1 一些和强化学习相关的证明(一)

1.1 价值函数定义

我们定义agent与环境交互的模式如下：

$$S_t \xrightarrow{A_t} R_{t+1}, S_{t+1} \xrightarrow{A_{t+1}} R_{t+2}, S_{t+2} \tag{1}$$

我们假设agent与环境交互时的动作选择过程遵循Markov Decision Process. 即当前状态和奖励只受上一个状态和其动作选择的影响:

$$p(s', r|s, a) = p(S_{t+1} = s, R_{t+1} = r|S_t = s, A_t = a) \tag{2}$$

π代表agent在与环境交互时遵循的策略, 它是一个动作选择的概率分布. 对于$s \in S$:

$$P_\pi \sim P(A_t|s) \tag{3}$$

我们定义状态$s \in S$的价值为:

$$v_\pi(s) \doteq E_\pi[G_t|S_t = s] \tag{4}$$

其中G_t代表从状态s出发的后续所有奖励，它是后续每一步奖励随机变量$\{R_{t+i}\}_{i=1,..}$的总和.

$$G_t = \sum_{i=1}^{\infty} R_{t+i}\gamma^{i-1} = R_{t+1} + \gamma G_{t+1} \tag{5}$$

1.2 价值函数展开

根据(5)，我们有：

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] = E_\pi[R_{t+1}|S_t = s] + \gamma E_\pi[G_{t+1}|S_t = s] \tag{6}$$

$E_\pi[R_{t+1}|S_t = s]$代表状态s在遵循策略π时的即时期望奖励:

$$E_\pi[R_{t+1}|S_t = s] = \sum_a \pi(a|s) \sum_r p(r|s, a)r \tag{7}$$

$E_\pi[G_{t+1}|S_t = s]$代表代表状态s在遵循策略π时的未来期望奖励:

$$E_\pi[G_{t+1}|S_t = s] = \sum_a \pi(a|s) \sum_{s'} p(r|s, a)E_\pi[G_{t+1}|S_{t+1} = s'] = \sum_a \pi(a|s) \sum_{s'} p(r|s, a)v_\pi(s') \tag{8}$$

把上述的展开合并到一起，就可以得出价值函数最常用的展开形式：

$$v_\pi(s) = \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)[r + \gamma v_\pi(s')] \tag{9}$$

上述的展开形式还有可以进一步挖掘的地方,注意到:

$$E_\pi[G_{t+1}|S_t = s] = \sum_a \pi(a|s) \sum_{s'} p(s'|s, a)v_\pi(s') = E_\pi[v_\pi(S_{t+1})|S_t = s] \tag{10}$$

我们可以推导出另一个形式:

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma G_{t+1}|S_t = s] = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s] \tag{11}$$

我们可以递推扩展:

$$E_\pi[G_{t+1}|S_t = s] = E_\pi[R_{t+2} + \gamma G_{t+2}|S_t = s] \tag{12}$$

$$E_\pi[G_{t+2}|S_t = s] = \sum_a \pi(a|s) \sum_{s'} p(r|s,a) E_\pi[G_{t+2}|S_{t+1} = s'] \tag{13}$$

根据(10)可得:

$$E_\pi[G_{t+2}|S_{t+1} = s'] = \sum_a \pi(a|s) \sum_{s'} p(r|s,a) E_\pi[v_\pi(S_{t+2})|S_{t+1} = s'] = E_\pi[v_\pi(S_{t+2})|S_t = s] \tag{14}$$

我们最后可以总结出以下递推展开规律:

$$v_\pi(s) = E_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s] = E_\pi[R_{t+1} + \gamma R_{t+2} + \gamma^2 v_\pi(S_{t+2})|S_t = s] \tag{15}$$

1.3 策略改进

假设我们现在遵循确定性策略π, 我们如何改进策略π? 对于这个问题，首先我们需要明确如何比较两个策略的优劣。

对于当前价值函数集合$v_{\pi'}(S)$。如果我们更换为确定性策略π', 对于遵循策略π'的价值函数集合$v_{\pi'}(S)$,如果对于任意$s \in S$有:

$$v_{\pi'}(s) \geq v_\pi(s) \tag{16}$$

我们就可以说策略π'优于π. 但是我们如何才能找到这个策略π'呢? 下面我先对改进方法进行直观的阐述。

假设我们现在有一个策略π':

$$\pi'(s) = \begin{cases} \pi(s) & \text{if } s \neq s_k, \\ a \neq \pi(s) & \text{if } s = s_k. \end{cases} \tag{17}$$

设$q(\cdot,\cdot)$代表固定状态和动作后的价值函数, 我们可以自然的推导出:

$$q(s, \pi'(s)) = \begin{cases} v_\pi(s) & \text{if } s \neq s_k, \\ q(s_k, a) & \text{if } s = s_k. \end{cases} \tag{18}$$

如果$q(s_k, a) > v_\pi(s)$, 实际上我们就可以说策略π'优于π. 而且π'在s_k上的局部优势在求解新的Bellmna方程后会扩散到每一个与s_k有直接或间接联系的s上。将状态转移图分解成多个独立子图，如果状态s和s_k在同一个子图上，那么价值函数$v(s)$会得到提升:

$$v_{\pi'}(s) > v_\pi(s) \tag{19}$$

如果状态s和s_k不在同一个子图上，那么价值函数$v(s)$得到保持:

$$v_{\pi'}(s) = v_\pi(s) \tag{20}$$

对于价值函数集合v_π，如果策略π'在一系列状态$\{s_k\}$上都有$q_\pi(s_k, \pi'(s)) > v_\pi(s)$. 那么与$s_k$同在一个子图上的其他状态的价值函数也都会得到提升。

现在让我们整理一下这个尚未证明的直观结论。对于任意$s \in S$,如果有:

$$q_\pi(s, \pi'(s)) \geq v_\pi(s) \tag{21}$$

那么我们可以得到一个更优的价值函数集合, 对于任意$s \in S$，我们有:

$$v_{\pi'}(s) \geq v_\pi(s) \tag{22}$$

Proof 1

我们先引入一个可以迭代求解价值函数的方法: Iterative Policy Evaluation.

Iterative Policy Evaluation, for estimating $V \approx v_\pi$

Input π, the policy to be evaluated
Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
  $\Delta \leftarrow 0$
  Loop for each $s \in \mathcal{S}$:
    $v \leftarrow V(s)$
    $V(s) \leftarrow \sum_a \pi(a|s) \sum_{s',r} p(s', r|s, a)\big[r + \gamma V(s')\big]$
    $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Figure 1: policy evaluation

如图所示, Iterative Policy Evaluation可以从任意价值函数集合出发, 用指定策略不断修正价值函数集合，最终得到遵循指定策略的价值函数集合。

我们现在从$v_\pi(S)$出发,用策略π'进行Iterative Policy Evaluation得到$v_{\pi'}(S)$. 如果迭代的每一步我们的价值函数集合都能得到提升, 那么我们就完成了证明。 设$v_\pi(S) = v(S)_0$, 那么对于任意$s \in S$, 在第一个循环:

$$\sum_{r,s'} p(r, s'|s, \pi'(s))[r + \gamma v(s')_0] = q(s, \pi'(s)) \tag{23}$$

$$v(s)_1 = q(s, \pi'(s)) \geq v(s)_0 = v_\pi(s) \tag{24}$$

对于任意$s \in S$, 在第二个循环中我们有:

$$\begin{aligned} v(s)_2 &= \sum_{r,s'} p(r, s'|s, \pi'(s))[r + \gamma v(s')_1] \\ &= \sum_{r,s'} p(r, s'|s, \pi'(s))[r + \gamma q(s, \pi'(s))] \\ &\geq \sum_{r,s'} p(r, s'|s, \pi'(s))[r + \gamma v(s)_0] \\ &= v(s)_1 \end{aligned} \tag{25}$$

对于第三个循环:

$$\begin{aligned} v(s)_3 &= \sum_{r,s'} p(r, s'|s, \pi'(s))[r + \gamma v(s')_2] \\ &\geq \sum_{r,s'} p(r, s'|s, \pi'(s))[r + \gamma v(s')_1] \\ &= v(s)_2 \end{aligned} \tag{26}$$

依次类推, 在迭代的过程中我们的$v(s)_t$保持递增直至收敛到$v_{\pi'}(s)$. QED.

对于策略π和状态s, 所有满足$q_\pi(s, A_t) > v_\pi(s)$的动作修改都可以提升价值函数. 将每一个状态s对应的动作修改串联起来就可以求出π'. 为了榨干提升空间，我们可以直接采用贪婪策略求出π':

$$a = argmax_{a'} q_\pi(s, a') \tag{27}$$

最后我们得到了一个Pipeline:

$$v_{\pi_0}(s) \xrightarrow{\pi_1^*} v_{\pi_1}(s) \xrightarrow{\pi_2^*} \xrightarrow{\pi_T^*} v_{\pi_T}(s) \approx v_*(s) \tag{28}$$

对每一步的稳定价值函数集合$v_{\pi_t}(S)$, 我们都用贪婪策略求出新的策略π_{t+1}^*, 随后进行Iterative Policy Evaluation得到$v_{\pi_{t+1}}(S)$, 周而复始。 这实际上形成了如下的评估-改进框架:

Policy Iteration (using iterative policy evaluation) for estimating $\pi \approx \pi_*$

1. Initialization
   $V(s) \in \mathbb{R}$ and $\pi(s) \in \mathcal{A}(s)$ arbitrarily for all $s \in \mathcal{S}$

2. Policy Evaluation
   Loop:
   $\Delta \leftarrow 0$
   Loop for each $s \in \mathcal{S}$:
   $v \leftarrow V(s)$
   $V(s) \leftarrow \sum_{s',r} p(s', r \mid s, \pi(s)) \big[r + \gamma V(s')\big]$
   $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
   until $\Delta < \theta$ (a small positive number determining the accuracy of estimation)

3. Policy Improvement
   policy-stable $\leftarrow$ *true*
   For each $s \in \mathcal{S}$:
   old-action $\leftarrow \pi(s)$
   $\pi(s) \leftarrow \operatorname{argmax}_a \sum_{s',r} p(s', r \mid s, a) \big[r + \gamma V(s')\big]$
   If *old-action* $\neq \pi(s)$, then *policy-stable* $\leftarrow$ $false$
   If *policy-stable*, then stop and return $V \approx v_*$ and $\pi \approx \pi_*$; else go to 2

Figure 2: Iterative Policy Evaluation

评估-改进框架的是否可以收敛并收敛到到最优呢? 我们先给出最优价值函数$v^*(s)$和最优价值函数$q^*(s, a)$的定义:

$$v^*(s) \doteq max_\pi v_\pi(x) \tag{29}$$

$$q^*(s, a) \doteq max_\pi q_\pi(s, a) \tag{30}$$

Proof 2

如果评估-改进框架不收敛，那么显然框架每次迭代都会产生一个与当前策略不同的新策略，即$\exists s \in S$:

$$\pi_{t+1}(s) \neq \pi_t(s) \tag{31}$$

新策略π_{t+1}也一定在$t = 1..t-1$中没有出现过. 假设$\exists k \leq t-1, \forall s \in S$:

$$\pi_k(s) = \pi_t(s) \tag{32}$$

而$k < t' < t, \forall s \in S$满足:

$$\pi_{t'}(s) \neq \pi_t(s) \tag{33}$$

根据Proof 1, 当条件严格成立时结论也严格成立. 那么我们有如下不等式关系:

$$v_{\pi_k}(s) < v_{\pi_{k+1}}(s) < \cdots < v_{\pi_{t-1}}(s) < v_{\pi_t}(s) = v_{\pi_k}(s) \tag{34}$$

这显然是矛盾的，所以我们每次迭代都会产生一个与之前过往策略都不同的新策略。这隐含着策略空间是无限的。但我们考虑的是离散状态和动作空间，所以策略数量必然是有限的。故评估-改进框架一定会收敛。

当框架收敛时，此时$\forall s \in S$均有:

$$v_{\pi_{T+1}}(s) = max_a q_{\pi_T}(s,a) = v_{\pi_T}(s) \tag{35}$$

之后就是递归重复:

$$v_{\pi_T}(s) = v_{\pi_{T+1}}(s) = v_{\pi_\infty}(s) \tag{36}$$

那么评估-改进框架能否收敛到最优价值函数集合呢？我们继续在1.2节的证明上进行讨论. 我们已知框架在π_T处收敛. 那么$\forall \pi' \in \{\pi\}$, $\forall s \in S$, 策略π_T满足:

$$q_{\pi_T}(s, \pi'(s)) \leq v_{\pi_T}(s) = max_a q_{\pi_T}(s,a) \tag{37}$$

我们将1.2节的证明中所有不等式符号都反过来就可以得到$\forall s \in S$:

$$v_{\pi'}(s) \leq v_{\pi_T}(s) \Leftrightarrow v_{\pi_T}(s) = max_\pi v_\pi(x) \tag{38}$$

根据定义(29)我们可知π_T是最优策略.这说明我们的评估-改进框架不仅收敛, 还可以收敛到最优价值函数集合。

Proof 2的证明过程还可以引出许多有趣的结论:

$$q_{\pi_T}(s,a) = \sum_{s',r} p(s',r|s,a)[r + v_{\pi_T}(s')] = max_\pi \sum_{s',r} p(s',r|s,a)[r + v_\pi(s')] = max_\pi q_\pi(s,a) \tag{39}$$

$$q_{\pi_T}(s,a) = q^*(s,a) \tag{40}$$

$$v^*(s) = max_a q_{\pi_T}(s,a) = max_a q^*(s,a) \tag{41}$$

1.4 价值迭代

我们在1.3节的最后得到了如下结论:

$$v^*(s) = max_a q^*(s,a) = max_a \sum_{s',r} p(s',r|s,a)[r + v^*(s')] \tag{42}$$

(42)类似于Ballman方程, 虽然无法直接求解。但我们可以继续使用Iterative Policy Evaluation进行求解。

Value Iteration, for estimating $\pi \approx \pi_*$

Algorithm parameter: a small threshold $\theta > 0$ determining accuracy of estimation
Initialize $V(s)$, for all $s \in \mathcal{S}^+$, arbitrarily except that $V(terminal) = 0$

Loop:
| $\Delta \leftarrow 0$
| Loop for each $s \in \mathcal{S}$:
| $v \leftarrow V(s)$
| $V(s) \leftarrow \max_a \sum_{s',r} p(s',r|s,a)[r + \gamma V(s')]$
| $\Delta \leftarrow \max(\Delta, |v - V(s)|)$
until $\Delta < \theta$

Output a deterministic policy, $\pi \approx \pi_*$, such that
  $\pi(s) = \operatorname{argmax}_a \sum_{s',r} p(s',r|s,a)[r + \gamma V(s')]$

Figure 3: Value Evaluation

上图过程被称为价值迭代。与1.3节的评估-改进框架相比，价值迭代框架对每种策略只做一次近似评估从而大大简化了过程。

2 一些和强化学习相关的证明(二)

第一章节中我们介绍了Model based methods: Policy iteration 和 Value iteration. 这两个方法都要求我们了解环境的完整建模, 即状态转移概率:$p(S_{t+1}, R_{t+1}|S_t, A_t)$. 但对于很多情况，我们对环境的了解并不这么充分，我们仅能获取到情节样本. 此时我们需要考虑Model free method.

Model free methods分为两种类别: Value-based methods和Policy-based methods. 前者先利用一系列情节样本去估计拟合价值函数，然后根据价值函数集合执行贪婪策略. 而后者则直接拟合最佳策略. **本章节主要介绍Policy-based methods**. 一个很自然的问题是: 与Value-based methods相比, Policy-based methods有哪些优势?

首先对于许多环境，最佳策略往往不是确定性的，而是一个概率分布. Value-based methods最后得到的是确定性的贪婪策略, 而Policy-based methods直接拟合策略，可以拟合出服从概率分布的最佳策略.此外Value-based methods鲁棒性较差, 因为Value-based methods对价值函数非常敏感. 微小的估计误差就会产生完全不同的贪婪策略. 而直接拟合策略可以在一定程度上提高鲁棒性.

我在下一节会介绍Value-based methods. 包括Monte Carlo methods和Temporal Difference(Sarsa, QL, DQL, DQN, DDQN). 本章节主要介绍Stochastic policy gradient theorem和Deterministic policy gradient (DPG) theorem.

策略拟合的核心策略是**随机梯度上升：Stochastic gradient ascent**.我们需要定义策略π的参数化形式π_θ和度量函数$J(\theta)$. π_θ可以用DNN网络表示,度量函数$J(\theta)$则用如下定义:

$$J_\theta \doteq \int_s p(s) v_\pi(s)\, ds \tag{43}$$

其中$p(S)$是状态的初始分布, 也就是每个状态s被选为情节起点的概率. 也就是说我们使用价值函数期望作为策略度量函数$J(\theta)$.

下一个需要思考的问题是: 我们如何根据已有的情节样本，去计算$\nabla J(\theta)$? 虽然Pytorch的自动微分框架可以自动计算梯度，但我们更希望得到一个显式的梯度计算模式. 显式的梯度计算模式让我们可以更灵活的编辑、优化算法.

下面我先介绍教材Chapter 13给出的梯度推导方法，然后再给出一个更直观的梯度推导方式.

书中P325页记录了如下策略梯度公式的推导。这里没有考虑折扣因子γ的影响，即$\gamma = 1$. 而且对度量函数的定义也和公式(43)不太一样，教材只关注给定起始点的价值函数$v_\pi(s_0)$. 不过这个区别没什么影响.

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a \nabla \pi(a|s) q_\pi(s,a) \tag{44}$$

Proof

$$\begin{aligned}
\nabla J(\theta) &= \nabla v_\pi(s_0) && (45)\\
&= \nabla \sum_a \pi(a|s_0) q_\pi(s_0,a) && (46)\\
&= \sum_a \nabla \pi(a|s_0) q_\pi(s_0,a) + \pi(a|s_0) \nabla q_\pi(s_0,a) && (47)\\
&= \sum_a \nabla \pi(a|s_0) q_\pi(s_0,a) + \sum_a \pi(a|s_0) \nabla \sum_s p(s|s_0,a)[r + v_\pi(s)] && (48)\\
&= \sum_a \nabla \pi(a|s_0) q_\pi(s_0,a) + \sum_a \pi(a|s_0) \sum_s p(s|s_0,a) \nabla v_\pi(s) && (49)
\end{aligned}$$

我们先重写第一项:

$$\sum_a \nabla \pi(a|s_0) q_\pi(s_0,a) = \sum_s Pr(s_0 \rightarrow s, 0, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) \tag{50}$$

$Pr(s_0 \rightarrow s, 0, \pi)$代表遵循策略$\pi$时$s_0$转移0步到达状态$s$的概率:

$$\sum_s Pr(s_0 \rightarrow s, 0, \pi) = Pr(s_0 \rightarrow s_0, 0, \pi) = 1 \tag{51}$$

我们如法炮制重写第二项:

$$\sum_a \pi(a|s_0) \sum_{s'} p(s|s_0,a) \nabla v_\pi(s) = \sum_s Pr(s_0 \rightarrow s, 1, \pi) \cdot \nabla v_\pi(s) \tag{52}$$

综上我们可以将$\nabla J(\theta)$重写为:

$$\nabla J(\theta) = \sum_s Pr(s_0 \rightarrow s, 0, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) + \sum_s Pr(s_0 \rightarrow s, 1, \pi) \cdot \nabla v_\pi(s) \tag{53}$$

公式(53)的第二项可以继续扩展:

$$\begin{aligned}
\sum_s Pr(s_0 \rightarrow s, 1, \pi) \nabla v_\pi(s) &= \sum_s Pr(s_0 \rightarrow s, 1, \pi)[\sum_a \nabla \pi(a|s) q_\pi(s,a) + \sum_{s'} Pr(s \rightarrow s', 1, \pi) \nabla v_\pi(s')] && (54)\\
&= \sum_s Pr(s_0 \rightarrow s, 1, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) + \sum_s Pr(s_0 \rightarrow s, 2, \pi) \nabla v_\pi(s) && (55)
\end{aligned}$$

所以我们可以把$\nabla J(\theta)$扩展成:

$$\nabla J(\theta) = \sum_s Pr(s_0 \to s, 0, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) + \sum_s Pr(s_0 \to s, 1, \pi) \cdot \nabla v_\pi(s) \tag{56}$$

$$= \sum_s \sum_{k=0}^{1} Pr(s_0 \to s, k, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) + \sum_s Pr(s_0 \to s, 2, \pi) \cdot \nabla v_\pi(s) \tag{57}$$

$$= \sum_s \sum_{k=0}^{2} Pr(s_0 \to s, k, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) + \sum_s Pr(s_0 \to s, 3, \pi) \nabla v_\pi(s) \tag{58}$$

$$= \sum_s \sum_{k=0}^{\infty} Pr(s_0 \to s, k, \pi) \sum_a \nabla \pi(a|s) q_\pi(s,a) \tag{59}$$

$$= \sum_s [\sum_{k=0}^{\infty} Pr(s_0 \to s, k, \pi)] \cdot [\sum_a \nabla \pi(a|s) q_\pi(s,a)] \tag{60}$$

$\sum_{k=0}^{\infty} Pr(s_0 \to s, k, \pi)$是一个很有趣的无穷级数,因为我们可以把它写成Bernoulli分布期望的求和形式. 我们可以定义事件$X_k(s)$为状态s在第k步是否出现. 显然有$X_k(s) \sim \text{Bernoulli}(Pr(s_0 \to s, k, \pi))$:

$$X_k(s) = \begin{cases} 1 & \text{if } x = s \\ 0 & \text{if } x \neq s \end{cases} \tag{61}$$

$$E[X_k(s)] = Pr(s_0 \to s, k, \pi) \tag{62}$$

$$\sum_{k=0}^{\infty} Pr(s_0 \to s, k, \pi) = E[\sum_{k=0}^{\infty} X_k(s)] \tag{63}$$

所以$\sum_{k=0}^{\infty} Pr(s_0 \to s, k, \pi)$代表着状态$s$在以$s_0$为起点的情节中出现的次数。我们将其记为$\eta(s)$.

$$\nabla J(\theta) = \sum_s \eta(s) \sum_a \nabla \pi(a|s) q_\pi(s,a) \tag{64}$$

我们可以利用$\eta(s)$定义状态$s \in S$的概率分布$\mu(s)$:

$$\mu(s) = p_\pi(s) = \frac{\eta(s)}{\sum_{s'} \eta(s')} \tag{65}$$

于是公式(64)可写成:

$$\nabla J(\theta) = [\sum_{s'} \eta(s')] \cdot \sum_s \mu(s) \sum_a \nabla \pi(a|s) q_\pi(s,a) \tag{66}$$

这里$\sum_{s'} \eta(s')$是一个常数, 表示以s_0为起点时情节的平均长度. 最后我们得到:

$$\nabla J(\theta) \propto \sum_s \mu(s) \sum_a \nabla \pi(a|s) q_\pi(s,a) \tag{67}$$

QED.

当考虑折扣因子$\gamma \in (0,1)$时，我们可以从公式(55)推出如下形式:

$$\sum_{k=0}^{\infty} \sum_{s} Pr(s_0 \to s, k, \pi) \sum_{a} \nabla\pi(a|s)\gamma^k q_\pi(s,a) \tag{68}$$

为了方便对情节进行采样，这里我们交换了求和次序. 实际上公式(55)也应该如此，我猜测作者只是为了推导出漂亮简洁的公式形式。我们可以把公式(67)进一步展开:

$$\begin{aligned}
\sum_{k=0}^{\infty} \sum_{s} Pr(s_0 \to s, k, \pi) \sum_{a} \nabla\pi(a|s)\gamma^k q_\pi(s,a) &= \sum_{k=0}^{\infty} \sum_{s} Pr(s_0 \to s, k, \pi) \sum_{a} \nabla\pi(a|s)\gamma^k E_\pi[G_k|s,a] && (69) \\
&= \sum_{k=0}^{\infty} \sum_{s} Pr(s_0 \to s, k, \pi) \sum_{a} \pi(a|s)\{\frac{\nabla\pi(a|s)}{\pi(a|s)}\gamma^k E_\pi[G_k|s,a]\} && (70) \\
&= \sum_{k=0}^{\infty} E_\pi^k[\gamma^k G_k \frac{\nabla\pi(A_k|S_k)}{\pi(A_k|S_k)}] && (71) \\
&= \sum_{k=0}^{\infty} E_\pi^k[\gamma^k G_k \nabla ln\, \pi(A_k|S_k)] && (72)
\end{aligned}$$

在每一个时间步t，我们依照$Pr(s_0 \to S_t, k, \pi)$进行情节进行空间采样，得到一系列的情节用于梯度下降.

REINFORCE: Monte-Carlo Policy-Gradient Control (episodic) for π_*

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Algorithm parameter: step size $\alpha > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\gamma^t G \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

现在让我们从情节的联合分布出发，以更直观的角度理解如何推导$\nabla J(\theta)$.

$$
\begin{aligned}
J_\theta &\doteq \int_{s\in S} p(s) v_\pi(s)\, ds && (73)\\
&= \int_{s\in S} p(s) E_\pi[G_0|S_0=s]\, ds && (74)\\
&= \int_{s,s'\in S} p(s)\pi(a|s)p(s'|s,a)E_\pi[r+\gamma v_\pi(s')|S_0=s, A_0=a]\, ds\, ds'\, da && (75)\\
&= \int_\tau p(\tau;\theta)R(\tau)\, d\tau && (76)\\
& && (77)
\end{aligned}
$$

其中$\tau=\{S_t,A_t,R_t\}_{t=0,1,2..T}$, $p(\tau;\theta)=p(s_0)\prod_{t=1}^{T-1}p(s_{t+1}|a_t,s_t)\pi(a_t|s_t)$, $R(\tau)=\sum_{t=0}^{T-1}\gamma^t R_{t+1}$

$$
\begin{aligned}
\nabla J_\theta &= \int_\tau p(\tau;\theta)R(\tau)\, d\tau && (78)\\
&= \nabla_\theta \int_\tau p(\tau;\theta)R(\tau)\, d\tau && (79)\\
&= \int_\tau \nabla_\theta\, p(\tau;\theta)\cdot R(\tau)\, d\tau && (80)\\
&= \int_\tau p(\tau;\theta)\cdot \frac{\nabla_\theta\, p(\tau;\theta)}{p(\tau;\theta)}\cdot R(\tau)\, d\tau && (81)\\
&= E_\tau[\frac{\nabla_\theta\, p(\tau;\theta)}{p(\tau;\theta)}\cdot R(\tau)] && (82)\\
&= E_\tau[\nabla_\theta\, ln\, p(\tau;\theta)\cdot R(\tau)] && (83)
\end{aligned}
$$

接下来我们展开$\nabla_\theta\, ln\, p(\tau;\theta)$:

$$
\begin{aligned}
\nabla_\theta\, ln\, p(\tau;\theta) &= \nabla_\theta\{ln\, p(s_0)+\sum_{t=1}^{T-1}[ln\, p(s_{t+1}|a_t,s_t)+ln\, \pi(a_t|s_t)]\} && (84)\\
&= \nabla_\theta \sum_{t=1}^{T-1} ln\, \pi(a_t|s_t) && (85)\\
&= \sum_{t=1}^{T-1}\nabla_\theta ln\, \pi(a_t|s_t) && (86)
\end{aligned}
$$

结合公式(83),(85):

$$
\begin{aligned}
\nabla J_\theta &= E_\tau[\nabla_\theta\, ln\, p(\tau;\theta)\cdot R(\tau)] && (87)\\
&= E_\tau[\sum_{t=0}^{T-1}\gamma^t R_{t+1}\cdot \sum_{t=1}^{T-1}\nabla_\theta ln\, \pi(a_t|s_t)] && (88)\\
&= \sum_{t=1}^{T-1}\gamma^t E_\tau[R_{t+1}\cdot \sum_{t=1}^{T-1}\nabla_\theta ln\, \pi(a_t|s_t)] && (89)
\end{aligned}
$$

由于R_{t+1}与未来的状态$\{S_{t+1},..S_T\}$无关，我们有如下结论:

$$
\begin{aligned}
E_\tau[R_{t+1}\cdot\sum_{k=t+1}^{T-1}\nabla_\theta ln\,\pi(a_k|s_k)] &= E_\tau[R_{t+1}]\cdot E_\tau[\sum_{k=t+1}^{T-1}\nabla_\theta ln\,\pi(a_k|s_k)] &(90)\\
&= E_\tau[R_{t+1}]\cdot E_\tau[\nabla_\theta\,ln\,p(\tau_k;\theta)] &(91)\\
&= E_\tau[R_{t+1}]\cdot\int\nabla_\theta\,p(\tau_k;\theta)d\tau_k &(92)\\
&= E_\tau[R_{t+1}]\cdot\nabla_\theta\int p(\tau_k;\theta)d\tau_k &(93)\\
&= E_\tau[R_{t+1}]\cdot\nabla_\theta\mathbf{1} &(94)\\
&= 0 &(95)
\end{aligned}
$$

所以公式(89)可以重写为:

$$
\begin{aligned}
\nabla J_\theta &= \sum_{t=0}^{T-1}\gamma^t E_\tau[R_{t+1}\cdot\sum_{k=0}^{t}\nabla_\theta ln\,\pi(a_k|s_k)] &(96)\\
&= E_\tau[\sum_{t=0}^{T-1}\gamma^t R_{t+1}\cdot\sum_{k=0}^{t}\nabla_\theta ln\,\pi(a_k|s_k)] &(97)\\
&= E_\tau[\sum_{t=0}^{T-1}\gamma^t\nabla_\theta ln\,\pi(a_t|s_t)\sum_{k=t}^{T-1}\gamma^{k-t}R_{k+1}] &(98)\\
&= E_\tau[\sum_{t=0}^{T-1}\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t G_t] &(99)
\end{aligned}
$$

公式(99)看上去就比公式(72)优雅许多. 公式(72)不能求换求和次序, 而且只能逐步采样. 而公式(99)可以直接根据分布τ采样整个情节.两者的计算结果并无差别.

2.1 REINFORCE with Baseline

使用蒙特卡洛采样会导致梯度具有很大的方差，这使得训练收敛缓慢、学习不稳定. 这小节我们介绍一种降低方差的优化方法:REINFORCE with Baseline.

原始的蒙特卡洛采样基于公式(99):

$$\nabla J_{\theta_t} = \nabla_\theta ln\,\pi(a_t|s_t)\gamma^t G_t \tag{100}$$

我们可以在G_t原有基础上减去一个baseline: $b(s_t)$. 注意$b(s_t)$是根据S_t估计出来的一个量，它与被采样变量是独立的.

$$\nabla J_{\theta_t} = \nabla_\theta ln\,\pi(a_t|s_t)\gamma^t [G_t - b(s_t)] \tag{101}$$

这种方法的好处是不会影响公式(99)中每一个时间步t采样期望值:

$$E_\tau[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t (G_t - b(s_t)] = E_\tau[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t G_t] - E_\tau[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t b(s_t)] \tag{102}$$

$$\begin{aligned}
E_\tau[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t b(s_t)] &= \gamma^t E_\pi[\nabla_\theta ln\,\pi(a_t|s_t) b(s_t)] && (103)\\
&= \gamma^t \sum_{s,a} \pi(s|a) p(s_t = s) \nabla_\theta ln\,\pi(a|s) b(s) && (104)\\
&= \gamma^t \sum_{s,a} p(s_t = s) \nabla_\theta\, \pi(a|s) b(s) && (105)\\
&= \gamma^t \sum_{s} p(s_t = s) b(s) \sum_a \nabla_\theta\, \pi(a|s) && (106)\\
&= \gamma^t \sum_{s} p(s_t = s) b(s) \nabla_\theta \sum_a \pi(a|s) && (107)\\
&= \gamma^t \sum_{s} p(s_t = s) b(s) \nabla_\theta \mathbf{1} && (108)\\
&= 0 && (109)
\end{aligned}$$

故可得出结论: baseline不会影响每一步的采样期望值

$$E_\tau[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t (G_t - b(s_t)] = E_\tau[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t G_t] \tag{110}$$

我们现在来证明baseline可以有效的降低方差:

$$\begin{aligned}
&E_\tau[(\nabla_\theta ln\,\pi(a_t|s_t))^2 (G_t - b(s_t))^2] - E_\tau^2[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t (G_t - b(s_t)] && (111)\\
&= E_\tau[(\nabla_\theta ln\,\pi(a_t|s_t))^2 (G_t - b(s_t))^2] - E_\tau^2[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t G_t] && (112)\\
&\le E_\tau[(\nabla_\theta ln\,\pi(a_t|s_t))^2 G_t^2] - E_\tau^2[\nabla_\theta ln\,\pi(a_t|s_t)\gamma^t G_t] && (113)\\
&QED. && (114)
\end{aligned}$$

那么我们如何表示$b(s_t)$呢？一个较为合理的基准线是:

$$b(s_t) = \hat{v}(s_t, w) \tag{115}$$

其中$\hat{v}(s_t, w)$是价值函数$v_\pi(s_t)$的拟合结果, 使用价值函数估计作为基准线可以有效的缩小$[G_t - b(s_t)]^2$,从而更有效的降低方差.

那么我们应该如何拟合$\hat{v}(s_t, w)$呢? 我们可以直接采用$[G_t - \hat{v}(s_t, w)]^2$作为损失函数并对$w$进行梯度下降:

$$L = [G_t - \hat{v}(s_t, w)]^2 \tag{116}$$
$$\nabla_w L = -[G_t - \hat{v}(s_t, w)]\nabla_w \hat{v}(s_t, w) \tag{117}$$
$$w \leftarrow w + \alpha^w [G_t - \hat{v}(s_t, w)]\nabla_w \hat{v}(s_t, w) \tag{118}$$

我们使用如下公式更新θ:

$$\theta \leftarrow \theta + \alpha^\theta \gamma^t [G_t - \hat{v}(s_t, w)]\nabla_\theta ln\, \pi(a_t|s_t) \tag{119}$$

REINFORCE with Baseline (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Algorithm parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g., to $\mathbf{0}$)

Loop forever (for each episode):
  Generate an episode $S_0, A_0, R_1, \ldots, S_{T-1}, A_{T-1}, R_T$, following $\pi(\cdot|\cdot, \boldsymbol{\theta})$
  Loop for each step of the episode $t = 0, 1, \ldots, T-1$:
    $G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$ (G_t)
    $\delta \leftarrow G - \hat{v}(S_t,\mathbf{w})$
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S_t,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} \gamma^t \delta \nabla \ln \pi(A_t|S_t, \boldsymbol{\theta})$

2.2 Actor-Critic

前面提到的方法都有一个共同的缺点:我们需要等到整个情节结束才能开始这一轮的参数学习. 我们希望方法可以具有增量学习的特点——随着情节的推进持续学习. 对此我们可以借鉴TD相关方法的思路:

$$\delta = R_{t+1} + \gamma \hat{v}(s_{t+1}, w) - \gamma \hat{v}(s_t, w) \tag{120}$$

此时, 当采样到s_t时, 我们只需要进行一步动作选择和状态观察就可以对参数进行更新. 并且将G_t替换$R_{t+1} + \gamma \hat{v}(s_{t+1}, w)$进一步降低了方差. 这个方法被称为Actor-Critic, 演员$\pi(\cdot|S_t, \theta)$进行一步动作选择, 评论家δ对动作选择带来的影响进行评估.

One-step Actor–Critic (episodic), for estimating $\pi_{\boldsymbol{\theta}} \approx \pi_*$

Input: a differentiable policy parameterization $\pi(a|s, \boldsymbol{\theta})$
Input: a differentiable state-value function parameterization $\hat{v}(s,\mathbf{w})$
Parameters: step sizes $\alpha^{\boldsymbol{\theta}} > 0$, $\alpha^{\mathbf{w}} > 0$
Initialize policy parameter $\boldsymbol{\theta} \in \mathbb{R}^{d'}$ and state-value weights $\mathbf{w} \in \mathbb{R}^d$ (e.g., to $\mathbf{0}$)
Loop forever (for each episode):
  Initialize S (first state of episode)
  $I \leftarrow 1$
  Loop while S is not terminal (for each time step):
    $A \sim \pi(\cdot|S, \boldsymbol{\theta})$
    Take action A, observe S', R
    $\delta \leftarrow R + \gamma \hat{v}(S',\mathbf{w}) - \hat{v}(S,\mathbf{w})$ (if S' is terminal, then $\hat{v}(S',\mathbf{w}) \doteq 0$)
    $\mathbf{w} \leftarrow \mathbf{w} + \alpha^{\mathbf{w}} \delta \nabla \hat{v}(S,\mathbf{w})$
    $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha^{\boldsymbol{\theta}} I \delta \nabla \ln \pi(A|S, \boldsymbol{\theta})$
    $I \leftarrow \gamma I$
    $S \leftarrow S'$