# Reinforcement Learning: An Introduction notebook

黎雷蕾

2017 年 12 月 28 日

# 目录

# Chapter 11

# Off-policy Methods with Approximation

## 11.1 Semi-gradient Methods

在函数值近似中，每步重要性采样比例公式可以写为：

$$\rho_t \doteq \rho_{t:t} = \frac{\pi(A_t|S_t)}{b(A_t|S_t)} \tag{11.1}$$

比如说，semi-gradient off-policy TD(0) 算法的权重向量更新可以写为：

$$\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\rho_t\delta_t\nabla\hat{v}(S_t, \mathbf{w}) \tag{11.2}$$

其中 $\delta_t$ 是采用平均回报的 TD error：

$$\begin{aligned} \delta_t &\doteq R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t), \text{or} \\ \delta_t &\doteq R_{t+1} - \overline{R}_t + \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t) \end{aligned} \tag{11.3}$$

对于动作-价值来说，上面的算法就变成了 one-step semi-gradient Expected Sarsa：

$$\begin{aligned} &\mathbf{w}_{t+1} \doteq \mathbf{w}_t + \alpha\rho_t\delta_t\nabla\hat{q}(S_t, A_t, \mathbf{w}_t), \text{with} \\ &\delta_t \doteq R_{t+1} + \gamma\sum_a \pi(a|S_t+1)\hat{q}(S_{t+1}, a, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t), \text{or} \\ &\delta_t \doteq R_{t+1} - \overline{R}_t + \sum_a \pi(a|S_{t+1})\hat{q}(S_{t+1}, a, \mathbf{w}_t) - \hat{q}(S_t, A_t, \mathbf{w}_t) \end{aligned} \tag{11.4}$$

同理，$n$-step semi-gradient Expected Sarsa 可以写为：

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

with

$$\begin{aligned} G_{t:t+n} &\doteq R_{t+1} + \cdots + \gamma^{n-1}R_{t+n} + \gamma^n\hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}), \text{or} \\ G_{t:t+n} &\doteq R_{t+1} - \overline{R}_t + \cdots + R_{t+n} - \overline{R}_{t+n-1} + \hat{q}(S_{t+n}, A_{t+n}, \mathbf{w}_{t+n-1}) \end{aligned} \tag{11.5}$$

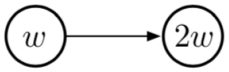
n-step semi-gradient tree-backup 算法可以写为：

$$\begin{aligned} &\mathbf{w}_{t+n} \doteq \mathbf{w}_{t+n-1} + \alpha[G_{t:t+n} - \hat{q}(S_t, A_t, \mathbf{w}_{t+n-1})]\nabla\hat{q}(S_t, A_t, \mathbf{w}_{t+n-1}), \text{with} \\ &G_{t:t+n} \doteq \hat{q}(S_t, A_t, \mathbf{w}_{t-1}) + \sum_{k=t}^{t+n-1}\delta_k\prod_{i=t+1}^{k}\gamma\pi(A_i|S_i) \end{aligned} \tag{11.6}$$

## 11.2 Examples of Off-policy Divergence

在近似值逼近中会遇到一个困难：行为策略的分布和目标策略的分布不一致 (the distribution of updates does not match the on-policy distribution)。

在下图中，有两个状态，一个动作，从左边状态到右边状态的 reward 为 0：




两个状态进行转换时产生的 TD error:

$$\begin{aligned} \delta_t &= R_{t+1} + \gamma\hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t) \\ &= 0 + \gamma 2w_t - w_t \\ &= (2\gamma - 1)w_t \end{aligned} \tag{11.7}$$

off-policy semi-gradient TD(0) 更新公式：

$$\begin{aligned} w_{t+1} &= w_t + \alpha\rho_t\delta_t\nabla\hat{v}(S_t, w_t) \\ &= w_t + \alpha\cdot 1\cdot(2\gamma - 1)w_t\cdot 1 \\ &= (1 + \alpha(2\gamma - 1))w_t \end{aligned} \tag{11.8}$$

其中重要性采样比率 $\rho_t = 1$。

这个例子的关键是，对于一个转化 (行为策略)，它重复发生时 $w$ 在目标策略上没有发生更新。发生这样的原因是由于行为策略可能采样时会选择目标策略可能永远不会选择的行为。这说明了目标策略和行为策略会造成差异。

为了减小这种差异，有两种途径：

- 采用重要性采样，来平衡行为策略之中的权重。
- 采用一个不依赖于样本的随机梯度下降策略。

## 11.3 The Deadly Triad

我们结合下面三个要素容易造成系统的不稳定与差距，所以我们称之为 (The Deadly Triad):

- **Function approximation**: 一种强大、可扩展的方法，使用大量的空间和计算资源来生成状态空间，（如线性函数近似、人工神经网络）；
- **Bootstrapping**: 更新包括现有估计的目标时完全依靠实际回报，和完全的回报。(如 DP 算法和 TD 算法)
- **Off-policy training**: 离策略指的是关于目标策略之外的另外的训练，如 DP 算法一样，扫描所有的状态空间并更新所有的状态，这个行为不需要遵循目标策略。

这个误差是无法避免的，至于解决方法到目前为止也没有太好的办法，只能是采用上节所提到的途径:

- 采用重要性采样，来平衡行为策略之中的权重。
- 采用一个不依赖于样本的随机梯度下降策略。

## 11.4 Linear Value-function Geometry

我们要注意的是，在绝大多数情况下，大多数价值函数并不符合任何的策略。对我们而言，更多的并不是需要通过函数值进行逼近，而是通过设计比状态量更少的参数 (More important for our purposes is that most are not representable by the function approximator, which by design has far fewer parameters than there are states.)??

我们给出三个状态 $\mathcal{S} = \{s_1, s_2, s_3\}$ 和两个参数 $\mathbf{w} = (w_1, w_2)^T$。在一个三维空间中，对于任意一组数 $(x, y)$，我们设 $w_1 = x, w_2 = y$ 那么可以得到下图：

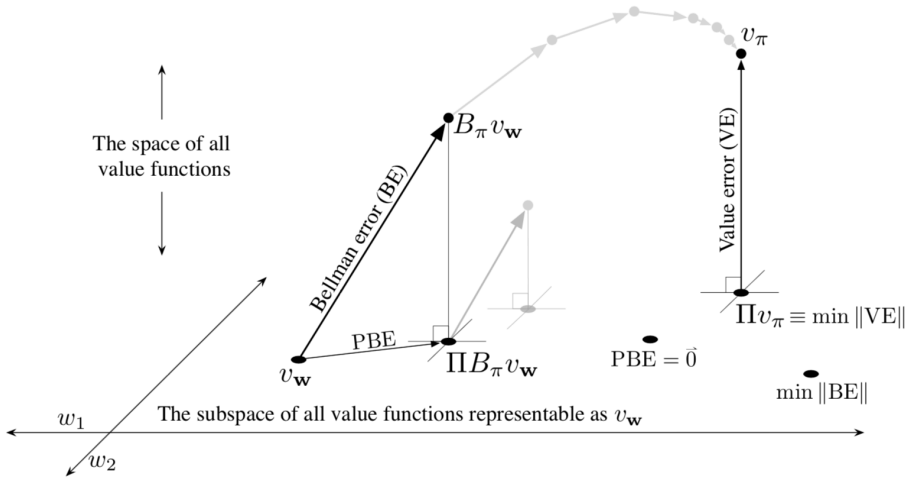



Figure 11.3: The geometry of linear value-function approximation. Shown as a plane is the subspace of all functions representable by the function approximator. The three-dimensional space above and below it is the much larger space of all value functions (functions from $\mathcal{S}$ to $\mathbb{R}$). The true value function $v_\pi$ is in this larger space and projects down to its best approximation in the value error (VE) sense. The best approximators in the Bellman error (BE) and projected Bellman error (PBE) senses are different and are also shown in the lower right. (VE, BE, and PBE are all treated as the corresponding vectors in this figure.) The Bellman operator takes a value function in the plane to one outside, which can then be projected back. If you could iteratively apply the Bellman operator outside the space (shown in gray above) you would reach the true value function, as in conventional DP.

假设一个固定的额策略 $\pi$，假设它的真实状态值函数是 $v_\pi$。为了度量两个不同的价值函数 (目标策略和行为策略)，有 $v = v_1 - v_2$。我们可以用权

重值 $\mu : \mathcal{S} \to \mathbb{R}$ 权衡我们关心的不同状态的精确量化。综上，我们可以定义如下公式用来衡量状态值函数之间的差异：

$$\|v\|_\mu^2 \doteq \sum_{s \in \mathcal{S}} \mu(s) v(s)^2 \tag{11.9}$$

引入均方误差 (MSVE) 有:

$$\text{MSVE}(\mathbf{w}) = \|v_\mathbf{w} - v_\pi\|_\mu^2 \tag{11.10}$$

我们可以定义一个操作 $\Pi$ 来表示最接近我们要求的状态值函数:

$$\begin{aligned} \Pi v &\doteq v_\mathbf{w} \\ &\text{where,} \\ \mathbf{w} &= \arg\min_\mathbf{w} \|v - v_\mathbf{w}\|_\mu^2 \end{aligned} \tag{11.11}$$

最接近真实值函数 $v_\pi$ 是 $\Pi$ 的投影，如下面所示：

The projection matrix

对于一个线性函数逼近，它的投影也是线性的，可以写为一个 $|\mathcal{S}| \times |\mathcal{S}|$ 的矩阵：

$$\Pi \doteq \mathbf{X} \left( \mathbf{X}^T D \mathbf{X} \right)^{-1} \mathbf{X}^T D \tag{11.12}$$

其中:

$$D \doteq \begin{bmatrix} \mu(1) & & & 0 \\ & \mu(2) & & \\ & & \ddots & \\ 0 & & & \mu(|\mathcal{S}|) \end{bmatrix} \tag{11.13}$$

$$\mathbf{X} \doteq \begin{bmatrix} -\mathbf{x}(1)^T - \\ -\mathbf{x}(2)^T - \\ \vdots \\ -\mathbf{x}(|\mathcal{S}|)^T - \end{bmatrix} \tag{11.14}$$

若上述存在广义逆矩阵，那么标准向量可以写为:

$$\|v\|_\mu^2 = v^T D v \tag{11.15}$$

线性逼近价值函数可以重写为:

$$v_{\mathbf{w}} = \mathbf{X}\mathbf{w} \tag{11.16}$$

在 TD 算法中，$v_\pi$ 可以写为:

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')], \quad \forall s \in \mathcal{S} \tag{11.17}$$

我们可以定义出 Bellman error:

$$\begin{aligned} \overline{\delta}_{\mathbf{w}}(s) &\doteq \left( \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_{\mathbf{w}}(s')] \right) - v_{\mathbf{w}}(s) \\ &= \mathbb{E}[R_{t+1} + \gamma v_{\mathbf{w}}(S_{t+1}) - v_{\mathbf{w}}(S_t)|S_t = s, A_t \sim \pi] \end{aligned} \tag{11.18}$$

这个公式说明了 Bellman error 是 TD error 的期望。引入均方 Bellman 误差 (Mean Squared Bellman Error, MSBE) 和 Bellman 误差向量, 有:

$$\text{MSBE}(\mathbf{w}) = \|\overline{\delta}_{\mathbf{w}}\|_\mu^2 \tag{11.19}$$

定义一个 Bellman 操作:

$$\begin{aligned} &(B_\pi v)(s) \doteq \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v(s')], \\ &\forall s \in \mathcal{S}, \quad \forall v : \mathcal{S} \to \mathbb{R} \end{aligned} \tag{11.20}$$

则 Bellman 误差向量可写作:

$$\overline{\delta}_{\mathbf{w}} = B_\pi v_{\mathbf{w}} - v_{\mathbf{w}} \tag{11.21}$$

其 Bellman 均方投影误差 (Mean Square Projected Bellman Error, MSPBE):

$$\text{MSPBE}(\mathbf{w}) = \|\Pi\overline{\delta}_{\mathbf{w}}\|_\mu^2 \tag{11.22}$$

## 11.5 Stochastic Gradient Descent in the Bellman Error

随机梯度下降 (Stochastic Gradient Descent, SGD) 具有较好的鲁棒性，能够较快地收敛，但是速度慢于半随机梯度下降 (semi-SGD)。

在 TD 学习中, 定义均方 TD 误差 (Mean Squared TD Error, MSTDE):

$$\begin{aligned}\text{MSTDE}(\mathbf{w}) &= \sum_{s\in\mathcal{S}} \mu(s)\mathbb{E}[\delta_t^2|S_t=s, A_t\sim\pi] \\ &= \sum_{s\in\mathcal{S}} \mu(s)\mathbb{E}[\rho_t\delta_t^2|S_t=s, A_t\sim b] \\ &= \mathbb{E}_b[\rho_t\delta_t^2]\end{aligned} \tag{11.23}$$

那么对应的更新步骤可以写为:

$$\begin{aligned}\mathbf{w}_{t+1} &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla(\rho_t\delta_t^2) \\ &= \mathbf{w}_t - \alpha\rho_t\delta_t\nabla\delta_t \\ &= \mathbf{w}_t + \alpha\rho_t\delta_t\left(\nabla\hat{v}(S_t,\mathbf{w}_t) - \gamma\nabla\hat{v}(S_t,\mathbf{w}_t)\right)\end{aligned} \tag{11.24}$$

这个算法被称为 naive residual-gradient 算法。虽然这个算法有较强的鲁棒性，但是常常收敛不到较好的值，对此比较好的是采用 Bellman 误差，对应的更新公式可以写为:

$$\begin{aligned}\mathbf{w}_{t+1} &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla(\mathbb{E}_\pi[\delta_t]^2) \\ &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla(\mathbb{E}_b[\rho_t\delta_t]^2) \\ &= \mathbf{w}_t - \alpha\mathbb{E}_b[\rho_t\delta_t]\nabla\mathcal{E}_b[\rho_t\delta_t] \\ &= \mathbf{w}_t - \alpha\mathbb{E}_b[\rho_t(R_{t+1}+\gamma\hat{v}(S_{t+1},\mathbf{w}) - \hat{v}(S_t,\mathbf{w}))]\mathbb{E}_b[\rho_t\nabla\delta_t] \\ &= \mathbf{w}_t + \alpha\left[\mathbb{E}_b[\rho_t(R_{t+1}+\gamma\hat{v}(S_{t+1},\mathbf{w})) - \hat{v}(S_t,\mathbf{w})\right]\left[\nabla\hat{v}(S_t,\mathbf{w}) - \gamma\mathbb{E}_b[\rho_t\nabla\hat{v}(S_{t+1},\mathbf{w})]\right]\end{aligned} \tag{11.25}$$

这种算法也称为残差梯度 (residual gradient)

## 11.6 Learnability of the Bellman Error

如果不能从现有的特征向量、动作或者奖励中进行计算和估计，那么我们称之为不能学习的 (not *learnable*). Bellman 误差就是一个不可学习过程。

首先定义两个马尔可夫奖励过程 (Markov reward processes, MRP):


其中数字代表的就是 reward，所有状态都是等概率的。看右边的图，转移回本状态 reward=0，转移到另一状态 reward=2，那么平均 reward=1，均方误差在两个图上为 0. 定义均方回报误差 (Mean Square Return Error, MSRE):

$$\begin{aligned}\text{MSRE}(\mathbf{w}) &= \mathbb{E}\left[(G_t - \hat{v}(S_t, \mathbf{w}))^2\right] \\ &= \text{MSVE}(\mathbf{w}) + \mathbb{E}\left[(G_t - v_\pi(S_t))^2\right]\end{aligned} \tag{11.26}$$

MSVE 和 MSRE 两者都不倚重权重参数向量，这说明了均方价值误差 MSVE 是不可学习的，而 MSRE 是可学习的。

总之均方 Bellman 误差 MSBE 是不可学习的，不能被可观察到的数值或者是参数中进行估计和计算，说明通过最小化 MSBE 来收敛到较优解是不可行的。我们应该研究的是均方 Bellman 投影误差 (MSPBE).

## 11.7 Gradient-TD Methods

我们通过随机梯度下降 (SGD) 来最小化均方 Bellman 投影误差 (MSPBE), 梯度 TD 算法在离策略和非线性函数逼近中具有较好的鲁棒性，在 TD 算法中引入随机梯度下降之后，算法复杂度将由 $O(d^2)$ 下降为 $O(d)$。公式可

以写为：

$$\begin{aligned}\text{MSPBE}(\mathbf{w}) &= \|\Pi\overline{\delta}_{\mathbf{w}}\|_{\mu}^{2} \\ &= (\Pi\overline{\delta}_{\mathbf{w}})^{T} D\Pi\overline{\delta}_{\mathbf{w}} \\ &= \overline{\delta}_{\mathbf{w}}^{T}\Pi^{T} D\Pi\overline{\delta}_{\mathbf{w}} \\ &= \overline{\delta}_{\mathbf{w}}^{T} D\mathbf{X}(\mathbf{X}^{T}D\mathbf{X})^{-1}\mathbf{X}^{T}D\overline{\delta}_{\mathbf{w}} \\ &= (\mathbf{X}^{T}D\overline{\delta}_{\mathbf{w}})^{T}(\mathbf{X}^{T}D\mathbf{X})^{-1}(\mathbf{X}^{T}D\overline{\delta}_{\mathbf{w}})\end{aligned} \tag{11.27}$$

其中：

$$\Pi^{T}D\Pi = D\mathbf{X}(\mathbf{X}^{T}DX)^{-1}\mathbf{X}^{T}D \tag{11.28}$$

那么 $\mathbf{w}$ 的梯度可以写为：

$$\nabla\text{MSPBE}(\mathbf{w}) = 2\nabla\left[\mathbf{X}^{T}D\overline{\delta}_{\mathbf{w}}\right]^{T}(\mathbf{X}^{T}D\mathbf{X})^{-1}(\mathbf{X}^{T}D\overline{\delta}_{\mathbf{w}}) \tag{11.29}$$

假设 $\mu$ 是访问行为策略的分布，那么：

$$\mathbf{X}^{T}D\overline{\delta}_{\mathbf{w}} = \sum_{s}\mu(s)\mathbf{x}(s)\overline{\delta}_{\mathbf{w}}(s) = \mathbb{E}[\mathbf{x}_{t}\rho_{t}\delta_{t}] \tag{11.30}$$

对应的梯度：

$$\begin{aligned}\nabla\mathbb{E}[\mathbf{x}_{t}\rho_{t}\delta_{t}]^{T} &= \mathbb{E}\left[\rho_{t}\nabla\delta_{t}^{T}\mathbf{x}_{t}^{T}\right] \\ &= \mathbb{E}\left[\rho_{t}\nabla(R_{t+1}+\gamma\mathbf{w}^{T}\mathbf{x}_{t+1}-\mathbf{w}^{T}\mathbf{x}_{t})^{T}\mathbf{x}_{t}^{T}\right] \\ &= \mathbb{E}\left[\rho_{t}(\gamma\mathbf{x}_{t+1}-\mathbf{x}_{t})\mathbf{x}_{t}^{T}\right]\end{aligned} \tag{11.31}$$

综上：

$$\nabla\text{MSPBE}(\mathbf{w}) = 2\mathbb{E}\left[\rho_{t}(\gamma\mathbf{x}_{t+1}-\mathbf{x}_{t})\mathbf{x}_{t}^{T}\right]\mathbb{E}\left[\mathbf{x}_{t}\mathbf{x}_{t}^{T}\right]^{-1}\mathbb{E}[\mathbf{x}_{t}\rho_{t}\delta_{t}] \tag{11.32}$$

上面的公式说明，即使是 MSPBE，也是需要下一个特征向量 $\mathbf{w}_{t+1}$。导致我们不能够简单地进行采样，求取期望。

在梯度 TD 算法中，分别存储一些估计，并且与采样的样本进行结合，是一种解决问题的思路。我们将要存储的信息设为 $\mathbf{v}$:

$$\mathbf{v} \approx \mathbb{E}[\mathbf{x}_{t}\mathbf{x}_{t}^{T}]^{-1}\mathbb{E}[\mathbf{x}_{t}\rho_{t}\delta_{t}] \tag{11.33}$$

最小化方差误差 (Least Mean Square,LMS) 为:

$$(\mathbf{v}^T\mathbf{x}_t - \rho_t\delta_t)^2 \tag{11.34}$$

那么特征向量的更新公式可以写为:

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \beta\rho_t(\delta_t - \mathbf{v}_t^T\mathbf{x}_t)\mathbf{x}_t \tag{11.35}$$

那么权重向量的更新就可以写为:

$$\begin{aligned}
\mathbf{w}_{t+1} &= \mathbf{w}_t - \frac{1}{2}\alpha\nabla\text{MSPBE}(\mathbf{w}_t) \\
&= \mathbf{w}_t - \frac{1}{2}\alpha 2\mathbb{E}\left[\rho_t(\gamma\mathbf{x}_{t+1} - \mathbf{x}_t)\mathbf{x}_t^T\right]\mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^T\right]^{-1}\mathbb{E}[\mathbf{x}_t\rho_t\delta_t] \\
&= \mathbf{w}_t + \alpha\mathbb{E}\left[\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^T\right]\mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^T\right]^{-1}\mathbb{E}[\mathbf{x}_t\rho_t\delta_t] \\
&= \mathbf{w}_t + \alpha\mathbb{E}\left[\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^T\right]\mathbf{v}_t \\
&= \mathbf{w}_t + \alpha\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^T\mathbf{v}_t
\end{aligned} \tag{11.36}$$

这个算法被称为 $GTD2$, 对于该算法，还可以有一点提升:

$$\begin{aligned}
\mathbf{w}_{t+1} &= \mathbf{w}_t + \alpha\mathbb{E}\left[\rho_t(\mathbf{x}_t - \gamma\mathbf{x}_{t+1})\mathbf{x}_t^T\right]\mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^T\right]^{-1}\mathbb{E}[\mathbf{x}_t\rho_t\delta_t] \\
&= \mathbf{w}_t + \alpha\left(\mathbb{E}\left[\rho_t\mathbf{x}_t\mathbf{x}_t^T\right] - \gamma\mathbb{E}\left[\rho_t\mathbf{x}_{t+1}\mathbf{x}_t^T\right]\right)\mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^T\right]^{-1}\mathbb{E}\left[\mathbf{x}_t\rho_t\delta_t\right] \\
&= \mathbf{w}_t + \alpha\left(\mathbb{E}\left[\mathbf{x}_t\rho_t\delta_t\right] - \gamma\mathbb{E}\left[\rho_t\mathbf{x}_{t+1}\mathbf{x}_t^T\right]\mathbb{E}\left[\mathbf{x}_t\mathbf{x}_t^T\right]^{-1}\mathbb{E}\left[\mathbf{x}_t\rho_t\delta_t\right]\right) \\
&= \mathbf{w}_t + \alpha\left(\mathbb{E}\left[\mathbf{x}_t\rho_t\delta_t\right] - \gamma\mathbb{E}\left[\rho_t\mathbf{x}_{t+1}\mathbf{x}_t^T\right]\mathbf{v}_t\right) \\
&= \mathbf{w}_t + \alpha\rho_t\left(\delta_t\mathbf{x}_t - \gamma\mathbf{x}_{t+1}\mathbf{x}_t^T\mathbf{v}_t\right)
\end{aligned} \tag{11.37}$$

这个算法被称为 $GTD(0)$ 或者 TDC。

GTD2 和 TDC 都包括两个学习进程，主要第一个的 $\mathbf{w}$ 和第二个 $\mathbf{v}$。如果主要的第一个学习有第二个影响，而第二个不会比第一个先进行，那么这种关系就称为级联 (cascade)。

综上，梯度 TD 算法 (GTD) 是当前最好理解并且最稳定的使用最广泛的离策略。

## 11.8 Emphatic-TD Methods

重要性 TD 算法 (Emphatic-TD Methods,ETD)，其中 one-step emphatic-TD 算法的参数可以写为：

$$\begin{aligned}\delta_t &= R_{t+1} + \gamma \hat{v}(S_{t+1}, \mathbf{w}_t) - \hat{v}(S_t, \mathbf{w}_t) \\ \mathbf{w}_{t+1} &= \mathbf{w}_t + \alpha M_t \rho_t \delta_t \nabla \hat{v}(S_t, \mathbf{w}_t) \\ M_t &= \gamma \rho_{t-1} M_{t-1} + I_t\end{aligned} \tag{11.38}$$

$I_t$ 表示 $t$ 时刻的兴趣 (interest)，是一个随机的值；$M_t$ 表示重要性 (emphasis)，其中初始化 $M_{t-1} = 0$。

在接下来的实验中，ETD 被证明出分歧 (variance) 太大，而无法进行实用。

## 11.9 Reducing Variance

off-policy 中分歧是不可避免的问题，在控制分歧，特别是离策略中需要十分谨慎。众所周知，重要性采样中重要性比率 ratios 通常期望为 1，但是在实际中 $ratios \gg 1$ 或者 $ratios \to 0$。理想中的 ratios 是互相之间无关联的。但是实际上就是这些相关联导致了目标策略和行为策略之间的分歧。

随机梯度下降 (SGD) 通过多次小步的计算得到良好的梯度，这样可以有效的减少分歧，但是 SGD 必须拥有比较大的样本才容易收敛，若样本过于单一的话，得到的结果也将变得不完善了。但是要注意的是，如果步长太小了的话，会导致预期的步骤变得非常小，计算的效率会特别地低。