WDS

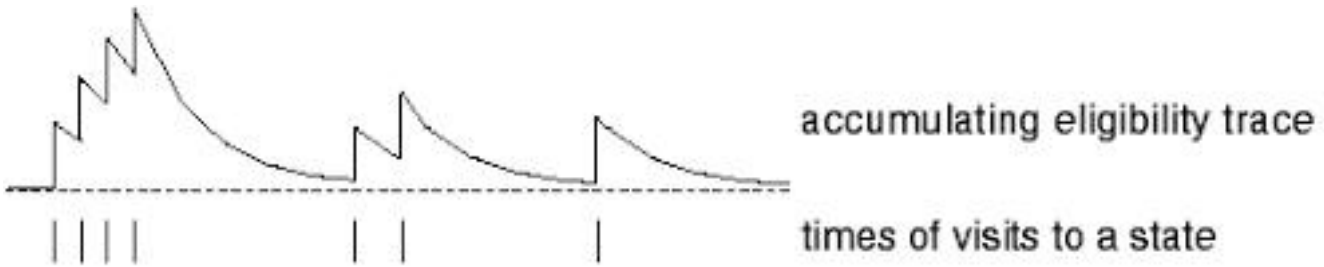
DataScience@Web

ELIGIBILITY TRACES

目录

What is Eligibility Traces

- 资格迹（Eligibility Traces）是RL的基本技术
  - 引入了短记忆向量 $\mathbf{z}_t \in \mathbb{R}^d$
  - 依赖于前面的状态（backward views）



Forward views：

- 蒙特卡洛算法：根据所有未来的奖励来进行更新状态。
- n-Step Bootstrapping：根据未来的n个奖励和未来的n个步来进行更新。

Backward views：

- TD(λ)：通过使用当前TD Error和Eligibility Traces查看之前的状态去实现近乎相同的更新。

Off-line λ-return

n-step return的一般形式：

$$G_{t:t+n} \doteq R_{t+1} + \gamma R_{t+2} + \cdots + \gamma^{n-1} R_{t+n} + \gamma^n \hat{v}(S_{t+n}, \mathbf{w}_{t+n-1}), \quad 0 \le t \le T - n.$$

目的：从n-step TD中的灵活性和误差减少性质中，更进一步提出平均衰减n-step returns，以更好的降低误差并提升鲁棒性

特点：前向视角(forward view)

复合更新(Compound update)：通过引入衰减参数 λ 将不同步长的n-step return线性加权组合成λ-returns，具体形式如下：

$$G_t^\lambda = (1-\lambda)[1 \cdot G_{t:t+1} + \lambda G_{t:t+2} + \lambda^2 G_{t:t+3} + \lambda^3 G_{t:t+4} + \ldots]$$

$$= (1-\lambda) \sum_{n=1}^{\infty} \lambda^{n-1} G_{t:t+n}$$

$$= (1-\lambda) \sum_{n=1}^{T-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{T-t-1} G_t \quad (\text{分离终止项})$$

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \left[G_t^\lambda - \hat{v}(S_t, \mathbf{w}_t) \right] \nabla \hat{v}(S_t, \mathbf{w}_t), \quad t = 0, \ldots, T-1.$$

Off-line λ-return

Off-line的体现：在等待当前episode结束时才能进行更新

```
...
while state not in END_STATE:
    take action 'A' and obervse 'next_state', 'R'
    self.trajectory.append(next_state)
    if next_state in END_STATES:
       self.off_line_learn()
    state = next_state
...
```

带有资格迹的TD(λ)

TD(λ) 在Off-line λ-return的基础上作了三点改进：

- 边采样边更新权重参数w，用TD error代替原式，即

$$U_t = R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t)$$

- 将Off-line终点时的计算分配到每个时间步上，即增量式计算，通过资格迹$\mathbf{z}_t \in \mathbb{R}^d$

在采样的同时记录当前时刻的 $\nabla \hat{v}(S_t, \mathbf{w}_t)$ 并通过折扣因子 γ 和迹衰减参数 λ 不断对求和项中的历史值函数梯度$\nabla \hat{v}(S_{t-k}, \mathbf{w}_{t-k})$进行衰减实现

- 使用函数逼近方法来构造近似值函数 $\hat{v}(S, \mathbf{w})$

带有资格迹的TD(λ)

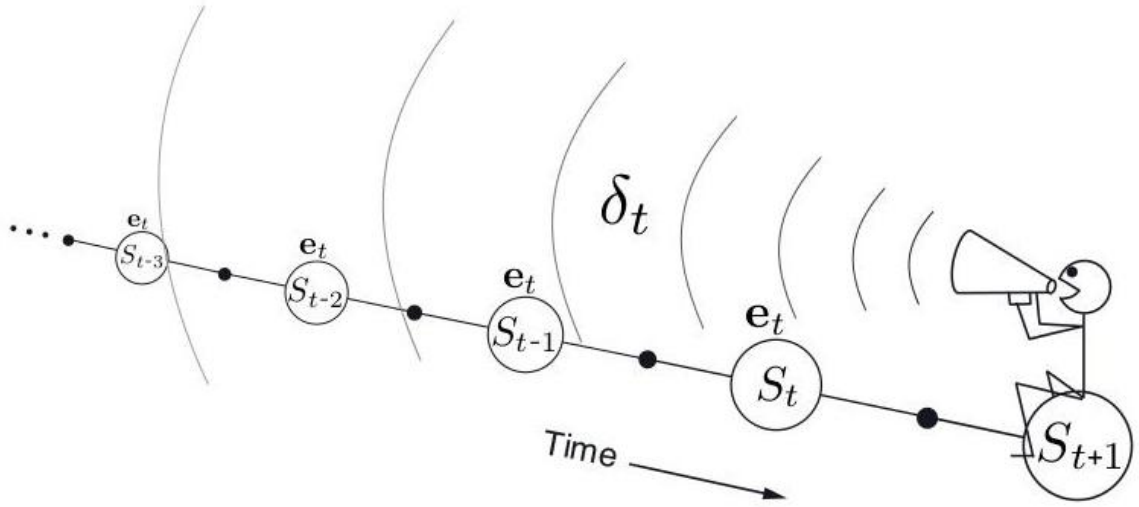
Eligibility Traces是一个维度和权重向量相同的向量，在TD(λ)中，其迭代方式如下：

$$\mathbf{z}_{-1} \doteq \mathbf{0},$$
$$\mathbf{z}_t \doteq \gamma\lambda\mathbf{z}_{t-1} + \nabla\hat{v}(S_t,\mathbf{w}_t), \quad 0 \le t \le T,$$

每一步的权重更新方式如下：

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

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\delta_t\mathbf{z}_t,$$



Forward-view TD(λ)

$$G_t^\lambda = (1-\lambda)\sum_{n=1}^{T-t-1}\lambda^{n-1}G_{t:t+n} + \lambda^{T-t-1}G_t,$$

$$V(S_t) \leftarrow V(S_t) + \alpha\left(G_t^\lambda - V(S_t)\right)$$

12.2. TD(λ) 239

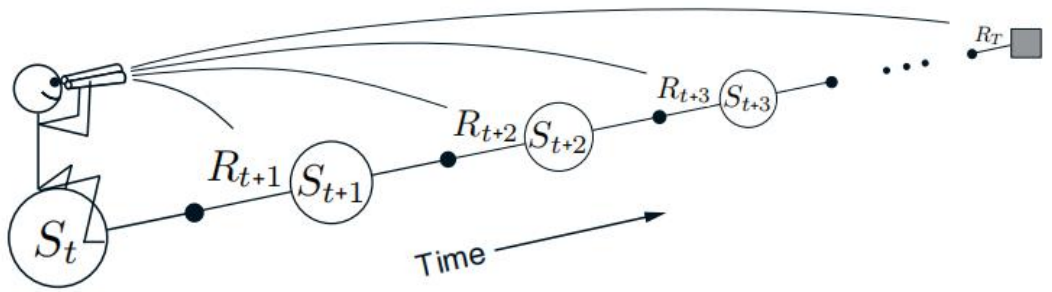


Figure 12.4: The forward view. We decide how to update each state by looking forward to future rewards and states.

带有资格迹的TD(λ)

$\mathbf{z}_t$ 的具体展开：

$$\mathbf{z}_t = 1 \cdot \nabla \hat{v}(S_t, \mathbf{w}_t) + (\gamma\lambda)\nabla \hat{v}(S_{t-1}, \mathbf{w}_{t-1}) + (\gamma\lambda)^2 \nabla \hat{v}(S_{t-2}, \mathbf{w}_{t-2}) + \ldots + (\gamma\lambda)^{t-1} \nabla \hat{v}(S_1, \mathbf{w}_1)$$

带有资格迹的TD(λ)

TD(λ)对比off-line λ-return算法的优点在于，它每一步都能及时更新迭代，学习会较快，同时可以处理连续性任务的问题。

- $\lambda = 0 \rightarrow$ one step TD算法，Eligibility Traces只考虑t时刻的梯度。
- $\lambda > 0$，则t时刻之前的状态对应的梯度也被考虑进去，即t时刻前的状态值也会被直接改变。
- $\lambda = 1$，这个时候是一种实现MC算法的方法

If $\lambda = 1$, then the credit given to earlier states falls only by γ per step. This turns out to be just the right thing to do to achieve Monte Carlo behavior. For example, remember that the TD error, δ_t, includes an undiscounted term of R_{t+1}. In passing this back k steps it needs to be discounted, like any reward in a return, by γ^k, which is just what the falling eligibility trace achieves. If $\lambda = 1$ and $\gamma = 1$, then the eligibility traces do not decay at all with time. In this case the method behaves like a Monte Carlo method for an undiscounted, episodic task. If $\lambda = 1$, the algorithm is also known as TD(1).

带有资格迹的TD(λ)

简单证明如下：

t时刻，eligibility traces为 $z_t = \sum_{i=0}^{t} y^{t-i}\Delta v(S_i, W_i)$

权重更新为 $W_{t+1} = W_t + \alpha\delta_t z_t$

所以，在一个事件结束后，权重总体更新为 $\alpha\sum \delta_t z_t$

此时，对于状态S0而言，总体更新为 $\sum_t y^t \Delta v(S_0, W_0)R_{t+1} + \Delta v(S_0, W_0)\ (v_{terminate} - v_{S_0})$

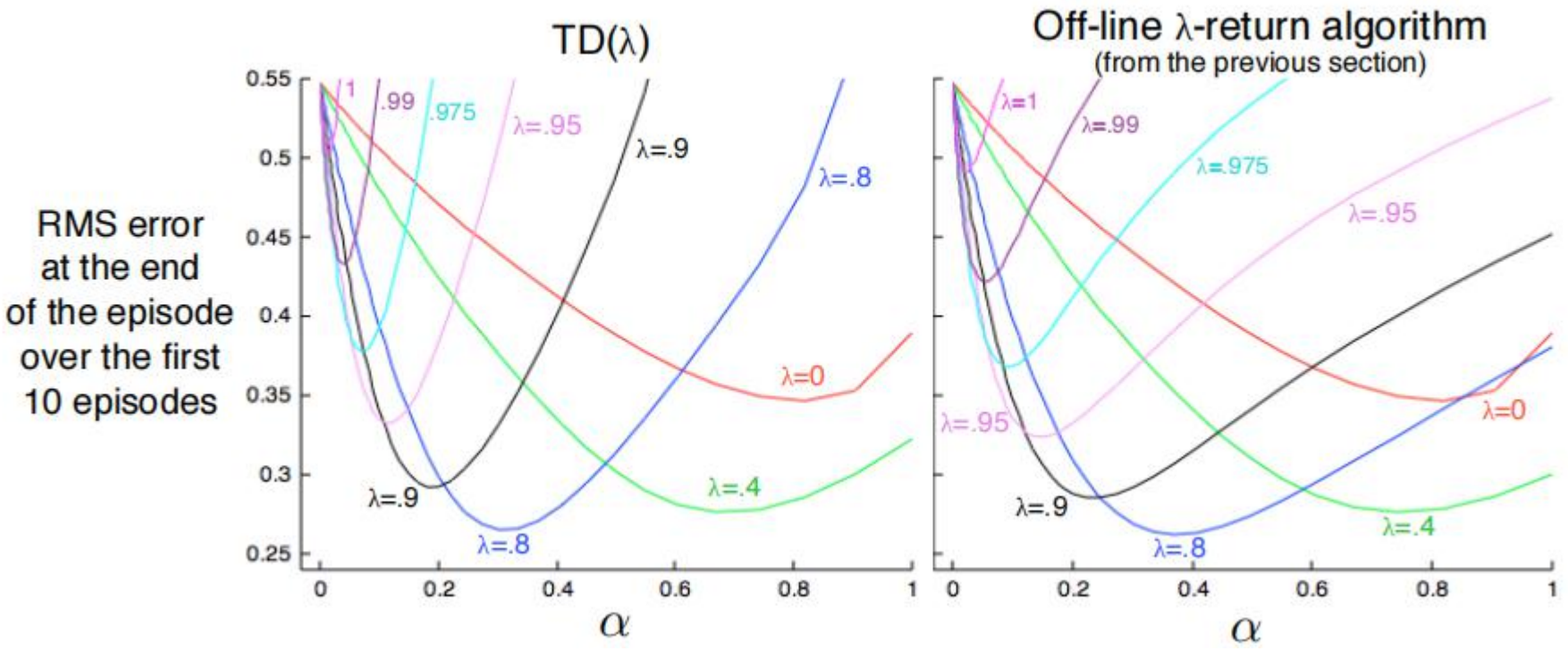
带有资格迹的TD(λ)

Semi-gradient TD(λ) for estimating $\hat{v} \approx v_\pi$

Input: the policy π to be evaluated
Input: a differentiable function $\hat{v} : \mathcal{S}^+ \times \mathbb{R}^d \to \mathbb{R}$ such that $\hat{v}(\text{terminal},\cdot) = 0$

Initialize value-function weights $\mathbf{w}$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Repeat (for each episode):
  Initialize S
  $\mathbf{z} \leftarrow \mathbf{0}$ (a d-dimensional vector)
  Repeat (for each step of episode):
  . Choose $A \sim \pi(\cdot|S)$
  . Take action A, observe R, S'
  . $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z} + \nabla\hat{v}(S,\mathbf{w})$
  . $\delta \leftarrow R + \gamma\hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$
  . $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
  . $S \leftarrow S'$
  until S' is terminal

TD(λ) VS Off-line λ-return



n-step Truncated λ-return Methods

λ-return考虑了后面所有的奖励值R，而Truncated λ-return简单讲就是只考虑到Rh，只考虑时刻t后面h-t步的奖励值，时刻h后的奖励值被截断，用Gt:h替代。

$$G_{t:h}^{\lambda} \doteq (1-\lambda) \sum_{n=1}^{h-t-1} \lambda^{n-1} G_{t:t+n} + \lambda^{h-t-1} G_{t:h}, \qquad 0 \leq t < h \leq T.$$

$$\mathbf{w}_{t+n} \doteq \mathbf{w}_{t+n-1} + \alpha \left[G_{t:t+n}^{\lambda} - \hat{v}(S_t, \mathbf{w}_{t+n-1}) \right] \nabla \hat{v}(S_t, \mathbf{w}_{t+n-1}), \qquad 0 \leq t < T.$$

The Online λ-return Algorithm

对于时刻h而言，Truncated λ-return截断到h，然后逐次更新S0到Sh-1的状态值，如下：

$$
\begin{aligned}
h=1: \quad & \mathbf{w}_1^1 \doteq \mathbf{w}_0^1 + \alpha\left[G_{0:1}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^1)\right]\nabla\hat{v}(S_0,\mathbf{w}_0^1), \\
h=2: \quad & \mathbf{w}_1^2 \doteq \mathbf{w}_0^2 + \alpha\left[G_{0:2}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^2)\right]\nabla\hat{v}(S_0,\mathbf{w}_0^2), \\
& \mathbf{w}_2^2 \doteq \mathbf{w}_1^2 + \alpha\left[G_{1:2}^{\lambda} - \hat{v}(S_1,\mathbf{w}_1^2)\right]\nabla\hat{v}(S_1,\mathbf{w}_1^2), \\
h=3: \quad & \mathbf{w}_1^3 \doteq \mathbf{w}_0^3 + \alpha\left[G_{0:3}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^3)\right]\nabla\hat{v}(S_0,\mathbf{w}_0^3), \\
& \mathbf{w}_2^3 \doteq \mathbf{w}_1^3 + \alpha\left[G_{1:3}^{\lambda} - \hat{v}(S_1,\mathbf{w}_1^3)\right]\nabla\hat{v}(S_1,\mathbf{w}_1^3), \\
& \mathbf{w}_3^3 \doteq \mathbf{w}_2^3 + \alpha\left[G_{2:3}^{\lambda} - \hat{v}(S_2,\mathbf{w}_2^3)\right]\nabla\hat{v}(S_2,\mathbf{w}_2^3).
\end{aligned}
$$

一般地

$$
\mathbf{w}_{t+1}^h \doteq \mathbf{w}_t^h + \alpha\left[G_{t:h}^{\lambda} - \hat{v}(S_t,\mathbf{w}_t^h)\right]\nabla\hat{v}(S_t,\mathbf{w}_t^h), \quad 0 \le t < h \le T.
$$

The Online λ-return Algorithm

对于时刻h而言，Truncated λ-return截断到h，然后逐次更新S0到Sh-1的状态值，如下：

$$
\begin{aligned}
h=1: \quad & \mathbf{w}_1^1 \doteq \mathbf{w}_0^1 + \alpha\left[G_{0:1}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^1)\right]\nabla\hat{v}(S_0,\mathbf{w}_0^1),\\
h=2: \quad & \mathbf{w}_1^2 \doteq \mathbf{w}_0^2 + \alpha\left[G_{0:2}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^2)\right]\nabla\hat{v}(S_0,\mathbf{w}_0^2),\\
& \mathbf{w}_2^2 \doteq \mathbf{w}_1^2 + \alpha\left[G_{1:2}^{\lambda} - \hat{v}(S_1,\mathbf{w}_1^2)\right]\nabla\hat{v}(S_1,\mathbf{w}_1^2),\\
h=3: \quad & \mathbf{w}_1^3 \doteq \mathbf{w}_0^3 + \alpha\left[G_{0:3}^{\lambda} - \hat{v}(S_0,\mathbf{w}_0^3)\right]\nabla\hat{v}(S_0,\mathbf{w}_0^3),\\
& \mathbf{w}_2^3 \doteq \mathbf{w}_1^3 + \alpha\left[G_{1:3}^{\lambda} - \hat{v}(S_1,\mathbf{w}_1^3)\right]\nabla\hat{v}(S_1,\mathbf{w}_1^3),\\
& \mathbf{w}_3^3 \doteq \mathbf{w}_2^3 + \alpha\left[G_{2:3}^{\lambda} - \hat{v}(S_2,\mathbf{w}_2^3)\right]\nabla\hat{v}(S_2,\mathbf{w}_2^3).
\end{aligned}
$$

一般地

$$
\mathbf{w}_{t+1}^h \doteq \mathbf{w}_t^h + \alpha\left[G_{t:h}^{\lambda} - \hat{v}(S_t,\mathbf{w}_t^h)\right]\nabla\hat{v}(S_t,\mathbf{w}_t^h), \quad 0 \le t < h \le T.
$$

True Online TD(λ)

由 on-line λ-return algorithm 产生一个三角形的权值向量序列：

$$
\begin{array}{ccccccc}
\mathbf{w}_0^0 & & & & & & \\
\mathbf{w}_0^1 & \mathbf{w}_1^1 & & & & & \\
\mathbf{w}_0^2 & \mathbf{w}_1^2 & \mathbf{w}_2^2 & & & & \\
\mathbf{w}_0^3 & \mathbf{w}_1^3 & \mathbf{w}_2^3 & \mathbf{w}_3^3 & & & \\
\vdots & \vdots & \vdots & \vdots & \ddots & & \\
\mathbf{w}_0^T & \mathbf{w}_1^T & \mathbf{w}_2^T & \mathbf{w}_3^T & \cdots & \mathbf{w}_T^T
\end{array}
$$

对于 $\hat{v}(s,\mathbf{w}) = \mathbf{w}^\top \mathbf{x}(s)$ 的线性情况，true online TD(λ)：

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\,\delta_t\,\mathbf{z}_t + \alpha\left(\mathbf{w}_t^\top \mathbf{x}_t - \mathbf{w}_{t-1}^\top \mathbf{x}_t\right)(\mathbf{z}_t - \mathbf{x}_t),$$

$$\mathbf{z}_t \doteq \gamma\lambda \mathbf{z}_{t-1} + \left(1 - \alpha\gamma\lambda \mathbf{z}_{t-1}^\top \mathbf{x}_t\right)\mathbf{x}_t.$$

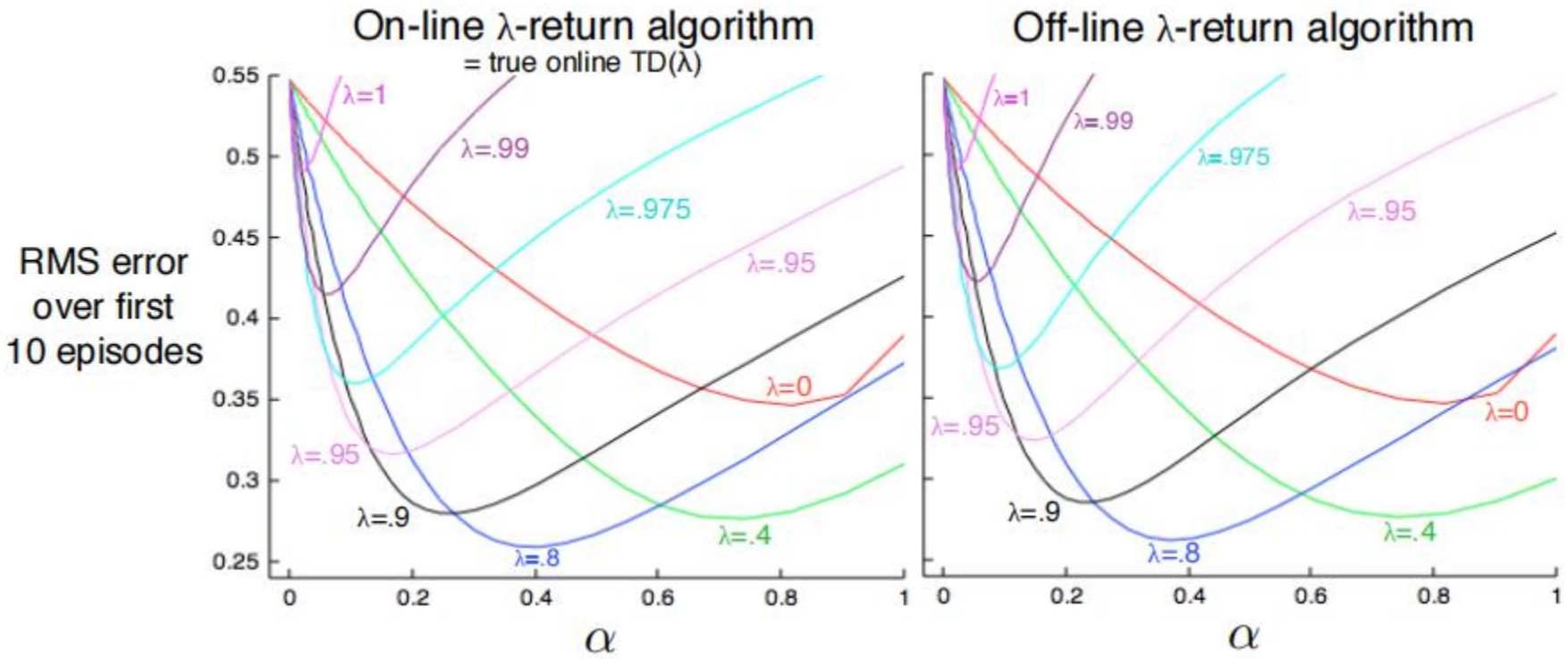
True Online TD(λ)

True Online TD(λ) for estimating $\mathbf{w}^\top \mathbf{x} \approx v_\pi$

Input: the policy π to be evaluated

Initialize value-function weights $\mathbf{w}$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Repeat (for each episode):
  Initialize state and obtain initial feature vector $\mathbf{x}$
  $\mathbf{z} \leftarrow \mathbf{0}$ (an d-dimensional vector)
  $V_{old} \leftarrow 0$ (a scalar temporary variable)
  Repeat (for each step of episode):
    Choose $A \sim \pi$
    Take action A, observe R, $\mathbf{x}'$ (feature vector of the next state)
    $V \leftarrow \mathbf{w}^\top \mathbf{x}$
    $V' \leftarrow \mathbf{w}^\top \mathbf{x}'$
    $\delta \leftarrow R + \gamma V' - V$
    $\mathbf{z} \leftarrow \gamma\lambda \mathbf{z} + \left(1 - \alpha\gamma\lambda \mathbf{z}^\top \mathbf{x}\right) \mathbf{x}$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha(\delta + V - V_{old})\mathbf{z} - \alpha(V - V_{old})\mathbf{x}$
    $V_{old} \leftarrow V'$
    $\mathbf{x} \leftarrow \mathbf{x}'$
  until $\mathbf{x}' = \mathbf{0}$ (signaling arrival at a terminal state)

True Online TD(λ) VS Off-line λ-return



Sarsa(λ)

λ return

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

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha \Big[G_t^\lambda - \hat{q}(S_t, A_t \mathbf{w}_t) \Big] \nabla \hat{q}(S_t, A_t, \mathbf{w}_t), \quad t = 0, \ldots, T-1$$

eligibility trace

$$\delta_t \doteq R_{t+1} + \gamma \hat{q}(S_{t+1}, A_{t+1}, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t)$$

$$\begin{aligned} \mathbf{z}_{-1} &\doteq \mathbf{0}, \\ \mathbf{z}_t &\doteq \gamma \lambda \mathbf{z}_{t-1} + \nabla \hat{q}(S_t, A_t, \mathbf{w}_t), \quad 0 \le t \le T \end{aligned}$$

Sarsa(λ)

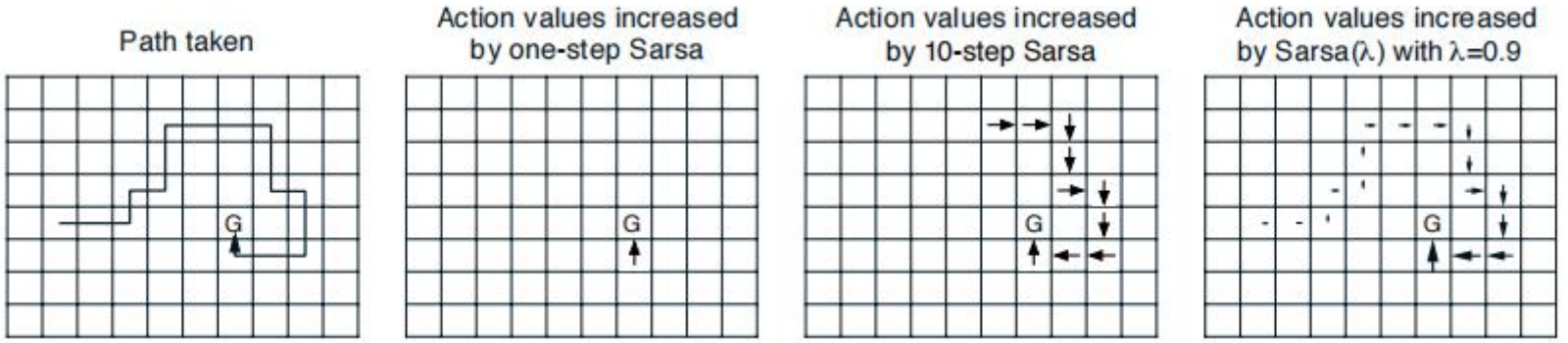
Sarsa(λ) with binary features and linear function approximation for estimating $\hat{q} \approx q_\pi$ or $\hat{q} \approx q_*$

Input: a function $\mathcal{F}(s,a)$ returning the set of (indices of) active features for s, a
Input: a policy π to be evaluated, if any

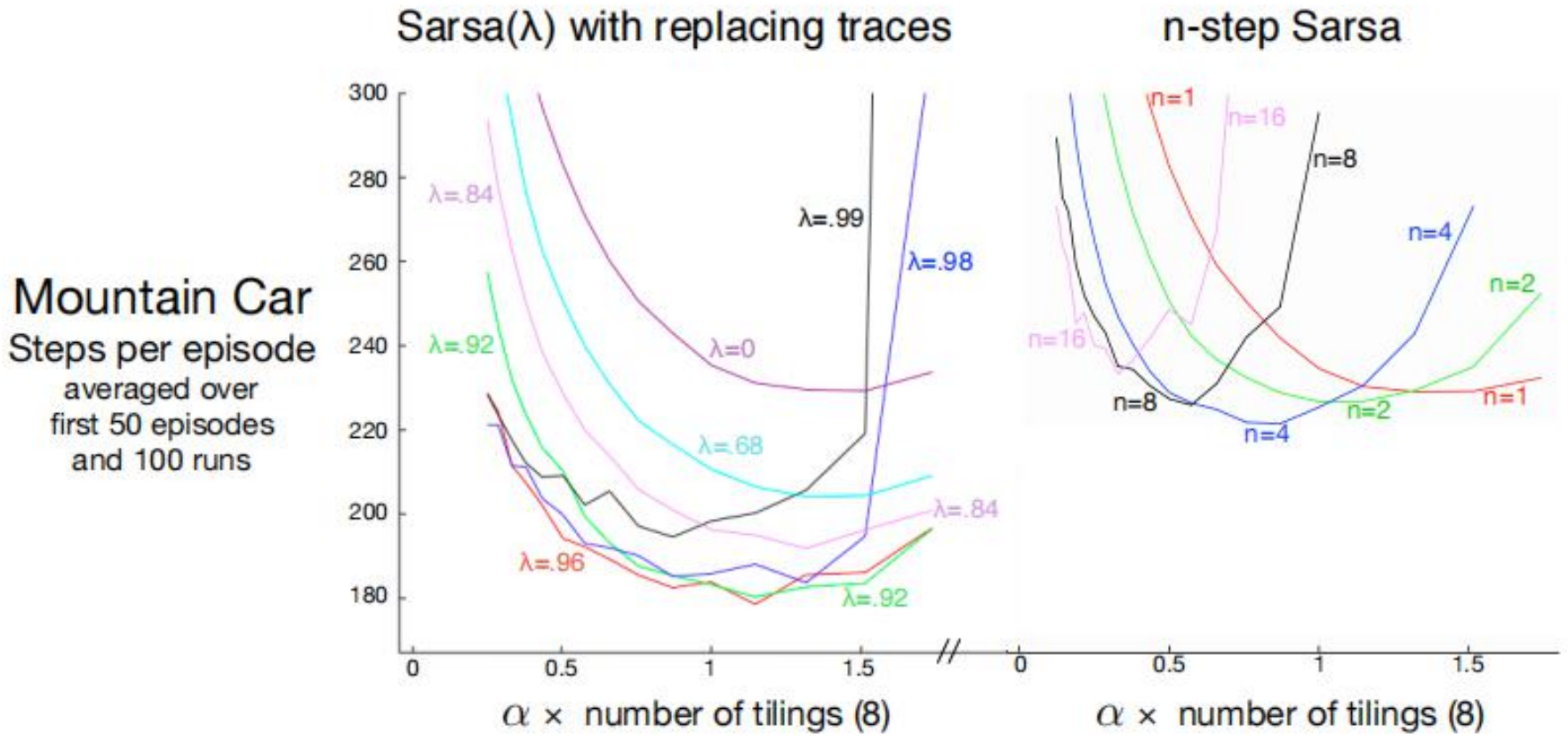
Initialize parameter vector $\mathbf{w} = (w_1, \ldots, w_n)$ arbitrarily (e.g., $\mathbf{w} = \mathbf{0}$)
Loop for each episode:
  Initialize S
  Choose $A \sim \pi(\cdot|S)$ or ε-greedy according to $\hat{q}(S, \cdot, \mathbf{w})$
  $\mathbf{z} \leftarrow \mathbf{0}$
  Loop for each step of episode:
    Take action A, observe R, S'
    $\delta \leftarrow R$
    Loop for i in $\mathcal{F}(S, A)$:
      $\delta \leftarrow \delta - w_i$
      $z_i \leftarrow z_i + 1$ (accumulating traces)
      or $z_i \leftarrow 1$ (replacing traces)
    If S' is terminal then:
      $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
      Go to next episode
    Choose $A' \sim \pi(\cdot|S')$ or near greedily $\sim \hat{q}(S', \cdot, \mathbf{w})$
    Loop for i in $\mathcal{F}(S', A')$: $\delta \leftarrow \delta + \gamma w_i$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha\delta\mathbf{z}$
    $\mathbf{z} \leftarrow \gamma\lambda\mathbf{z}$
    $S \leftarrow S'; A \leftarrow A'$

Sarsa(λ)

Example 12.1: Traces in Gridworld The use of eligibility traces can substantially increase the efficiency of control algorithms over one-step methods and even over n-step methods. The reason for this is illustrated by the gridworld example below.



Sarsa(λ)


Mountain Car
Steps per episode
averaged over
first 50 episodes
and 100 runs
Sarsa(λ) with replacing traces
n-step Sarsa
300
280
260
240
220
200
180
λ=.84
λ=.99
λ=.98
λ=.92
λ=0
λ=.68
λ=.84
λ=.96
λ=.92
n=1
n=16
n=8
n=4
n=2
n=16
n=8
n=2
n=1
n=4
0
0.5
1
1.5
0
0.5
1
1.5
α × number of tilings (8)
α × number of tilings (8)

总结

优点

- Eligibility Traces与TD Error结合，提供了一种高效、增量的折中TD和MC的方式
- 学习速度快（无需等episode结束再进行回报估计）
- 节省存储（不用存储中间n步的奖赏和状态序列（n-step TD））

缺点

- 比one-step算法需要更多的计算
- 不适合off-line的应用

Thanks!