ② Least-squares TD (LSTD)
  - $O(d^2)$ complexity......
- ③ Gradient-TD Methods
  - True SGD and $O(d)$ complexity
  - Qiangwudi!!

Gradient-TD Methods

- SGD derivation for the MSPBE:

$$
\begin{aligned}
\mathrm{MSPBE}(\mathbf{w}) &= \left\| \Pi \bar{\delta}_{\mathbf{w}} \right\|_{\mu}^{2} \\
&= (\Pi \bar{\delta}_{\mathbf{w}})^{\top} D \Pi \bar{\delta}_{\mathbf{w}} \\
&= \bar{\delta}_{\mathbf{w}}^{\top} \Pi^{\top} D \Pi \bar{\delta}_{\mathbf{w}} \\
&= \bar{\delta}_{\mathbf{w}}^{\top} D \mathbf{X} \left(\mathbf{X}^{\top} D \mathbf{X} \right)^{-1} \mathbf{X}^{\top} D \bar{\delta}_{\mathbf{w}} \\
&= (\mathbf{X}^{\top} D \bar{\delta}_{\mathbf{w}})^{\top} (\mathbf{X}^{\top} D \mathbf{X})^{-1} (\mathbf{X}^{\top} D \bar{\delta}_{\mathbf{w}})
\end{aligned}
$$

- The gradient with respect to w:

$$
\nabla \mathrm{MSPBE}(\mathbf{w}) = 2 \nabla \left[\mathbf{X}^{\top} D \bar{\delta}_{\mathbf{w}} \right]^{\top} (\mathbf{X}^{\top} D \mathbf{X})^{-1} (\mathbf{X}^{\top} D \bar{\delta}_{\mathbf{w}})
$$

Gradient-TD Methods

- *Rewrite the three factors in terms of expectations*

$$\mathbf{X}^\top D\bar{\delta}_{\mathbf{w}} = \sum_s \mu(s)\mathbf{x}(s)\bar{\delta}_{\mathbf{w}}(s) = \mathbb{E}[\mathbf{x}_t\rho_t\delta_t]\,,$$

$$\begin{aligned}\nabla\mathbb{E}[\mathbf{x}_t\rho_t\delta_t]^\top &= \mathbb{E}\left[\rho_t\nabla\delta_t^\top\mathbf{x}_t^\top\right] \\ &= \mathbb{E}\left[\rho_t\nabla(R_{t+1}+\gamma\mathbf{w}^\top\mathbf{x}_{t+1}-\mathbf{w}^\top\mathbf{x}_t)^\top\mathbf{x}_t^\top\right] \\ &= \mathbb{E}\left[\rho_t(\gamma\mathbf{x}_{t+1}-\mathbf{x}_t)\mathbf{x}_t^\top\right].\end{aligned}$$

$$\mathbf{X}^\top D\mathbf{X} = \sum_s \mu(s)\mathbf{x}_s\mathbf{x}_s^\top = \mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^\top\right]$$

- *Thus* $\nabla\mathrm{MSPBE}(\mathbf{w}) = 2\mathbb{E}\left[\rho_t(\gamma\mathbf{x}_{t+1}-\mathbf{x}_t)\mathbf{x}_t^\top\right]\mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^\top\right]^{-1}\mathbb{E}[\mathbf{x}_t\rho_t\delta_t]\,.$

Gradient-TD Methods

- We devote a learned vector as $\mathbf{v} \approx \mathbb{E}\left[\mathbf{x}_t \mathbf{x}_t^\top\right]^{-1} \mathbb{E}[\mathbf{x}_t \rho_t \delta_t]$

- And it is updated by Least Mean Square (LMS) rule:

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \beta \rho_t \left(\delta_t - \mathbf{v}_t^\top \mathbf{x}_t\right) \mathbf{x}_t$$

Gradient-TD Methods

- So far, we can get a simple algorithm with **v** :

$$
\begin{aligned}
\mathbf{w}_{t+1} &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla\text{MSPBE}(\mathbf{w}_t) \\
&= \mathbf{w}_t - \frac{1}{2}\alpha 2\mathbb{E}\Big[\rho_t(\gamma\mathbf{x}_{t+1} - \mathbf{x}_t)\mathbf{x}_t^\top\Big]\,\mathbb{E}\Big[\mathbf{x}_t\mathbf{x}_t^\top\Big]^{-1}\mathbb{E}[\mathbf{x}_t\rho_t\delta_t] \\
&= \mathbf{w}_t + \alpha\mathbb{E}\Big[\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^\top\Big]\,\mathbb{E}\Big[\mathbf{x}_t\mathbf{x}_t^\top\Big]^{-1}\mathbb{E}[\mathbf{x}_t\rho_t\delta_t] \\
&= \mathbf{w}_t + \alpha\mathbb{E}\Big[\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^\top\Big]\,\mathbf{v}_t \\
&= \mathbf{w}_t + \alpha\rho_t\left(\mathbf{x}_t - \gamma\mathbf{x}_{t+1}\right)\mathbf{x}_t^\top\mathbf{v}_t.
\end{aligned}
$$

- called **GTD2**

Gradient-TD Methods

- A slightly better algorithm can be derived:

$$
\begin{aligned}
\mathbf{w}_{t+1} &= \mathbf{w}_t + \alpha \mathbb{E}\left[\rho_t(\mathbf{x}_t - \gamma \mathbf{x}_{t+1})\mathbf{x}_t^\top\right] \mathbb{E}\left[\mathbf{x}_t \mathbf{x}_t^\top\right]^{-1} \mathbb{E}[\mathbf{x}_t \rho_t \delta_t] \\
&= \mathbf{w}_t + \alpha \left(\mathbb{E}\left[\rho_t \mathbf{x}_t \mathbf{x}_t^\top\right] - \gamma \mathbb{E}\left[\rho_t \mathbf{x}_{t+1} \mathbf{x}_t^\top\right]\right) \mathbb{E}\left[\mathbf{x}_t \mathbf{x}_t^\top\right]^{-1} \mathbb{E}[\mathbf{x}_t \rho_t \delta_t] \\
&= \mathbf{w}_t + \alpha \left(\mathbb{E}[\mathbf{x}_t \rho_t \delta_t] - \gamma \mathbb{E}\left[\rho_t \mathbf{x}_{t+1} \mathbf{x}_t^\top\right] \mathbb{E}\left[\mathbf{x}_t \mathbf{x}_t^\top\right]^{-1} \mathbb{E}[\mathbf{x}_t \rho_t \delta_t]\right) \\
&= \mathbf{w}_t + \alpha \left(\mathbb{E}[\mathbf{x}_t \rho_t \delta_t] - \gamma \mathbb{E}\left[\rho_t \mathbf{x}_{t+1} \mathbf{x}_t^\top\right] \mathbf{v}_t\right) \\
&= \mathbf{w}_t + \alpha \rho_t \left(\delta_t \mathbf{x}_t - \gamma \mathbf{x}_{t+1} \mathbf{x}_t^\top \mathbf{v}_t\right),
\end{aligned}
$$

- called TD(0) with gradient correction (**TDC**) or **GTD(0)**

Emphatic-TD Methods

Emphatic-TD Methods

- 累了不想做了，有空补上。。。。。。。

emem~该去睡觉了，还是狗命重要！