强 化 学 习

目 录

图 片

表 格

算 法

要 点

1 导论

特征 智能体与环境交互（采样），在不断尝试中学习策略，使收益最大化。

- 试错探索：不会获知应采取的行动，通过尝试获得。
- 延迟收益：一个动作的收益可能无法短期体现，而是长期浮现。
- 环境不确定性：当前动作不但会影响当前收益，还会影响后续环境，进而影响后续收益。
- 影响未知性：无法预测动作的影响，需要与环境频繁交互。
- 试探（开拓动作空间）与开发/贪心（根据经验获得收益）折中。

优化方法对比

- 凸优化：状态空间较小。可线性规划。
- 最优控制：已知模型，解析回报函数。可动态规划，解HJB方程。
- 进化算法：控制策略简单。如遗传算法。
- 机器学习
  - 有监督学习：有标签数据，注重推断与泛化能力。
  - 无监督学习：无标签数据，寻找数据隐含结构。
- 强化学习：交互数据，优化策略以优化收益。

分类

1. 模型依赖性
   - 有模型：学习模型，规划策略。
   - 无模型：直接试错策略。
2. 策略更新方法
   - 值函数：求解值函数重构策略。
   - 直接策略搜索：策略梯度等方法，搜索策略空间。

返回目录

   - Actor-Critic方法：类似策略梯度，同时逼近值函数和策略。

3. 回报函数是否已知
   - 正向：从回报到策略。
   - 逆向：从专家示例到回报。

4. 任务体量：分层强化学习、元强化学习、多智能体强化学习、迁移学习等

5. 框架
   - 间接强化学习：充分利用有限经验，获得更好策略，减少与环境的交互。
   - 直接强化学习：不受模型设计偏差影响。

发展

值函数→直接策略搜索（策略梯度等）→深度强化学习。详见12.1.1。

发展方向：与深度学习结合，与专业知识结合，理论分析型增强，与认知科学结合，体量增大，与贝叶斯结合。

2 马尔可夫决策过程与贝尔曼方程

2.1 马尔可夫决策过程（Markov decision process，MDP）

2.1.1 要素 [1]

- 状态（state，S）：强化学习依赖的概念。
- 动作（action，A）：智能体做出的选择。
- 奖励/收益（reward，R）：短期学习目标，环境给予智能体的信号。
- 策略（policy，π）：特定状态下动作空间上的分布$\pi(a|s) = p[A_t = a|S_t = s]$。
- 回报（return，G）：长期收益累计，可能含有折扣，需综合评估。
- 折扣因子（$\gamma \in [0, 1]$）。

返回目录

- 值函数（value function，V）：对s预估的期望回报。
- 行为/动作值函数（Q）：对(s,a)预估的期望回报。
- 环境模型（P）：模拟环境的反应，可以是确定性转移，也可以是随机性转移。
- 大写字母表示空间，小写字母表示个体，上标*表示最优。

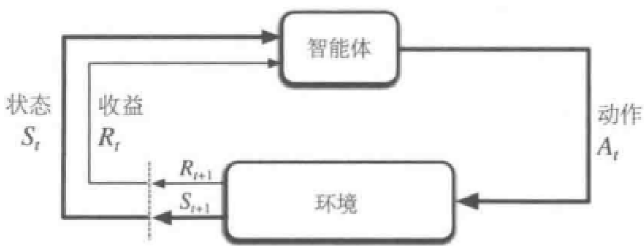


图 1: 马尔可夫决策过程

2.1.2 状态、动作与收益

序贯交互轨迹（TRAJECTORY） $\tau = s_0, a_0, r_1, s_1, a_1, r_2, \ldots$

随机变量s', r服从离散概率分布$p(s', r|s, a) \doteq \Pr\{S_t = s', R_t = r|S_{t-1} = s, A_{t-1} = a\}$

$$\sum_{s' \in S} \sum_{r \in R} p(s', r|s, a) = 1$$

。

马尔可夫性 [2] 即“无记忆性”，未来状态仅依赖当前状态，而独立于过去状态。

$$P(S_{t+1}|S_t, S_{t-1}, \ldots, S_0) = P(S_{t+1}|S_t)$$

状态转移与期望收益

由s和a转移到s′的概率，包括s′下各可能收益情况：

$$p(s'|s, a) \doteq \Pr\{S_t = s'|S_{t-1} = s, A_{t-1} = a\} = \sum_{r \in R} p(s', r|s, a)$$

若不指定a，由s转移到s′的概率为：

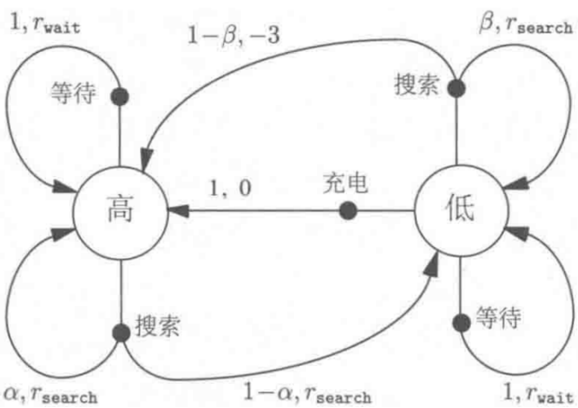
$$p(s'|s) = \sum_{a \in A} [p(s'|s, a)p(a|s)]$$

返回目录

有无s'的两种期望收益：

$$r(s,a) \doteq E[R_t|S_{t-1}=s, A_{t-1}=a] = \sum_{r\in R} r \sum_{s'\in S} p(s',r|s,a)$$

$$r(s,a,s') \doteq E[R_t|S_{t-1}=s, A_{t-1}=a, S_t=s'] = \sum_{r\in R} r \frac{p(s',r|s,a)}{p(s'|s,a)}$$



(a) 状态转移图（节点不能重复）

s	a	s'	$p(s'\|s,a)$	$r(s,a,s')$
high	search	high	α	r_{search}
high	search	low	$1-\alpha$	r_{search}
low	search	high	$1-\beta$	-3
low	search	low	β	r_{search}
high	wait	high	1	r_{wait}
high	wait	low	0	r_{wait}
low	wait	high	0	r_{wait}
low	wait	low	1	r_{wait}
low	recharge	high	1	0
low	recharge	low	0	0

(b) 状态转移表

图 2: 回收机器人状态转移

2.1.3 策略

贪婪策略 $\pi(a|s) = \text{argmax}_a q(a)$。

探索-利用平衡策略

- ϵ-greedy策略 [3] ：靠近贪心策略，但所有动作概率不为零。实际使用时需注意多最优情况。

$$a = \begin{cases} \text{argmax}_a q(a) & ,p = 1-\epsilon \\ \text{random}(a) & ,p = \epsilon \end{cases} \Rightarrow \pi(a|s) = \begin{cases} 1-\epsilon+\frac{\epsilon}{|A|} & ,a = \text{argmax}_a q(a) \\ \frac{\epsilon}{|A|} & ,\text{otherwise} \end{cases}$$

- UCB（Upper Confidence Bound）策略：可以自适应平衡探索与利用。

$$\pi(a|s) = Q(a) + c\sqrt{\frac{\ln N}{n(a)}}$$

其中，c控制探索强度，N是当前轮数，$n(a)$是a被选次数。

返回目录

- 玻尔兹曼分布（Boltzmann）：可以动态调整探索强度。

$$\pi(a|s)=\frac{e^{Q(a)/\tau}}{\sum_{a'}e^{Q(a)/\tau}}$$

其中τ是温度参数，控制随机程度，趋于0时贪心，趋于∞时随机。

- 高斯策略：

$$\pi=\mu+\epsilon,\epsilon\sim N(0,\sigma^2)$$

增量式更新 [4] 将轮次更新转化为递推关系，减少空间复杂度，如运行均值：

$$Q_{n+1}=\frac{1}{n}\sum_{i=1}^{n}R_i=Q_n+\frac{1}{n}(R_n-Q_n)$$

2.1.4 回报与折扣 [5]

- 幕（episode）：一次交互序列，幕间没有联系。
- 终止时刻T：划分非终结状态集S和所有状态集S^+。
- 分幕式任务（episodic tasks）：有终止状态，可分幕。

$$G_t=R_{t+1}+R_{t+2}+\cdots+R_T=\sum_{k=t+1}^{T}R_k$$

- 持续性任务（continuing tasks）：没有终止状态，持续进行，不能自然分幕。

$$G_t=R_{t+1}+\gamma R_{t+2}+\gamma^2R_{t+3}+\cdots=\sum_{k=0}^{\infty}\gamma^kR_{t+k+1}\leqslant\frac{1}{1-\gamma}maxR_t$$

其中γ越大代表长期收益越重要。

- 统一表示：有限项终止后，状态持续转移回自己，相当于无限项。

$$G_t\doteq\sum_{k=t+1}^{T}\gamma^{k-t-1}R_k$$

返回目录

2.1.5 值函数 [6]

值函数

$$
\begin{aligned}
v_\pi(s) &\doteq E_\pi[G_t|S_t=s] = E_\pi[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t=s], s\in S \\
&= E_\pi[\underbrace{R_{t+1}}_{\text{即时奖励}} + \gamma \underbrace{G_{t+1}}_{\text{未来奖励}} |S_t=s](\text{后继递推关系}) \\
&= \sum_{a\in A}\pi(a|s)[\sum_{r\in R}p(r|s,a)r+\gamma\sum_{s'\in S}p(s'|s,a)v_\pi(s')](\text{分项全概率展开}) \\
&= \underbrace{\sum_{a\in A}\pi(a|s)\overbrace{\sum_{s'\in S,r\in R}p(s',r|s,a)[r+\gamma v_\pi(s')]}^{q_\pi(s,a)}}_{\text{贝尔曼方程}}(\text{值函数递推关系})
\end{aligned}
$$

行为值函数

$$
\begin{aligned}
q_\pi(s,a) &\doteq E_\pi[G_t|S_t=s,A_t=a] = E_\pi[\sum_{k=0}^{\infty}\gamma^k R_{t+k+1}|S_t=s,A_t=a], s\in S \\
&= \sum_{r\in R}p(r|s,a)r+\gamma\sum_{s'\in S}P(s'|s,a)\underbrace{\sum_{a'\in A}\pi(a'|s')q_\pi(s'|a')}_{v_\pi(s')}(\text{状态值函数递推关系})
\end{aligned}
$$

回溯算法　s'的价值信息回传给s。

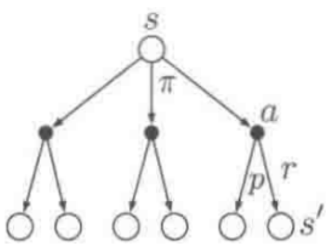


图 3: DP回溯图（节点可以重复）

最优值函数与最优策略

- $\forall s\in S, q_\pi(s,\pi'(s)) = v_{\pi'}(s) \geqslant v_\pi(s)$，则称$\pi'$优于或等于$\pi$。
- $v_\pi(s)$定义了π的偏序关系，π^*存在且可能不唯一，它们共享：

$$v^*(s) \doteq \max_\pi v_\pi(s) \quad q^*(s,a) \doteq \max_\pi q_\pi(s,a)$$

返回目录

2.1.6 构建要点

- 确定s, a, r（不含先验知识，不为达到子目标而舍弃最终目标）。
- 奖励与惩罚是相对的，可以全奖励或全惩罚。
- 同一问题可能有多层次MDP。
- 利用先验知识，人为排除愚蠢动作。

2.2 贝尔曼方程 [7]

2.2.1 贝尔曼方程

$$\begin{aligned}
\underline{v_\pi(s)} &= \sum_{a\in A}\pi(a|s)\sum_{s'\in S, r\in R}P(s', r|s, a)[r+\gamma \underline{v_\pi(s')}] \\
&= \underbrace{\sum_{a\in A}\pi(a|s)\sum_{r\in R}P(r|s, a)}_{r_\pi(s)}+\gamma\sum_{s'\in S}\underbrace{[\sum_{a\in A}\pi(a|s)P(s'|s, a)]}_{p_\pi(s'|s)}\underline{v_\pi(s')}
\end{aligned}$$

2.2.2 贝尔曼最优方程

方程组中方程数为$|S|$，如P已知，并具有马尔可夫性，则可求解。但一般难以满足，且计算资源有限，求近似解。

形式

- $s\to a^*$：

$$\begin{aligned}
v^*(s) &= \max_{a\in A} q_{\pi^*}(s, a)(\text{凸组合最优}) \\
&= \max_{\pi}\sum_{s\in S}\pi(a|s)q(s, a)(\text{元素}) \\
&= \max_{\pi}(r_\pi+\gamma P_\pi v)(\text{矩阵})
\end{aligned}$$

- $(s,a) \to (s,a)^*_{next}$：

$$q^*(s,a) = E[R_{t+1} + \gamma \max_{a'} q^*(S_{t+1}, a') | S_t = s, A_t = a]$$
$$= \sum_{s' \in S, r \in R} p(s', r | s, a)[r + \gamma \max_{a'} q^*(s', a')]$$

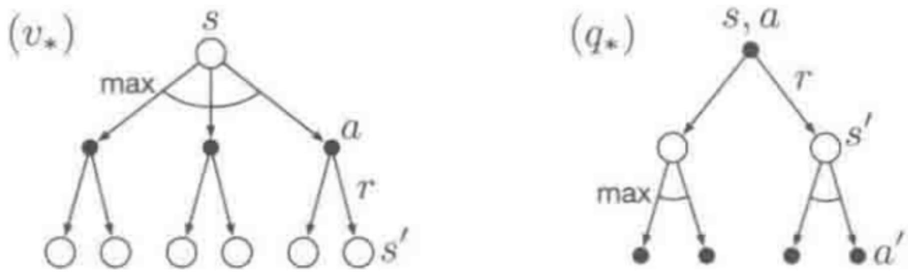


图 4: DP回溯图的两种形式（最优）

求解　伸缩映射性，见12.1.2。

贪婪最优策略　π^*中，$v(s) = E[r(a^*|s)]$，可使用贪心策略求取（证明：凸组合最大值为最大一项）。

3 动态规划（DYNAMIC PROGRAMMING，DP）：期望更新

使用值函数结构化组织最优策略搜索，将贝尔曼方程转化成近似逼近理想值函数的递归更新公式，即将多阶段决策问题转化为多个单阶段决策问题。

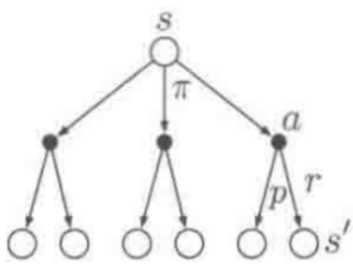


图 5: DP回溯图：显示一步的所有转移

3.1 策略迭代 [8]

$$\pi_0 \xrightarrow{PE} v_{\pi_0} \xrightarrow{PI} \pi_1 \xrightarrow{PE} v_{\pi_1} \xrightarrow{PI} \pi_2 \xrightarrow{PE} v_{\pi_2} \xrightarrow{PI} \dots$$

反复进行PE和PI，得到改进的v_π估计和π，最后收敛到最优。

- 策略评估（PE）：
  - 直接求解：$v_{\pi_k} = (I - \gamma P_{\pi_k})^{-1} r_{\pi_k}$。
  - 迭代求解：

$$\underbrace{v_{\pi_k}^{(j+1)} = r_{\pi_k} + \gamma P_{\pi_k} v_{\pi_k}^{(j)} = \sum_{a \in A} \pi(a|s)[r(a|s) + \gamma \sum_{s' \in S} P(s'|s,a) v_\pi(s')], j = 0, 1, 2, \dots}_{\text{期望更新：基于后继可能状态的期望值}}$$

  - 截断策略评估：不需要完全收敛。
- 策略改进（PI） [9] ：
  - 理论：$v_\pi(s) \leqslant q_\pi[s, \pi'(s)]$则$\pi'$不次于$\pi$

$$\begin{aligned} v_\pi(s) &\leqslant q_\pi[s, \pi'(s)] \\ &= E_{\pi'}[R_{t+1} + \gamma v_\pi(S_{t+1}) | S_t = s, A_t = \pi'(s)] \\ &\leqslant E_{\pi'}\{R_{t+1} + \gamma q_\pi[S_{t+1}, \pi'(S_{t+1})] | s_t = s\} \\ &\leqslant E_{\pi'}[R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots | S_t = s] \\ &= v_{\pi'}(s) \end{aligned}$$

  - 贪心策略：$\pi_{k+1} = \mathrm{argmax}_\pi (r_\pi + \gamma P_\pi v_{\pi_k})$

算法 1：策略迭代

参数：估计精度阈值$\theta > 0$
初始化：$\forall s \in S$，任意初始化$v(s) \in R, \pi(s)$
循环
    循环 ▷ 策略评估
        $\Delta \leftarrow 0$
        对于 $\forall s \in S$ 执行
            $v_{\pi_k}^{(j+1)}(s) \leftarrow \sum_{a \in A} \pi_k(a|s)[\sum_{r \in R} p(r|s,a) r + \gamma \sum_{s' \in S} p(s'|s,a) v_{\pi_k}^{(j)}(s')]$
            $\Delta \leftarrow \max(\Delta, |v - v_{\pi_k}^{(j+1)}(s)|)$

返回目录

算法 1：策略迭代

直到 $\Delta < \theta$

策略稳定$\leftarrow$true ▷ 策略改进

对于 $\forall s \in S$ 执行

  $a_{old} \leftarrow \pi(s)$

  对于 $\forall a \in A(s)$ 执行

    $q_{\pi_k}(s,a) \leftarrow \sum_{r\in R} p(r|s,a)r + \gamma \sum_{s'\in S} p(s'|s,a)v_{\pi_k}(s')$

  $a_k^*(s) \leftarrow \mathrm{argmax}_a q_{\pi_k}(s,a)$，并更新$\pi(s)$

  如果 $a_{old} \neq a_k^*(s)$ 那么

    策略稳定$\leftarrow$false

直到 策略稳定

3.2 值迭代 [10]

$$u_0 \xrightarrow{PU} \pi_1' \xrightarrow{VU} u_1 \xrightarrow{PU} \pi_2' \xrightarrow{VU} u_2 \xrightarrow{PU} \dots$$

结合极端PE和PI，只进行一次PE遍历，对每个状态更新一次。

$$v_{k+1} = \max_{\pi} r_\pi + \gamma P_\pi v_k, k = 1,2,3,\dots$$

- 策略更新（PU）：$\pi_{k+1} = \mathrm{argmax}_\pi (r_\pi + \gamma P_\pi v_k)$，贪婪选取。
- 价值更新（VU）：$v_{k+1} = r_{\pi_{k+1}} + \gamma P_{\pi_{k+1}} v_k = \max_{a\in A} q_k$。

算法 2：值迭代

参数：估计精度阈值$\theta > 0$

初始化：$\forall s \in S^+$，任意初始化$v(s), v(终止) = 0$

循环

  $\Delta \leftarrow 0$

  对于 $\forall s \in S$ 执行

    对于 $\forall a \in A(s)$ 执行

      $q_k(s,a) \leftarrow \sum_{r\in R} p(r|s,a)r + \gamma \sum_{s'\in S} p(s'|s,a)v_k(s')$

      $a_k^*(s) \leftarrow \mathrm{argmax}_a q_k(s,a)$ ▷ 贪婪策略

算法 2：值迭代

若$a = a_k^*$且$\pi_{k+1}(a|s) = 0$，则令$\pi_{k+1}(a|s) = 1$ ▷ 策略更新

$v_{k+1}(s) \leftarrow \max_a q_k(s,a)$ ▷ 价值更新

$\Delta \leftarrow \max(\Delta, |v_{k+1} - v_k|)$ ▷ 一轮中反复更新精度

直到 $\Delta < \theta$

例题 11 出界和障碍−1分，目标1分，折扣率λ。

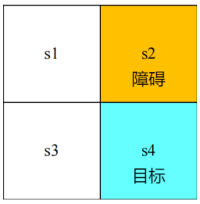


图 6: 值函数计算例题

$k=0$	a_1	a_2	a_3	a_4	a_5
s_1	-1	-1	0	-1	0
s_2	-1	-1	1	0	-1
s_3	0	1	-1	-1	0
s_4	1	1	1	0	1

$k=1$	a_1	a_2	a_3	a_4	a_5
s_1	$-1+\gamma 0$	$-1+\gamma 1$	$0+\gamma 1$	$-1+\gamma 0$	$0+\gamma 0$
s_2	$-1+\gamma 1$	$-1+\gamma 1$	$1+\gamma 1$	$0+\gamma 0$	$-1+\gamma 1$
s_3	$0+\gamma 0$	$1+\gamma 1$	$-1+\gamma 1$	$-1+\gamma 1$	$0+\gamma 1$
s_4	$1+\gamma 1$	$1+\gamma 1$	$1+\gamma 1$	$0+\gamma 1$	$1+\gamma 1$

表 1: 值函数计算例题

3.3 对比与补充

	策略迭代	值迭代
维护内容	值函数+策略	值函数
收敛速度	较快	较慢
收敛性	依赖初始策略质量，可能陷入局部最优	保证全局最优
适用策略空间	简单	复杂
计算成本	较低	较高（迭代遍历所有动作）

表 2: DP对比

补充见12.2.1。

返回目录

4 蒙特卡洛（MONTE CARLO，MC）：采样更新

针对分幕式任务，不需要P，通过多幕采样数据获得经验代替值函数解决问题。

4.1 概念 [12]

核心需求 由于P的缺失，$v(s)$是不够的，需要评估$q(s,a)$。

估计 $q(s,a)$

- 访问（visit）：给定的一幕中，指定状态的一次出现。
- 首次访问（first visit）：$\hat{q}(s,a)=\frac{G_{11}(s,a)+G_{21}(s,a)+\dots}{N(s,a)}$。
- 每次访问（every visit）：$\hat{q}(s,a)=\frac{G_{11}(s,a)+G_{12}(s,a)+\cdots+G_{21}(s,a)+\dots}{N(s,a)}$。

$N(s)$是s的访问次数，$N(s)\to\infty, \hat{q}(s,a)\to q_\pi(s,a)$。

图 7: MC回溯图：显示一幕所有采样到的转移

幕长 靠近目标的状态比远离目标的状态更早具有非零值，幕长应足够长，无需无限长。

优势

- 不需要P，可从实际经历和模拟经历中学习。
- 对每个状态的估计是独立的，可聚焦于状态子集，无需考虑其他状态，效率高。
- 无马尔可夫性时性能损失较小。

恒温策略： $\forall(s,a),\pi(a|s)>0$

1. 试探性出发（ES）：为采样部分无法正常获得的(s,a)，可设定所有(s,a)都有概率作为起始。满足充分探索的理论要求，但实际中很难实现。
2. ϵ-greedy策略。

返回目录

4.2 on-policy（同轨）[13]

采样并改进相同策略。

算法 3：MC-On-policy（first visit）

参数：$\epsilon > 0$
初始化：$\forall s \in S, a \in A(s)$，任意初始化$q(s,a) \in R$，初始化$Returns(s,a)$为空列表，$\epsilon$-greedy初始化$\pi$
循环
  根据π生成一幕序列$S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$
  $G \leftarrow 0$
  对于 $t = T-1, T-2, \ldots, 0$ 执行
    $G \leftarrow \gamma G + R_{t+1}$
    如果 S_t在此幕中首次出现 那么
      将G加入$Returns(S_t, A_t)$
      $q(S_t, A_t) \leftarrow average[Returns(S_t, A_t)]$
      $a^* \leftarrow argmax_a q(S_t, a)$
      ϵ-greedy策略选取$\pi(a|S_t)$

4.3 off-policy（离轨）[14]

采样与改进不同策略，前者称为行为策略（Behavior Policy）b（保证对所有可能动作的采样），后者称为目标策略（Target Policy）π。

重要度采样（IMPORTANCE SAMPLING）[15] off-policy的数学基础。

计算G时，对轨迹在π和b中出现的相对概率进行加权：

$$\rho_{t:T-1} = \Pi_{k=t}^{T-1} \frac{\pi(A_k|S_k)}{b(A_k|S_k)} (\text{约去相同的转移概率})$$

- 普通重要度采样：$v(s) \doteq \frac{\sum_{t \in \tau(s)} \rho_{t:T(t)-1} G_t}{|\tau(s)|}$，无偏但无界。
- 加权重要度采样：$v(s) \doteq \frac{\sum_{t \in \tau(s)} \rho_{t:T(t)-1} G_t}{\sum_{t \in \tau(s)} \rho_{t:T(t)-1}}$，有偏但偏差值渐近收敛。

减小方差的方法见12.2.2。

增量式更新

$$v_{n+1} \doteq v_n + \frac{W_n}{C_n}[G_n - v_n](v_n\text{和}G_n\text{线性组合})$$
$$C_{n+1} \doteq C_n + W_{n+1}$$

其中，W_i是随机权重，C_i是其累加和。

算法 4：MC-Off-policy（every visit）

初始化：$\forall s \in S, a \in A(s)$，任意初始化$q(s,a) \in R, C(s,a) = 0$，初始化$\pi(s) = \mathrm{argmax}_a q(s,a)$ ▷ 目标策略为贪婪策略
循环
    根据b生成一幕序列$S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T$ ▷ 行为策略为ϵ-greedy策略
    $G \leftarrow 0, W \leftarrow 1$
    对于 $t = T-1, T-2, \ldots, 0$ 执行
        $G \leftarrow \gamma G + R_{t+1}$
        $C(S_t, A_t) \leftarrow C(S_t, A_t) + W$
        $q(S_t, A_t) \leftarrow q(S_t, A_t) + \frac{W}{C(S_t,A_t)}[G - q(S_t, A_t)]$ ▷ 增量式更新
        $\pi(S_t) \leftarrow \mathrm{argmax}_a q(S_t, a)$
        如果 $A_t \neq \pi(S_t)$ 那么
            break ▷ 如果不是最优动作则退出内层循环
        $W \leftarrow \frac{W}{b(A_t|S_t)}$ ▷ 更新重要度采样权重

潜在问题：贪心行为普遍时，只会从幕尾学习；贪心行为不普遍时，学习速度较慢。

4.4 对比

策略类型	稳定性	收敛性
on-policy	较稳定	需要更多样本（更新需要新的数据）
off-policy	不太稳定（使用行为策略）	更快找到优质解

表 3: MC对比

返回目录

5 时序差分（TEMPORAL DIFFERENCE，TD）：采样更新

TD可直接从与环境的互动中获取信息，不需要P，同时运用自举思想，可基于已得到的其他状态估计来更新当前$v(s)$，相当于结合了DP和MC的优点。

5.1 TD(0) [16]

更新公式为：

$$v_{t+1}(s_t) = v_t(s_t) + \alpha_t(s_t)[\overbrace{\underbrace{r_{t+1} + \gamma v_t(s_{t+1})}_{\text{TD目标（target）}} - v_t(s_t)}^{\text{新息：TD误差（error）}\delta_t}]$$

- G_t是$v_\pi(s_t)$的无偏估计；$r_{t+1} + \gamma v_\pi(s_{t+1})$是无偏估计，而$r_{t+1} + \gamma v(s_{t+1})$是有偏估计。
- MC误差可写成TD误差之和$G_t - v(s_t) = \sum_{k=t}^{T-1} \gamma^{k-t} \delta_k$，其在步长较小时成立。

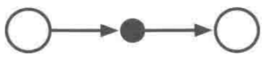


图 8: TD回溯图

优势

- 不需要P, R。
- 更新快：MC须等到幕尾确定增量，更新G_t；而TD只需等到下一时刻，更新TD目标。
- 方差小：MC更新依赖很多随机动作，TD更新仅依赖一个随机动作，因此TD目标的方差比G_t的方差要小很多。
- 随机任务中，TD(0)收敛速度要比常量αMC快。前者的最优性与预测回报更相关，找出的是完全符合马尔可夫过程的最大似然估计参数，收敛到确定性等价估计；而后者只在有限方面最优，找出的是最小化训练集均方误差的估计。
- 自举性：TD需要初始猜测值，MC没有进行初始猜测。
- 只评估当前动作，与后续动作无关。

返回目录

算法 5：TD(0)

输入：待评估策略π
参数：步长$\alpha \in (0, 1]$
初始化：$\forall s \in S^+$，任意初始化$v(s), v(\text{终止}) = 0$
对于 每一幕 执行
　　初始化s
　　当 s不是终止状态 执行
　　　　由$\pi(s)$获得a并执行，观察r, s'
　　　　$v(s) \leftarrow v(s) + \alpha[r + \gamma v(s') - v(s)]$
　　　　$s \leftarrow s'$

5.2 Sarsa（on-policy-TD）[17]

Sarsa（State-Action-Reward-State-Action），更新单元为$(s_t, a_t, r_t, s_{t+1}, a_{t+1})$，是TD算法的$q(s, a)$版本：

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) + \alpha_t(s_t, a_t)[r_{t+1} + \gamma q_t(s_{t+1}, a_{t+1}) - q_t(s_t, a_t)]$$

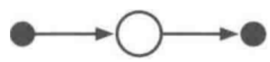


图 9: Sarsa回溯图

算法 6：Sarsa（on-policy-TD）

参数：步长$\alpha \in (0, 1], \epsilon > 0$
初始化：$\forall s \in S^+$，任意初始化$q(s, a), q(\text{终止}, \cdot) = 0$
对于 每一幕 执行
　　初始化s
　　使用从q得到的ϵ-greedy策略，在s选择a
　　当 s不是终止状态 执行
　　　　执行a，观察r, s'
　　　　使用从q得到的ϵ-greedy策略，在s'处选择a'

返回目录

算法 6：Sarsa（on-policy-TD）

$q(s,a) \leftarrow q(s,a) + \alpha[R + \gamma q(s',a') - q(s,a)]$

$s \leftarrow s', a \leftarrow a'$

期望SARSA

$$q_{t+1}(s_t,a_t) = q_t(s_t,a_t) + \alpha_t(s_t,a_t)\{r_{t+1} + \gamma\underbrace{[\sum_a \pi(a|s_{t+1})q_t(s_{t+1},a)]}_{v_t(s_{t+1})} - q_t(s_t,a_t)\}$$

- 相较Sarsa，期望Sarsa虽然计算复杂，但消除了随机选择带来的方差。
- α的选择会影响长期稳态性等指标。
- 生成策略可以基于相同或不同策略，即离轨或在轨是可变的，因此Q-learning可视为期望Sarsa的特例。

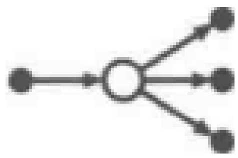


图 10: 期望Sarsa回溯图

5.3 Q-learning（off-policy-TD）[18]

Q-learning旨在求解行为值贝尔曼最优方程，直接逼近$q^*(s,a)$，更新单元为(s_t,a_t,r_t,s_{t+1})。

$$q_{t+1}(s_t,a_t) = q_t(s_t,a_t) + \alpha_t(s_t,a_t)[r_{t+1} + \gamma \max_a q_t(s_{t+1},a) - q_t(s_t,a_t)]$$

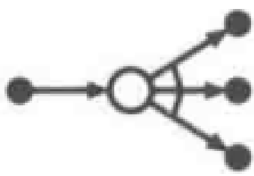


图 11: Q-learning回溯图

算法 7：Q-learning（off-policy-TD）

参数：步长$\alpha \in (0,1]$，探索率$\epsilon > 0$

初始化：$\forall s \in S^+, a \in A(s)$，任意初始化$q(s,a), q(终止,\cdot) = 0$

对于 每一幕 执行

算法 7：Q-learning（off-policy-TD）

初始化s
当 s不是终止状态 执行
　　使用从q得到的ϵ-greedy策略，在s选择a并执行，观察r, s'
　　$q(s,a) \leftarrow q(s,a) + \alpha[r + \gamma \max_a q(s',a) - q(s,a)]$
　　$s \leftarrow s'$

双Q-learning　双学习：划分样本，学习两个独立的估计$q_1(a), q_2(a)$，确定$a^* = \text{argmax}_a q_1(a)$，再计算$q_2(a*) = q_2(\text{argmax}_a q_1(a))$，后者是无偏的（可以交换再来一次）。需要双倍内存，但是计算量不会增大。

$$q_{1_{t+1}}(s_t,a_t) = q_{1_t}(s_t,a_t) + \alpha_t(s_t,a_t)\{r_{t+1} + \gamma q_{2_t}[s_{t+1}, \text{argmax}_a q_{1_t}(s_{t+1},a)] - q_{1_t}(s_t,a_t)\}$$

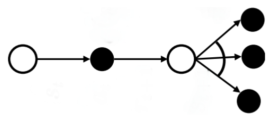


图 12: 双Q-learning回溯图

算法 8：双Q-learning

参数：步长$\alpha \in (0,1]$，探索率$\epsilon > 0$
初始化：$\forall s \in S^+, a \in A(s)$，任意初始化$q_1(s,a), q_2(s,a), q_1(终止,\cdot) = q_2(终止,\cdot) = 0$
对于 每一幕 执行
　　初始化s
　　当 s不是终止状态 执行
　　　　基于$q_1 + q_2$，使用ϵ-greedy策略在s选择a并执行，观察r, s'
　　　　如果 以0.5的概率 那么
　　　　　　$q_1(s,a) \leftarrow q_1(s,a) + \alpha[r + \gamma q_2(s', \text{argmax}_a Q_1(s',a)) - Q_1(s,a)]$
　　　　否则
　　　　　　$q_2(s,a) \leftarrow q_2(s,a) + \alpha[r + \gamma q_1(s', \text{argmax}_a Q_2(s',a)) - Q_2(s,a)]$
　　　　$s \leftarrow s'$

5.4 n-TD [19]

n-TD作为MC和TD的一般推广，在两种极端方法间找到了性能更好的平衡点。n-TD在n步后进行更新，截断得到n步回报。

$$G_{t:t+n} \doteq r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^{n} v_{t+n-1}(s_{t+n})$$

其中$v_{t+n}(s_t) \doteq v_{t+n-1}(s_t) + \alpha[G_{t:t+n} - v_{t+n-1}(s_t)]$。

算法 9：n-TD

输入：待评估策略π
参数：步长$\alpha \in (0,1], n \in N_+$
初始化：$\forall s \in S$，任意初始化$v(s)$
对于 每一幕 执行
  初始化s_0为非终止状态
  $T \leftarrow \infty$
  对于 $t = 0, 1, 2, \ldots$ 执行
    如果 $t < T$ 那么
      根据$\pi(\cdot|s_t)$获得a_t并执行，观察r_{t+1}, s_{t+1}
      如果 S_{t+1}是终止状态 那么
        $T \leftarrow t + 1$
    $\tau \leftarrow t - n + 1$ ▷ τ是正在更新的状态的时间
    如果 $\tau \geqslant 0$ 那么
      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
      如果 $\tau + n < T$ 那么
        $G \leftarrow G + \gamma^{n} V(S_{\tau+n})$
      $V(S_\tau) \leftarrow V(S_\tau) + \alpha[G - V(S_\tau)]$
    如果 $\tau = T - 1$ 那么
      break

5.5 n-Sarsa [20]

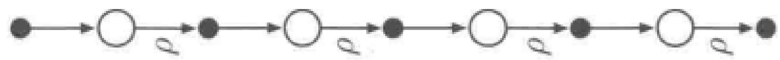
n-Sarsa统一了Sarsa和MC，其节点转移全部基于采样得到的单独路径：

$$q_{t+n}(s_t, a_t) \doteq q_{t+n-1}(s_t, a_t) + \alpha[G_{t:t+n} - q_{t+n-1}(s_t, a_t)]$$

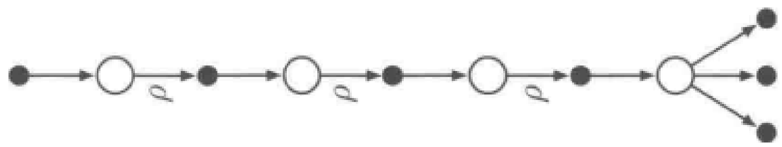
返回目录

n-期望Sarsa只对最后一个状态到动作的转移展开：

$$G_{t:t+n} \doteq r_{t+1} + \gamma r_{t+2} + \cdots + \gamma^{n-1} r_{t+n} + \gamma^n q_\pi(s_{t+n}, a_{t+n})$$



(a) n-Sarsa



(b) n-期望Sarsa

图 13: n-Sarsa回溯图

算法 10：n-Sarsa

参数：步长$\alpha \in (0,1]$，探索率$\epsilon > 0$，步数$n \in N_+$

初始化：$\forall s \in S, a \in A$，任意初始化$Q(s,a)$，初始化π（如基于Q的ϵ-greedy策略）

对于 每一幕 执行

  初始化s_0为非终止状态，根据$\pi(\cdot|s_0)$选取a_0

  $T \leftarrow \infty$

  对于 $t = 0, 1, 2, \ldots$ 执行

    如果 $t < T$ 那么

      执行a_t，观察r_{t+1}, s_{t+1}

      如果 s_{t+1}是终止状态 那么

        $T \leftarrow t + 1$

      否则

        根据$\pi(\cdot|s_{t+1})$选取a_{t+1}

    $\tau \leftarrow t - n + 1$ ▷ τ是正在更新的状态的时间

    如果 $\tau \geqslant 0$ 那么

      $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$

      如果 $\tau + n < T$ 那么

        $G \leftarrow G + \gamma^n Q(s_{\tau+n}, a_{\tau+n})$

      $Q(S_\tau, A_\tau) \leftarrow Q(s_\tau, a_\tau) + \alpha[G - Q(s_\tau, a_\tau)]$

    如果 $\tau = T - 1$ 那么

算法 10：n-Sarsa

break

OFF-POLICY-N-TD

$$v_{t+n}(s_t) \doteq v_{t+n-1}(s_t) + \alpha\rho_{t:t+n-1}[G_{t:t+n} - v_{t+n-1}(s_t)]$$

其中重要度采样率为目标策略和行为策略采取n个动作的相对概率：

$$\rho_{t:h} \doteq \prod_{k=t}^{\min(h,T-1)} \frac{\pi(a_k|s_k)}{b(a_k|s_k)}$$

算法 11：n-期望Sarsa-off-policy

输入：$b(a|s) > 0$
参数：步长$\alpha \in (0,1]$，探索率$\epsilon > 0$，步数$n \in N_+$
初始化：$\forall s \in S, a \in A$，任意初始化$Q(s,a)$，初始化π
对于 每一幕 执行
    初始化s_0为非终止状态，根据$b(\cdot|s_0)$选取a_0
    $T \leftarrow \infty$
    对于 $t = 0, 1, 2, \ldots$ 执行
        如果 $t < T$ 那么
            执行a_t，观察r_{t+1}, s_{t+1}
            如果 s_{t+1}是终止状态 那么
                $T \leftarrow t+1$
            否则
                根据 $b(\cdot|s_{t+1})$选取a_{t+1}
        $\tau \leftarrow t - n + 1$ ▷ τ是正在更新的状态的时间
        如果 $\tau \geqslant 0$ 那么
            $\rho \leftarrow \prod_{i=\tau+1}^{\min(\tau+n-1,T-1)} \frac{\pi(A_i|S_i)}{b(A_i|S_i)}$ ▷ 重要性采样权重
            $G \leftarrow \sum_{i=\tau+1}^{\min(\tau+n,T)} \gamma^{i-\tau-1} R_i$
            如果 $\tau + n < T$ 那么
                $G \leftarrow G + \gamma^n \sum_a \pi(a|s_{\tau+n}) Q(s_{\tau+n}, a)$ ▷ 期望Sarsa使用期望值
            $Q(s_\tau, a_\tau) \leftarrow Q(s_\tau, a_\tau) + \alpha\rho[G - Q(s_\tau, a_\tau)]$
        如果 $\tau = T - 1$ 那么

返回目录

算法 11：n-期望Sarsa-off-policy
22: break

5.6 对比与补充

Sarsa较为保守，在存在风险的任务中，会避开低回报动作；Q-learning较为乐观，更倾向于探索并找到最优解。在存在陷阱的任务中，Sarsa会比Q-learning取得更好的结果。

补充见12.2.3。

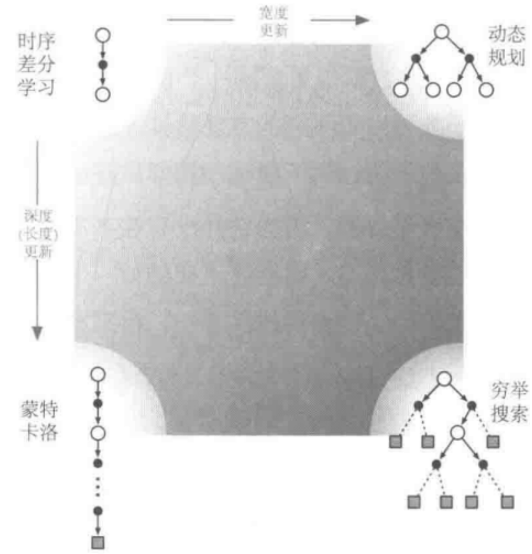
6 表格型方法总结对比

[21] 基于模型的方法（DP、启发式搜索）主要进行规划，无模型的方法（MC、TD）主要进行学习，二者的核心都是值函数的计算。

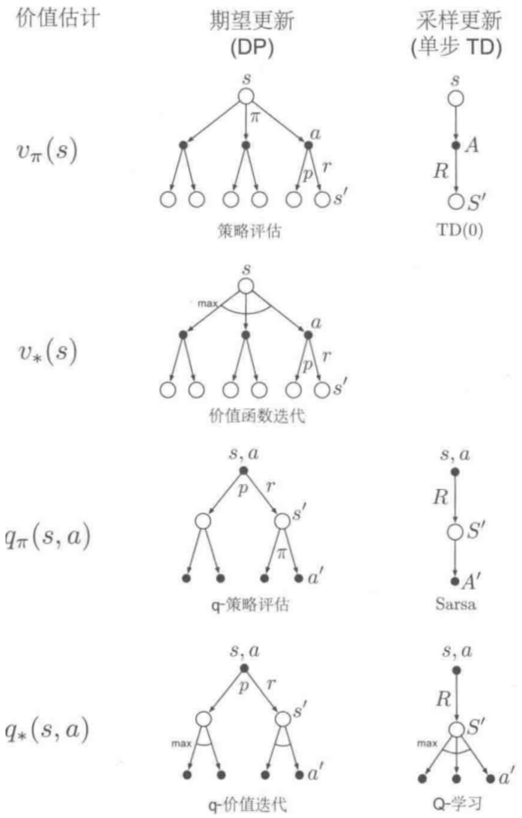
表格型方法介绍 见12.2.4

三个维度

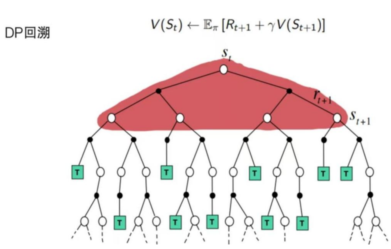
- 更新：期望更新能产生更好的估计，但是需要更多的计算。
- 自举程度。
- 同轨/离轨。



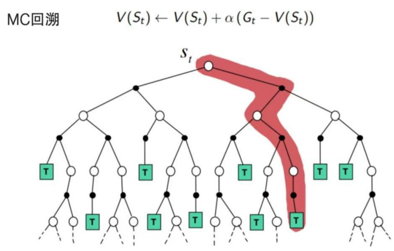
(a) 总结对比



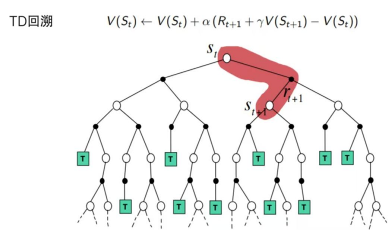
(b) 更新对比



(c) DP表达式



(d) MC表达式



(e) TD表达式

图 14: 表格型方法对比

表达式对比 统一格式：

$$q_{t+1}(s_t, a_t) = q_t(s_t, a_t) + \alpha_t(s_t, a_t)[\bar{q}_t - q_t(s_t, a_t)]$$

返回目录

算法	$\bar{q}_t$表达式	求解目标
Sarsa	$r_{t+1}+\gamma q_t(s_{t+1},a_{t+1})$	贝尔曼方程
n-Sarsa	$r_{t+1}+\gamma r_{t+2}+\cdots+\gamma^n q_t(s_{t+n},a_{t+n})$	
期望Sarsa	$r_{t+1}+\gamma\sum_{a\in A}\pi_t(a\|s_{t+1})q_t(s_{t+1},a)$	
Q-learning	$r_{t+1}+\gamma\max_a q_t(s_{t+1},a)$	贝尔曼最优方程
MC	$r_{t+1}+\gamma r_{t+2}+\cdots$	贝尔曼方程

表 4: 各算法中的$\bar{q}_t$表达式

7 值函数近似

7.1 值函数近似 [22]

$$\hat{v}(s,\omega)\approx v_\pi(s),\omega\in R^d,d\ll|S|$$

目标函数

$$J(\omega)=E[(v_\pi(s)-\hat{v}(s,\omega))^2]$$

对s按重要程度加权：

$$\overline{VE}(\omega)\doteq\sum_{s\in S}\mu(s)[v_\pi(s)-\hat{v}(s,w)]^2$$

一般无法保证最优，求解局部最优。

近似方法

- $v_\pi(s_t)$：
  - MC：G_t。
  - TD：$r_{t+1}+\gamma\hat{v}(s_{t+1},\omega_t)$。
- $\hat{v}(s,\omega)$：
  - 线性参数：$\hat{v}(s,\omega)=\phi(s)^T\omega$，$\phi(s)$为特征函数。
    * 多项式基函数。

返回目录

        * 傅里叶基函数$\phi_i(s) = \cos(i\pi s), s \in [0, 1]$。
        * 径向基函数$\phi_i(s) = e^{-\frac{||s-c_i||^2}{2\sigma_i^2}}$。
        * 表格法可视为特殊情况。
    - 非线性参数：
        * 神经网络：输入状态，网络参数为ω，输出$\hat{v}(s, \omega)$。
        * 决策树。
        * 模糊网络。
    - 非参数方法：核函数（见12.3）、高斯回归等。

优势

- 具有一定泛化能力，适应部分观测问题。
- 曲线拟合：用少量参数储存状态，阶数越高越近似。

7.2 随机梯度下降（SGD）

$$\omega_{k+1} = \omega_k - \alpha_k \nabla_\omega J(\omega_k)$$

其中，

$$\begin{aligned}\nabla_\omega J(\omega) &= \nabla_\omega E[(v_\pi(s) - \hat{v}(s, \omega))^2] \\ &= E[\nabla_\omega (v_\pi(s) - \hat{v}(s, \omega))^2] (\text{有界可换求导与期望顺序}) \\ &= -2E[(v_\pi(s) - \hat{v}(s, \omega)) \nabla_\omega \hat{v}(s, \omega)]\end{aligned}$$

因此$\omega_{k+1} = \omega_k + \underbrace{\alpha}_{\text{步长}} [v_\pi(s_k) - \hat{v}(s_k, \omega_k)] \nabla_\omega \hat{v}(s_k, \omega_k)$。

负梯度方向降速最快 梯度方向增长最快，负梯度方向下降最快。

算法 12：梯度MC
1: 输入：待评估π，可微函数$\hat{v} : S \times R^d \to R$ 2: 参数：步长$\alpha > 0$ 3: 初始化：任意初始化$w \in R^d$

返回目录

算法 12：梯度MC

4: 循环 ▷ 对每一幕
5: 根据π生成一幕交互数据$s_0, a_0, r_1, s_1, a_1, \cdots, r_T, s_T$
6: 对于 $t = 0, 1, \cdots, T-1$ 执行
7: $w_{t+1} \leftarrow w_t + \alpha_t[G_t - \hat{v}(s_t, w_t)]\nabla_\omega \hat{v}(s_t, w_t)$

半梯度下降 只考虑w_t对估计值的影响，而忽略对目标的影响。在使用自举目标时，目标本身依赖于w，有偏。

- 优势：学习速度较快，支持持续在线学习，无需等待幕结束。
- 局限：稳健性差，在非线性函数近似中可能不稳定。

算法 13：半梯度TD(0)

1: 输入：待评估π，可微函数$\hat{v}: S^+ \times R^d \to R, \hat{v}(\text{终止}, \cdot) = 0$
2: 参数：步长$\alpha > 0$
3: 初始化：任意初始化$w \in R^d$
4: 循环 ▷ 对每一幕
5: 初始化s
6: 对于 $t = 0, 1, \cdots, T-1$ 执行
7: 选取$a_t \sim \pi(\cdot|s_t)$并采取，观察r_t, s_{t+1}
8: $w_{t+1} \leftarrow w_t + \alpha_t[r_{t+1} + \gamma\hat{v}(s_{t+1}, w_t) - \hat{v}(s_t, w_t)]\nabla_\omega \hat{v}(s_t, w_t)$
9: 如果 s_{t+1} 为终止状态 那么
10: break

7.3 批方法

最小二乘法减少迭代计算量：

$$LS(\omega) = \sum_{t=1}^{T}[q_t^\pi - \hat{q}(s_t, a, \omega)]^2 = E_D[(q^\pi - \hat{q}(s, a, \omega))^2]$$

返回目录

7.4 DQN（Deep Q-Network，深度Q网络）[23]

利用卷积神经网络作为非线性函数近似器，最小化损失函数，适用于高维空间：

$$J(\omega) = E\{[R + \gamma \max_{a' \in A(S')} \hat{q}(S', a', \underbrace{\omega^-}_{\text{目标网络}}) - \hat{q}(S, A, \underbrace{\omega}_{\text{主网络}})]^2\}$$

目标网络：$y^{DQN} = r + \gamma \max_{a'} Q(s', a'; \theta^-)$。

主要技术

- 两个网络：主网络$\hat{q}(s, a, \omega)$和目标网络$\hat{q}(s', a', \omega^-)$，后者参数阶段性从前者同步。
  - 防止过拟合：
    - 随机丢弃法（dropout）。
    - 批量归一化（batch normalization）。
    - 残差直连边。
  - 更新：
    - 软更新：部分更新。
    - 硬更新：直接复制。
- 经验回放（Experience Replay）[24]：存储经验到固定大小的回放缓冲区，训练时从中随机选取。可以打乱样本相关性，提升训练稳定性。可改进为优先经验回放。
- 帧堆叠：将图像作为神经网络输入时，堆叠多帧图像作为输入，并跳帧选取放入帧，增加时间信息。
- 奖励裁剪（Reward Clipping）：将奖励限制在特定范围内（甚至使用符号函数），避免大奖励幅度波动，提升训练稳定性，适用于奖励范围差异大的环境。

算法 14：DQN

```
1: 初始化：ω, ω⁻，经验回放缓冲区B = {(s, a, r, s′)}，计数器t ← 0
2: 循环
3:     如果 t mod C = 0 那么                    ▷ 每隔C步更新目标网络（初始化一致）
4:         ω⁻ ← ω
5:     从B中均匀采样小批量样本{(s, a, r, s′)}
6:     对于 每个样本 执行
```

算法 14：DQN

如果 s' 是终止状态 那么

$y \leftarrow r$

否则

$y \leftarrow r + \gamma \max_{a'} \hat{q}(s', a', \omega^-)$ ▷ 计算目标值

使用小批量样本$\{(s, a, y)\}$更新主网络参数ω，最小化损失$[y - \hat{q}(s, a, \omega)]^2$

$t \leftarrow t + 1$

DOUBLE-DQN 两个值函数逼近网络，一个选择动作，一个评估值函数。

目标网络：$y^{DDQN} = r + \gamma Q[s', \operatorname{argmax}_a Q(s_{t+1}, a, \theta_t), \theta^-]$。

8 策略梯度（POLICY GRADIENT）

8.1 概念 [25]

将策略参数化，搜索策略空间，是同轨策略：

$$\pi(a|s, \theta) = \pi_\theta(a|s)$$

梯度与梯度上升（倪推导）

学习θ使以下指标最大。

- 平均状态价值：

$$\bar{v}_\pi = \sum_{s \in S} d(s) v_\pi(s) = E[v_\pi(S)]$$

  其中$d(s) \geqslant 0$为s的权重，$\sum_{s \in S} d(s) = 1$，其可由以下方法选取：

  - 均匀分布：$d(s) = \frac{1}{|S|}$。
  - 只关心s_0：$d(s_0) = 1, d(s \neq s_0) = 0$。
  - 平稳分布：$d_\pi^T P_\pi = d_\pi^T$，根据访问频次赋予概率。

返回目录

- 平均单步奖励：

$$
\begin{aligned}
\bar{r}_\pi = \sum_{s\in S} \underbrace{d_\pi(s)}_{\text{平稳分布}} \underbrace{r_\pi(s)}_{\sum_{a\in A}\pi(a|s)\underbrace{r(s,a)}_{\sum_{r\in R} rp(r|s,a)}} &= E[r_\pi(S)] \\
= \lim_{n\to\infty}\frac{1}{n}E[\sum_{k=1}^{n} R_{t+k}] = \lim_{n\to\infty}\frac{1}{n}E[\sum_{k=1}^{n} R_{t+k}|S_t = s_0]
\end{aligned}
$$

梯度为：

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \sum_{s\in S}\eta(s)\sum_{a\in A}\nabla_\theta\pi(a|s,\theta)q_\pi(s,a) \\
&= \sum_{s\in S}\eta(s)\sum_{a\in A}\pi(a|s,\theta)\nabla_\theta \ln\pi(a|s,\theta)q_\pi(s,a) \\
&= E[\nabla_\theta \ln\pi(A|S,\theta)q_\pi(S,A)] \\
&\approx \nabla_\theta \ln\pi(a|s,\theta)q_\pi(s,a)(\text{采样近似})
\end{aligned}
$$

为确保$\pi(a|s,\theta) > 0$，使用softmax函数，$\pi(a|s,\theta) = \frac{e^{h(s,a,\theta)}}{\sum_{a'\in A} e^{h(s,a',\theta)}}$。

$$
\begin{aligned}
\theta_{t+1} &= \theta_t + \alpha\nabla_\theta J(\theta) = \theta_t + \alpha E[\nabla_\theta \ln\pi(A|S,\theta_t)q_\pi(S,A)] \\
&= \theta_t + \alpha \underbrace{\nabla_\theta \ln\pi(a_t|s_t,\theta_t)}_{\beta_t=\frac{q_\pi(s_t,a_t)}{\pi(a_t|s_t,\theta_t)}} \underbrace{q_\pi(s_t,a_t)}_{q(s_t,a_t)\text{近似}}(\text{随机梯度})
\end{aligned}
$$

- $\alpha\beta_t$足够小时，若$\beta_t > 0$，则选择(s_t,a_t)的概率增加，且幅度与β_t正相关。
- β_t与$q_\pi(s_t,a_t)$正相关，与$\pi(a_t|s_t,\theta_t)$负相关，倾向于选择高价值动作，探索低概率动作。

似然率策略梯度（郭推导）

记$R(\tau) = \sum_{t=0}^{H} R(s_t,u_t)$，目标函数为$U(\theta) = \sum_\tau P(\tau,\theta)R(\tau)$，其梯度为：

$$
\begin{aligned}
\nabla_\theta U(\theta) &= \nabla_\theta \sum_\tau P(\tau,\theta)R(\tau) = \sum_\tau \nabla_\theta P(\tau,\theta)R(\tau)(\text{运算换序}) \\
&= \sum_\tau P(\tau,\theta)\frac{\nabla_\theta P(\tau,\theta)}{P(\tau,\theta)}R(\tau) = \sum_\tau P(\tau,\theta)\nabla_\theta \ln P(\tau,\theta)R(\tau)(\text{复合求导})
\end{aligned}
$$

经验平均为：

$$\nabla_\theta U(\theta) \approx \hat{g} = \frac{1}{m}\sum_{i=1}^{m} \nabla_\theta \ln P(\tau, \theta) R(\tau)$$

其无偏但方差很大，其中

$$\begin{aligned}\nabla_\theta \ln P(\tau^{(i)}, \theta) &= \nabla_\theta \ln[\prod_{t=0}^{H} P(s_{t+1}^{(i)}|s_t^{(i)}, u_t^{(i)}) \cdot \pi_\theta(u_t^{(i)}|s_t^{(i)})] \\ &= \nabla_\theta[\underbrace{\sum_{t=0}^{H} \ln P(s_{t+1}^{(i)}|s_t^{(i)}, u_t^{(i)})}_{\text{与动力学无关}} + \underbrace{\sum_{t=0}^{H} \ln \pi_\theta(u_t^{(i)}|s_t^{(i)})]}_{\text{只与策略相关}} \\ &= \nabla_\theta[\sum_{t=0}^{H} \ln \pi_\theta(u_t^{(i)}|s_t^{(i)})] = \underbrace{\sum_{t=0}^{H} \nabla_\theta \ln \pi_\theta(u_t^{(i)}|s_t^{(i)})}_{\text{策略梯度}}\end{aligned}$$

减小方差：

- 基线。
- 修改值函数。

优势

- 可以逼近确定性策略，可以逼近任意概率分布，不受$q(s,a)$限制，策略是更简单的函数逼近。
- 策略参数化更容易加入先验知识。
- 在状态空间大时，存储和泛化能力强。

8.2 REINFORCE（MC-policy gradient）[26]

用MC估计$q_\pi(s,a)$，使用与θ无关的G_t代替$q_\pi(s_t,a_t)$：

$$\theta_{t+1} \doteq \theta_t + \alpha G_t \nabla_\theta \ln \pi(a_t|s_t, \theta_t)$$

算法 15：REINFORCE

输入：可微分的$\pi(a|s,\theta)$
参数：步长$\alpha > 0$，折扣因子$\gamma \in (0,1)$
初始化：初始化$\theta \in R^{d'}$
循环
  按照$\pi(\cdot|\cdot,\theta)$生成一幕$s_0, a_0, r_1, \cdots, s_{T-1}, a_{T-1}, r_T$
  对于 $t = 0, 1, \cdots, T-1$ 执行
    $G_t \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} R_k$
    $\theta \leftarrow \theta + \alpha G_t \nabla_\theta \ln \pi(a_t|s_t,\theta_t)$

9 ACTOR-CRITIC方法

9.1 概念 [27]

结合策略梯度和价值方法，一般是on-policy方法。

$$\theta_{t+1} = \theta_t + \alpha \nabla_\theta \ln \pi(a_t|s_t,\theta_t) q_\pi(s_t, a_t)$$

- 演员（Actor）：策略更新，用于采取行动，对应算法更新。
- 评论家（Critic）：策略评估或价值估计，用于评判策略，对应估计$q_\pi(s,a)$，采用TD方法。

算法 16：QAC

初始化：策略参数θ和评论家参数w
对于 每个回合 执行
  对于 $t = 0, 1, 2, \ldots, T-1$ 执行
    根据$\pi(a|s_t,\theta_t)$选择a_t，观察r_{t+1}, s_{t+1}，再根据$\pi(a|s_{t+1},\theta_t)$选择a_{t+1}
    $\delta_t = r_{t+1} + \gamma q(s_{t+1}, a_{t+1}, w_t) - q(s_t, a_t, w_t)$ ▷ TD误差
    $w_{t+1} = w_t + \alpha_w \delta_t \nabla_w q(s_t, a_t, w_t)$ ▷ 评论家价值更新
    $\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \ln \pi(a_t|s_t,\theta_t) q(s_t, a_t, w_{t+1})$ ▷ 演员策略更新

返回目录

9.2 基线优势 [28]

基本的Actor-Critic方法有较大方差，引入基线降低。

基线

$$\nabla_\theta J(\theta) = E_{S\sim\eta, A\sim\pi}\{\nabla_\theta \ln \pi(A|S, \theta_t)[q_\pi(S, A) - b(S)]\}$$

策略梯度期望不变：

$$\begin{aligned}
E_{S\sim\eta, A\sim\pi}[\nabla_\theta \ln \pi(A|S, \theta_t) b(S)] &= \sum_{s\in S} \eta(s) \sum_{a\in A} \nabla_\theta \pi(a|s, \theta_t) b(s) \\
&= \sum_{s\in S} \eta(s) b(s) \sum_{a\in A} \nabla_\theta \pi(a|s, \theta_t) \\
&= \sum_{s\in S} \eta(s) b(s) \nabla_\theta 1 (\text{交换求和与求导}) \\
&= 0
\end{aligned}$$

为使策略梯度方差最小化，求偏导得：

$$\begin{aligned}
b^*(s) &= \frac{E_{A\sim\pi}[||\nabla_\theta \ln \pi(A|s, \theta_t)||^2 q_\pi(s, A)]}{E_{A\sim\pi}[||\nabla_\theta \ln \pi(A|s, \theta_t)||^2]} \\
&= E_{A\sim\pi}[q_\pi(s, A)] (\text{省略权重}) \\
&= v_\pi(s)
\end{aligned}$$

如果直接用$b(s) = q_\pi(s, a)$，会导致策略梯度为0。

优势函数

$$\theta_{t+1} = \theta_t + \alpha E\{\nabla_\theta \ln \pi(A|S, \theta_t) \underbrace{[q_\pi(S, A) - v_\pi(s)]}_{\text{优势函数}\delta_\pi(S,A)}\}$$

此时，$\beta_t = \frac{\delta_\pi(s_t, s_t)}{\pi(a_t|s_t, \theta_t)}$，正相关项为相对值，而非绝对值，更合理。
使用TD进行近似：

$$\delta_t = q_t(s_t, a_t) - v_t(s_t) \approx r_{t+1} + \gamma v_t(s_{t+1}) - v_t(s_t)$$

这时只需要一个网络进行估计。

算法 17：A2C

初始化：策略参数θ和评论家参数w
对于 每个回合 执行
    对于 $t = 0, 1, 2, \ldots, T-1$ 执行
        根据$\pi(a|s_t, \theta_t)$选择a_t，执行后观察r_{t+1}, s_{t+1}
        $\delta_t = r_{t+1} + \gamma v(s_{t+1}, w_t) - v(s_t, w_t)$ ▷ 优势函数
        $w_{t+1} = w_t + \alpha_w \delta_t \nabla_w v(s_t, w_t)$ ▷ 评论家价值更新
        $\theta_{t+1} = \theta_t + \alpha_\theta \delta_t \nabla_\theta \ln \pi(a_t|s_t, \theta_t)$ ▷ 演员策略更新

OFF-POLICY 加入重要性采样由on-policy变成off-policy：

$$\nabla_\theta J(\theta) = E_{S \sim \rho, A \sim \beta}[\frac{\pi(A|S, \theta)}{\beta(A|S)} \nabla_\theta \ln \pi(A|S, \theta) q_\pi(S, A)]$$

其中β是行为策略，ρ是状态分布。其也可以采用上述基线$b^*(s)$。

算法 18：重要性采样off-policy Actor-Critic

初始化：$\beta(a|s), \pi(a|s, \theta_0), v(s, w_0)$
对于 每个回合 执行
    对于 $t = 0, 1, 2, \ldots, T-1$ 执行
        根据$\beta(s_t)$选择a_t，观察r_{t+1}, s_{t+1}。
        $\delta_t = r_{t+1} + \gamma v(s_{t+1}, w_t) - v(s_t, w_t)$ ▷ 优势函数
        $w_{t+1} = w_t + \alpha_w \frac{\pi(a_t|s_t, \theta_t)}{\beta(a_t|s_t)} \delta_t \nabla_w v(s_t, w_t)$ ▷ 评论家价值更新
        $\theta_{t+1} = \theta_t + \alpha_\theta \frac{\pi(a_t|s_t, \theta_t)}{\beta(a_t|s_t)} \delta_t \nabla_\theta \ln \pi(a_t|s_t, \theta_t)$ ▷ 演员策略更新

9.3 TRPO与PPO [29]

9.3.1 *TRPO*（*Trust Region Policy Optimization*，信赖域策略优化）

限制每次策略更新的幅度，保证稳定性和单调提升。

返回目录

替代回报函数

$$\eta(\tilde{\pi}) = \eta(\pi) + \underbrace{E_{s_0,a_0,\cdots\sim\tilde{\pi}}[\sum_{t=0}^{\infty}\gamma^t A_\pi(s_t,a_t)]}_{\text{新旧策略回报差}}$$
$$= \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s)\sum_a \tilde{\pi}(a|s)A^\pi(s,a)$$
$$L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s)\sum_a \tilde{\pi}(a|s)A^\pi(s,a)(\text{忽略状态分布变化})$$
$$= \eta(\pi) + E_{s\sim\rho_{\theta_{old}},a\sim\pi_{\theta_{old}}}[\frac{\tilde{\pi}_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}A_{\theta_{old}}(s,a)](\text{重要性采样动作分布})$$

有$L_{\pi_{\theta_{old}}}(\pi_{\theta_{old}}) = \eta(\pi_{\theta_{old}}), \nabla_\theta L_{\pi_{\theta_{old}}}(\pi_\theta)|_{\theta=\theta_{old}} = \nabla_\theta\eta(\pi_\theta)|_{\theta=\theta_{old}}$。
令$\alpha = D_{TV}^{max}(\pi,\tilde{\pi}), \epsilon = \max_{s,a}|A_\pi(s,a)|$，惩罚因子$C = \frac{2\epsilon\gamma}{(1-\gamma)^2}$，则有：

$$\eta(\tilde{\pi}) \geqslant L_\pi(\tilde{\pi}) - \frac{4\epsilon}{(1-\gamma)^2}\alpha^2 \quad \eta(\tilde{\pi}) \geqslant L_\pi(\tilde{\pi}) - CD_{KL}^{max}(\pi,\tilde{\pi})$$

优化：共轭梯度搜索　问题转化为：

$$\max_\theta[L_{\theta_{old}}(\theta) - CD_{KL}^{max}(\theta_{old},\theta)]$$
$$\xRightarrow{\text{迭代步长较小}} \max_\theta E_{s\sim\rho_{\theta_{old}},a\sim\pi_{\theta_{old}}}[\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}A_{\theta_{old}}(s,a)], \quad D_{KL}^{max}(\theta_{old},\theta) \leqslant \delta$$
$$\xRightarrow[\text{平均KL散度}]{\text{策略状态空间分布}} \max_\theta E_{s\sim\pi_{\theta_{old}},a\sim\pi_{\theta_{old}}}[\frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)}A_{\theta_{old}}(s,a)], \quad E_{s\sim\pi_{\theta_{old}}}\{D_{KL}[\pi_{\theta_{old}}(\cdot|s)\pi_\theta(\cdot|s)]\} \leqslant \delta$$
$$\xRightarrow[\text{约束二次逼近}]{\text{目标线性逼近}} \max_\theta[\nabla_\theta L_{\theta_{old}}(\theta)|_{\theta=\theta_{old}}\cdot(\theta-\theta_{old})], \quad \frac{1}{2}(\theta_{old}-\theta)^T\underbrace{A(\theta_{old})}_{\text{Fisher信息矩阵}}(\theta_{old}-\theta) \leqslant \delta$$

9.3.2 *PPO*（*Proximal Policy Optimization*，近端策略优化）

限制新旧策略的变化幅度，保证策略更新的稳定性，简化TRPO的实现并提升效率。

$$L^{CLIP}(\theta) = E_t[\min\{r_t(\theta)\hat{A}_t, \text{clip}[r_t(\theta),1-\epsilon,1+\epsilon]\hat{A}_t\}]$$
$$L_t^{CLIP+VF+S}(\theta) = \hat{E}_t\{L_t^{CLIP}(\theta) - c_1L_t^{VF}(\theta) + c_2S[\pi_\theta](s_t)\}$$

其中，

- 值函数损失函数L_t^{VF}。
- 熵S。

返回目录

- 优势函数估计$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1}$，$\delta$为TD误差。

算法 19：PPO

初始化θ, θ_{old}
对于 每个回合 执行
    对于 actor $= 1, 2, \ldots, N$ 执行
        用$\pi_{\theta_{old}}$采集T步序列$\{(s_t, a_t, r_t, s_{t+1})\}_{t=1}^{T}$
        优势估计$\hat{A}_t$
        回报$\hat{R}_t$
    汇总所有样本，数量为NT
    对于 $k = 1, 2, \ldots, K$ 执行
        随机采样M个minibatch
        概率比$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$
        clip损失：$L^{CLIP} = \min(r_t(\theta)\hat{A}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)$
        值函数损失L^{VF}和熵正则项$S[\pi_\theta](s_t)$
        总损失$L = E[L^{CLIP} - c_1 L^{VF} + c_2 S]$
        对θ梯度下降优化L
    $\theta_{old} \leftarrow \theta$

9.4 确定性策略Actor-Critic方法（off-policy）

9.4.1 *DPG*

在策略是确定性时：

$$\nabla_\theta J(\theta) = \sum_{s\in S} \underbrace{\rho_\mu(s)}_{\text{状态分布}} \nabla_\theta \mu(s)[\nabla_a q_\mu(s,a)]\Big|_{a=\mu(s)} = E_{S\sim\rho_\mu}\{\nabla_\theta \mu(S)[\nabla_a q_\mu(S,a)]\Big|_{a=\mu(S)}\}$$

算法 20：确定性策略Actor-Critic（DPG）

初始化：$\beta(a|s), \mu(s, \theta_0), q(s, a, w_0)$。 ▷ β可用μ+噪声替代
对于 每个回合 执行
    对于 $t = 0, 1, 2, \ldots, T-1$ 执行
        根据$\beta(s_t)$生成a_t，观察r_{t+1}, s_{t+1}

返回目录

算法 20：确定性策略Actor-Critic（DPG）

$\delta_t = r_{t+1} + \gamma q(s_{t+1}, \mu(s_{t+1}, \theta_t), w_t) - q(s_t, a_t, w_t)$ ▷ 优势函数

$w_{t+1} = w_t + \alpha_w \delta_t \nabla_w q(s_t, a_t, w_t)$ ▷ 评论家价值更新

$\theta_{t+1} = \theta_t + \alpha_\theta \nabla_\theta \mu(s_t, \theta_t)[\nabla_a q(s_t, a, w_{t+1})]|_{a=\mu(s_t)}$ ▷ 演员策略更新

9.4.2 DDPG [30]

结合DQN和DPG，演员、评论家各有主网络和目标网络，一共四个网络。为在确定性策略中保障探索性，引入噪声。

OU（ORNSTEIN-UHLENBECK）噪声

均值回归随机过程$dX_t = \underbrace{\theta}_{\text{回归速度参数}}(\underbrace{\mu}_{\text{长期均值}} - X_t)dt + \underbrace{\sigma}_{\text{噪声强度}}\underbrace{dW_t}_{\text{维纳过程}}$，离散为：

$$X_{t+1} = X_t + \theta(\mu - X_t) + \sigma\epsilon_t$$

- 相邻时刻噪声值相关，适合连续控制任务，使动作平滑变化。
- 噪声逐渐回归均值，避免长期偏离，提供了自然的探索衰减机制。

算法 21：DDPG

初始化：随机初始化评论家网络$Q(s, a|\theta^Q)$和演员网络$\mu(s|\theta^\mu)$，初始化目标网络Q', μ'，使$\theta^{Q'} \leftarrow \theta^Q, \theta^{\mu'} \leftarrow \theta^\mu$，初始化经验回放池R

对于 每个回合 执行

  初始化探索噪声过程N

  接收初始状态s_1

  对于 $t = 1, \ldots, T$ 执行

    选择$a_t = \mu(s_t|\theta^\mu) + N_t$ ▷ 带噪声的确定性策略

    执行a_t，观察r_t, s_{t+1}

    存储转移(s_t, a_t, r_t, s_{t+1})到R ▷ 经验回放

    从R中随机采样N个小批量(s_i, a_i, r_i, s_{i+1})

    目标值$y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$ ▷ 目标网络

    更新评论家：最小化损失$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$

    更新演员：$\nabla_{\theta^\mu} J \approx \frac{1}{N}\sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu}\mu(s|\theta^\mu)|_{s_i}$

返回目录

算法 21：DDPG

软更新目标网络参数

10 策略搜索方法总结对比

[31]

- 策略搜索方法
  - 无模型
    - 随机策略
      - 策略梯度：TRPO
      - 统计学习
      - 路径积分
    - 确定性策略：DDPG
    - EM方法
    - 粒子滤波方法
  - 基于模型：指导策略搜索(GPS)，传统控制器+监督学习

算法	REINFORCE	PPO	DDPG
策略类型	随机策略 on-policy	随机策略 on-policy	确定性策略 off-policy
适用场景	离散动作空间	离散/连续动作空间	连续动作空间
网络架构	策略网络	策略、价值网络	演员、评论家网络（主+目标）
回报	完整轨迹回报G_t	优势函数$\hat{A}_t$	评论家Q值
探索	策略分布本身的随机性		OU噪声
稳定性	方差大	限制策略更新大小	目标网络、经验回放、软更新
样本效率	低（需完整轨迹）	中等	高（可复用样本）

表 5: REINFORCE、PPO、DDPG比较

11 多智能体强化学习

11.1 概念 [32]

分类

- 完全协作。
- 完全竞争。
- 混合策略。

挑战

- 单智能体视角下，其它智能体是动态的，值函数为相互依赖的联合值函数。
- 智能体间存在博弈关系，求均衡解。

博弈论

- 参与人行动先后顺序：
  - 静态博弈：同时行动。
  - 动态博弈：有先后行动顺序，常用博弈树拓展表述。
- 参与人知识储备：
  - 完美信息博弈：已知相关信息。
  - 非完美信息博弈：不完全知道相关信息。

完美信息静态博弈的纳什均衡策略（NASH EQUILIBRIUM） 所有智能体以最佳策略应对，全理性，没有智能体能单独偏离自身策略来改善自身回报。所有智能体采取纳什均衡策略应比部分智能体采取纳什均衡策略的价值高。

纳什均衡求解（双人零和博弈） [33]

双人策略为π_1, π_2，玩家i值函数为$V_i = \pi_1 R_i \pi_2^T$，有$R_1 = -R_2 = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}$。纳什均衡指$V_i(\pi_i^*, \pi_{-i}^*) \geqslant V_i(\pi_i, \pi_{-i}^*)$，求解等价于：

$$\max_{\pi_i} \min_{\pi_{-i}} \sum_{a_i \in A_i} R_i^T \pi_i(a_i)$$

返回目录

可转化为线性规划问题，用单纯形法求解：

$$
\begin{aligned}
&\max V_1\\
&\begin{cases}
r_{11}p_1 + r_{21}p_2 \geqslant V_1\\
r_{12}p_1 + r_{22}p_2 \geqslant V_1\\
p_1 + p_2 = 1
\end{cases}\\
&p_j \geqslant 0, j = 1, 2
\end{aligned}
$$

非完美信息博弈的扩展式博弈 七元组$\{H, Z, P, p, u, I, \sigma_c\}$，分别是当前节点已知所有信息（包含个人私有信息），终止状态集合，玩家集合，非终止状态到玩家映射，非终止状态到实数映射，终止状态到实数映射（玩家到终止状态时获得的回报），信息集，策略。

11.2 算法 [34]

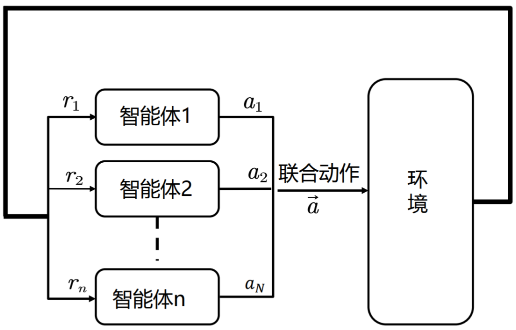


图 15: 多智能体强化学习

- 完全协作。
- 完全竞争：MinimaxQ-learning算法。
- 混合策略：纳什Q-learning算法，Friend-or-foe Q-learning算法，wolf策略爬山算法。
- 基于微分对策略。
- 深度强化学习算法。

返回目录

算法 22：MADDPG

初始化：初始化值网络、动作网络和目标网络，初始化经验回放池D，初始化状态s
对于 每一幕 执行
  初始化随机过程N进行动作探索
  对于 每一步 执行
    $a_i = \mu_{\theta_i}(o_i) + N_i$ ▷ 随机策略选择动作
    执行$a = (a_1, \cdots, a_N)$，观测r, s'
    将(s, a, r, s')存储到D中
    $s \leftarrow s'$
    对于 每一个智能体 执行
      从D中随机采样数据(s^j, a^j, r^j, s'^j)
      $y^j = r_i^j + \gamma Q_i^{\mu'}(s'^j, a_1', \cdots, a_N')|_{a_k'=\mu_k'(o_k^j)}$ ▷ TD目标
      值网络：$L(\theta_i^Q) = \frac{1}{S}\sum_j (y^j - Q_i^{\mu}(s^j, a_1^j, \cdots, a_N^j))^2$ ▷ 最小化损失函数
      动作网络：$\nabla_{\theta_i} J = \frac{1}{S}\sum_j \nabla_{\theta_i}\mu_i(a_i|o_i)\nabla_{a_i} Q_i^{\mu}(s, a_1, \cdots, a_N)|_{a_i=\mu_i(o_i)}$ ▷ 策略梯度
      目标网络：$\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i'$

12 附录

12.1 概念与原理

12.1.1 历史与发展

1. 源于动物学习心理学的试错法：效应定律（Edward Thorndike），条件反射（巴普洛夫），快乐-痛苦系统（图灵），向“老师”学习到向“评论家”学习，自动学习机（M.L.Tsetlin），分类器系统（救火队算法和遗传算法）。
2. 最优控制：贝尔曼方程与马尔可夫决策过程（Richard Bellman），维度灾难。
3. 时序差分方法：次级强化物，广义强化（Klopf），与试错法结合（actor-critic方法，Sutton），与最优控制结合（Q-learning，Chris Watkins）。

返回正文1。

12.1.2 贝尔曼最优方程求解

收缩映射定理 若$f(x)$是收缩映射，则存在唯一一个不动点x^*满足$f(x^*) = x^*$。针对$x_{k+1} = f(x_k)$，在$x_k \to x^*, k \to \infty$的过程中，收敛速度成指数级增长。

- 存在性：$\|x_{k+1} - x_k\| = \|f(x_{k+1}) - f(x_k)\| \leqslant \gamma\|x_k - x_{k-1}\| \leqslant \cdots \leqslant \gamma^k\|x_1 - x_0\| \xRightarrow{\gamma<1,\gamma^k\to 0} x_{k+1} - x_k \to 0$。同理可得$\|x_m - x_n\| \leqslant \frac{\gamma^n}{1-\gamma}\|x_1 - x_0\| \to 0$，进而得到$\{x_k\}$是收敛数列，存在$\lim_{k\to\infty} x_k = x^*$。
- 唯一性：$\|f(x_k) - x_k\| = \|x_{k+1} - x_k\|$，其快速收敛到0，则在极限处有不动点$f(x^*) = x^*$。假设存在另一不动点，其必与该不动点相等。
- 指数级收敛：$\|x^* - x_n\| = \lim_{m\to\infty} \|x_m - x_n\| \leqslant \frac{\gamma^n}{1-\gamma}\|x_1 - x_0\| \to 0$。

贝尔曼最优方程的伸缩映射性

$\forall v1, v2$，有贝尔曼最优方程$\pi_i^* \doteq \arg\max_\pi(r_\pi + \gamma P_\pi v_i)$，
故$f(v_i) = \max_\pi(r_\pi + \gamma P_\pi v_i) = r_{\pi_i^*} + \gamma P_{\pi_i^*} v_i \geqslant r_{\pi_j^*} + \gamma P_{\pi_j^*} v_i (i \neq j)$，则

$$
\begin{aligned}
f(v_1) - f(v_2) &= r_{\pi_1^*} + \gamma P_{\pi_1^*} v_1 - (r_{\pi_2^*} + \gamma P_{\pi_2^*} v_2) \\
&\leqslant r_{\pi_1^*} + \gamma P_{\pi_1^*} v_1 - (r_{\pi_1^*} + \gamma P_{\pi_1^*} v_2) \\
&= \gamma P_{\pi_1^*}(v_1 - v_2)
\end{aligned}
$$

同理有$f(v_2) - f(v_1) \leqslant \gamma P_{\pi_2^*}(v_2 - v_1)$，故$\gamma P_{\pi_2^*}(v_1 - v_2) \leqslant f(v_1) - f(v_2) \leqslant \gamma P_{\pi_1^*}(v_1 - v_2)$，
取边界极值z，有$|f(v_1) - f(v_2)| \leqslant z$，即$\|f(v_1) - f(v_2)\|_\infty \leqslant \|z\|_\infty$。
又有$\|z\|_\infty = \max_i |z_i| \leqslant \gamma\|v_1 - v_2\|_\infty$，所以$\|f(v_1) - f(v_2)\|_\infty \leqslant \gamma\|v_1 - v_2\|_\infty$。

贝尔曼最优方程解的性质

- 唯一性：唯一解v^*能通过$v_{k+1} = f(v_k) = \max_{\pi\in\Pi}(r_\pi + \gamma P_\pi v_k)$迭代求解，其对应策略$\pi^* = argmax_{\pi\in\Pi}(r_\pi + \gamma P_\pi v^*)$。
- 最优性（$v^* = v_{\pi^*} \geqslant v_\pi$）：由$v_\pi = r_\pi + \gamma P_\pi v_\pi$和$v^* = \max_\pi(r_\pi + \gamma P_\pi v^*) = r_{\pi^*} + \gamma P_{\pi^*} v^* \geqslant r_\pi + \gamma P_\pi v^*$，可得$v^* - v_\pi \geqslant (r_\pi + \gamma P_\pi v^*) - (r_\pi + \gamma P_\pi v_\pi) = \gamma P_\pi(v^* - v_\pi)$，即有$v^* - v_\pi \geqslant \gamma P_\pi(v^* - v_\pi) \geqslant \cdots \geqslant \gamma^n P_\pi^n(v^* - v_\pi)$，由于$\gamma < 0$，$\forall p_{ij} \in P_\pi, p_{ij} \leqslant 1$，$\lim_{n\to\infty} \gamma^n P_\pi^n(v^* - v_\pi)$趋于0，所以$v^* \geqslant v_\pi$。

返回正文2.2.2。

返回目录

12.2 表格型方法

12.2.1 *DP*补充

1. 异步动态规划：使用任意可用状态值，以任意顺序更新，避免遍历更新，减小计算量。
2. 广义策略迭代（GPI）：策略评估和策略改进以更细粒度进行交替，可视为竞争与合作。
3. 效率：DP的时间复杂度是动作与状态数量的多项式级，在面对维度灾难时，优于线性规划和直接搜索。

返回正文3.3。

12.2.2 *MC*补充

减小重要性采样的方差

- 折扣敏感：把γ视作幕终止的概率，得到第n步终止的无折扣部分回报$\sum_{i=1}^{n} R_{t+i}$，即平价部分回报。全回报$G_t = \sum_{i=1}^{T-t} \gamma^{i-1} R_{t+i}$可视为各平价部分回报的加权和，即该步截止得到的回报与概率之积的和。适用于普通型和加权型。
- 每次决策型：$E[\rho_{t:T-1} G_t] = E[\tilde{G}_t] = E[\sum_{i=1}^{T-t} \gamma^{i-1} \rho_{t:t+i-1} R_{t+i}]$。适用于普通型。

返回正文15。

12.2.3 *TD*补充

改进方法

- 批量更新：值函数根据增量和改变，在处理整批数据后才更新。
- 最大化偏差：贪心策略和柔性策略都在隐式估计最大值，会产生正偏差，致使回报值偏离，带来明显错误决策。
- 后位状态：利用先验知识，知晓动作后状态，并有后位值函数。在后位状态相同的时候可以迁移，减少计算量。

返回目录

带控制变量的每次决策模型

为保证不被选择的动作不会因$\rho_t = 0$而回报为0，使方差较大，采取以下n步回报off-policy方法：

$$G_{t:h} \doteq \rho_t(r_{t+1} + \gamma G_{t+1:h}) + \underbrace{(1-\rho_t)v_{h-1}(s_t)}_{\text{控制变量}}$$

其中控制变量保证$\rho_t = 0$时估计值不收缩，且不改变更新期望。

可写为以下递归形式：

$$\begin{aligned} G_{t:h} &\doteq r_{t+1} + \gamma[\rho_{t+1}G_{t+1:h} + \bar{v}_{h-1}(s_{t+1}) - \rho_{t+1}Q_{h-1}(s_{t+1}, a_{t+1})] \\ &= r_{t+1} + \gamma\rho_{t+1}[G_{t+1:h} - Q_{h-1}(s_{t+1}, a_{t+1})] + \gamma\bar{v}_{h-1}(s_{t+1}) \end{aligned}$$

树回溯

树回溯不使用重要度采样，缓解了off-policy因所学内容相关性小比on-policy慢的问题。相比以沿途收益和底部节点估计值为更新目标的算法，树回溯的更新源于整个树的行为值估计，即各叶子节点的行为值估计按出现概率加权。

单步回溯树：

$$G_{t:t+1} \doteq r_{t+1} + \gamma\sum_a \pi(a|s_{t+1})Q_t(s_{t+1}, a)$$

n-回溯树（递归形式），其对路径可能分支进行展开，不进行采样：

$$G_{t:t+n} \doteq r_{t+1} + \gamma\sum_{a\neq a_{t+1}} \pi(a|s_{t+1})Q_{t+n-1}(s_{t+1}, a) + \gamma\pi(a_{t+1}|s_{t+1})G_{t+1:t+n}$$

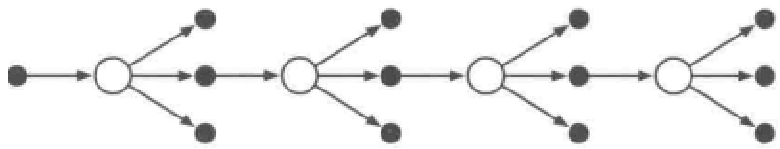


图 16: n-树回溯回溯图

算法 23：n-树回溯
1: 参数：步长$\alpha \in (0, 1], n \in N_+$
2: 初始化：$\forall s \in S, a \in A$，任意初始化$Q(s, a)$，初始化π
3: 对于 每一幕 执行

返回目录

算法 23：n-树回溯

初始化s_0为非终止状态，根据它任意选取a_0
$T \leftarrow \infty$
对于 $t = 0, 1, 2, \ldots$ 执行
  如果 $t < T$ 那么
    执行a_t，观察r_{t+1}, s_{t+1}
    如果 s_{t+1}是终止状态 那么
      $T \leftarrow t + 1$
    否则
      根据s_{t+1}选取a_{t+1}
  $\tau \leftarrow t - n + 1$ ▷ τ是正在更新的状态的时间
  如果 $\tau \geqslant 0$ 那么
    如果 $t + 1 \geqslant T$ 那么
      $G \leftarrow r_T$
    否则
      $G \leftarrow r_{t+1} + \gamma \sum_a \pi(a|s_{t+1}) Q(s_{t+1}, a)$
    对于 $k = \min(t, T-1)$递减到$\tau + 1$ 执行
      $G \leftarrow r_k + \gamma \sum_{a \neq a_k} \pi(a|s_k) Q(s_k, a) + \gamma \pi(a_k|s_k) G$
    $Q(s_\tau, a_\tau) \leftarrow Q(s_\tau, a_\tau) + \alpha[G - Q(s_\tau, a_\tau)]$
  如果 $\tau = T - 1$ 那么
    break

N-Q(σ) 结合采样的Sarsa和展开的树回溯，在每个状态由参数σ决定是采样还是展开，将两种线性情况组合起来：

$$G_{t:h} \doteq r_{t+1} + \gamma[\sigma_{t+1}\rho_{t+1} + (1 - \sigma_{t+1})\pi(a_{t+1}|s_{t+1})][G_{t+1:h} - Q_{h-1}(s_{t+1}, a_{t+1})] + \gamma \bar{v}_{h-1}(s_{t+1})$$

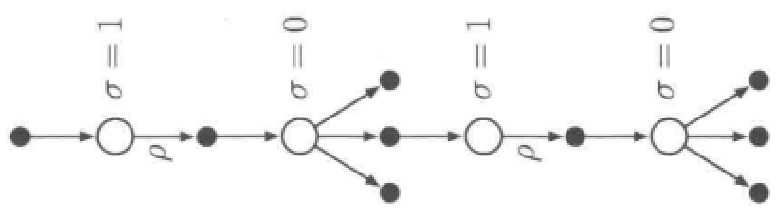


图 17: Q(sigma)回溯图

返回目录

算法 24：n-Q(σ)-off-policy

输入：$b(a|s) > 0$

参数：步长$\alpha \in (0,1]$，探索率$\epsilon > 0$，步数$n \in N_+$

初始化：$\forall s \in S, a \in A$，任意初始化$Q(s,a)$，初始化π

对于 每一幕 执行

初始化s_0为非终止状态

根据$b(\cdot|s_0)$选取a_0

$T \leftarrow \infty$

对于 $t = 0, 1, 2, \ldots$ 执行

如果 $t < T$ 那么

执行a_t，观察r_{t+1}, s_{t+1}

如果 s_{t+1}是终止状态 那么

$T \leftarrow t+1$

否则

根据$b(\cdot|s_{t+1})$选取a_{t+1}

选择σ_{t+1} ▷ 指示是采样还是展开

$\rho_{t+1} \leftarrow \frac{\pi(a_{t+1}|s_{t+1})}{b(a_{t+1}|s_{t+1})}$ ▷ 重要性采样比率

$\tau \leftarrow t - n + 1$ ▷ τ是正在更新的状态的时间

如果 $\tau \geqslant 0$ 那么

$G \leftarrow 0$

对于 $k = \min(t, T-1)$递减到$\tau + 1$ 执行

如果 $k = T$ 那么

$G \leftarrow r_T$

否则

$\bar{V} \leftarrow \sum_a \pi(a|s_k) Q(s_k, a)$ ▷ 计算期望状态值

$G \leftarrow r_k + \gamma[\sigma_k \rho_k + (1-\sigma_k)\pi(a_k|s_k)][G - Q(s_k, a_k)] + \gamma \bar{v}$

$Q(S_\tau, A_\tau) \leftarrow Q(s_\tau, a_\tau) + \alpha[G - Q(s_\tau, a_\tau)]$

如果 $\tau = T - 1$ 那么

break

返回正文5.6。

12.2.4 模型和规划

模型

- 分布模型：生成所有可能的结果的描述与概率分布。
- 样本模型：从所有可能中按概率分布采样一个确定结果。可由分布模型生成，一般更容易获得。

规划

- 规划：以环境模型为输入，生成或改进与其交互的策略。
- 规划空间：
  - 状态空间规划：在状态空间搜索最优策略。
  - 方案空间规划：进化算法、偏序规划。
- 规划时间：
  - 后台规划：从环境模型生成模拟经验，改进策略或值函数。
  - 决策时规划：使用模拟经验为状态选择动作。

统一的状态空间规划算法 通过仿真经验的回溯操作计算值函数，进而改进策略。

$$\text{模型} \Longrightarrow \text{模拟经验} \overset{\text{回溯}}{\Longrightarrow} \text{值函数} \Longrightarrow \text{策略}$$

12.2.5 *Dyna-Q*

学习和规划由相同算法完成，真实经验用于学习，模拟经验用于规划。

算法 25：Dyna-Q

1: 初始化：$\forall s \in S, a \in A(S)$，初始化$Q(s,a)$和$Model(s,a)$
2: 循环
3:     $s \leftarrow$ 当前（非终止）状态 ▷ 学习
4:     基于(s,Q)选取a，执行后观察r, s' ▷ 可用ϵ-greedy策略
5:     $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_a Q(s',a) - Q(s,a)]$ ▷ 直接强化学习更新
6:     $Model(s,a) \leftarrow r, s'$
7:     对于 $i = 1, \ldots, n$ 执行 ▷ 规划

算法 25：Dyna-Q

随机选择已观测过的s和其下采取过的a
$r, s' \leftarrow Model(s, a)$ ▷ 从模型获取预测
$Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_a Q(s',a) - Q(s,a)]$ ▷ 规划更新

12.2.6 改进方法

鼓励长期未出现动作，模型可能不正确，需规避在次优解收敛。

优先遍历 集中更新有收益动作，而非均匀采样。

关联前导动作和前导状态，在后续动作有收益时先更新前导动作价值，进行有效更新。按照价值改变多少对状态-动作对进行优先级排序，并由后至前反向传播出高影响序列。

算法 26：确定性环境下的优先遍历

初始化：$\forall s \in S, a \in A(s)$，初始化$Q(s,a), Model(s,a)$，初始化优先级队列$PQueue = NULL$
循环
  $s \leftarrow$当前（非终止）状态
  基于(s,q)选取a，执行后观察r, s' ▷ 可用ϵ-greedy策略
  $Model(s,a) \leftarrow r, s'$
  $P \leftarrow |r + \gamma \max_a Q(s',a) - Q(s,a)|$ ▷ 优先级
  如果 $P > 0$ 那么
    将(s,a)以优先级P插入$PQueue$
  对于 $i = 1, \dots, n$ 执行
    如果 $PQueue = NULL$ 那么
      break
    $(s,a) \leftarrow PQueue(0)$ ▷ 最高优先级
    $r, s' \leftarrow Model(s,a)$ ▷ 从模型获取预测
    $Q(s,a) \leftarrow Q(s,a) + \alpha[r + \gamma \max_a Q(s',a) - Q(s,a)]$ ▷ 规划更新
    对于 每个可达到s的$(\bar{s}, \bar{a})$ 执行 ▷ 反向传播更新
      $\bar{r}, \bar{s}' \leftarrow Model(\bar{s}, \bar{a})$
      如果 $\bar{s}' = s$ 那么

返回目录

算法 26：确定性环境下的优先遍历

18: $P \leftarrow |\bar{r} + \gamma \max_a Q(s,a) - Q(\bar{s},\bar{a})|$
19: 如果 $P > 0$ 那么
20: 将$(\bar{s},\bar{a})$以优先级P插入PQueue

ON-POLICY轨迹采样 借助模拟生成经验回溯更新，能跳过无关状态，获得最优部分策略。实时动态规划（RTDP）是其异步值迭代版本，可在较少访问频率下找到最优策略，并且产生轨迹所用的策略也会接近最优策略。

启发式搜索 聚焦于当前状态。

预演算法 作为MC的特例，通过平均多个起始于可能动作并遵循给定策略的模拟轨迹的回报来估计行为值。蒙特卡洛树搜索（MCTS）通过累积蒙特卡洛值估计来不断优化模拟轨迹的收益。

返回正文21。

12.3 值函数近似

核函数

- 基于记忆样本，使用RBF核，存储样本状态。核函数$k(s,s')$可表示为特征向量$x(s)$的内积，每个特征对应一个样本状态：

$$k(s,s') = x(s)^T x(s')$$

- 非参数化，不需要学习参数。
- 避免高维计算，高效处理特征。
- 线性参数化方法皆可重塑为核函数，相同训练数据下会得到近似结果。

返回正文7.1。

12.4 数学基础

概率空间(Ω, F, P)

- 性质
  - 非负性：$\forall A \in F, P(A) \geqslant 0$。
  - 规范性：$P(\Omega) = 1$。
  - 可列可加性：若$A_1, A_2, \ldots$互斥，则$P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$。
- 运算
  - 补集：$P(A^c) = 1 - P(A)$。
  - 交集：$P(A \cap B) = P(A) + P(B) - P(A \cup B)$。

随机变量

- 离散型
  - 概率质量函数(PMF)：$P(X = x) = p(x), \sum_x p(x) = 1$。
  - 期望：$E[X] = \sum_x xp(x)$。
- 连续型
  - 概率密度函数(PDF)：$f(x) \geqslant 0, \int_{-\infty}^{\infty} f(x)dx = 1$。
  - 期望：$E[X] = \int_{-\infty}^{\infty} xf(x)dx$。
- 方差：$Var(X) = E[(X - E[X])^2]$。

条件概率与独立性

- 条件概率：$P(B|A) = \frac{P(A\cap B)}{P(A)}, P(A) > 0$。
- 全概率公式：$P(B) = \sum_{A \subseteq F} P(B|A)P(A)$。
- 贝叶斯定理：$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$。
- 独立性： A, B独立 $\Longleftrightarrow P(A \cap B) = P(A)P(B)$。
- 条件独立：$P(A, B|C) = P(A|C)P(B|C)$。

大数定律与中心极限定理

- 弱大数律：$\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{p} E[X]$。
- 强大数律：$\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{a.s.} E[X]$。
- 中心极限定理：$X_1, X_2, \ldots$独立同分布，均值为μ，方差为$\sigma^2 < \infty$，则$\frac{1}{\sqrt{n}}\sum_{i=1}^{n}(X_i - \mu) \xrightarrow{d} N(0, \sigma^2)$。

泛函分析

- 期望的线性：$E[aX + bY] = aE[X] + bE[Y]$。
- 协方差：$Cov(X, Y) = E[(X - \mu_X)(Y - \mu_Y)] = E[XY] - \mu_X\mu_Y$。
- 相关系数：$\rho(X, Y) = \frac{Cov(X,Y)}{\sigma_X\sigma_Y}$。

状态分布

- 均匀分布（各状态同等重要）：$J(\omega) = \frac{1}{|S|}\sum_{s\in S}[v_\pi(s) - \hat{v}(s, \omega)]^2$。
- 平稳分布（马氏过程长期行为）：$J(\omega) = \sum_{s\in S} d_\pi(s)[v_\pi(s) - \hat{v}(s, \omega)]^2$。

信息论

- 熵：不确定度的度量。
  - 二值熵：$H = -p\log(p) - (1 - p)\log(1 - p)$。
  - 交叉熵：$H(P, Q) = -E_{P(x)}Q(x) = -\int P(x)\log Q(x)dx$。
- KL散度：衡量两个概率分布之间的距离。

$$D_{KL}(P||Q) = E_{x\sim P}[\log \frac{P(x)}{Q(x)}] = \int P(x)\log P(x)dx - \int P(x)\log Q(x)dx$$