练习 5.12：赛车轨道 (Racetrack) ——离策略蒙特卡洛控制解题报告

作者：JEFF

2025 年 3 月 3 日

1 引言

本报告讨论了《*Reinforcement Learning: An Introduction*》(Sutton & Barto) 第五章练习 5.12 "Racetrack" 任务的实现方法。该任务要求我们在离散化的赛车轨道上驾驶赛车，以最短时间到达终点线，同时避免冲出赛道边界。为此，我们采用了**离策略蒙特卡洛控制 (Off-policy Monte Carlo Control)** 的方法，对赛车进行策略改进与价值估计。

2 问题描述与环境设定

2.1 网格与赛车状态

题目将赛道抽象为一个离散网格，其中每个网格单元可属于以下类型：

- `EDGE` (E)：赛道边界，赛车若到达此处即视为越界。
- `TRACK_LINE` (0)：赛道区域，可安全行驶。
- `START_LINE` (S)：起始线所在区域，赛车的初始位置从此区域随机选取。
- `FINISH_LINE` (F)：终点线区域，赛车驶过即回合结束。

赛车的状态由位置和速度组成：

$$S_t = \big(\text{position} = (r, c),\ \text{velocity} = (v_x, v_y)\big).$$

其中，速度的两个分量 v_x, v_y 均限制在区间 $[0, \text{VELOCITY\_LIMIT}]$，代码中设定 VELOCITY_LIMIT 4。此外，赛车若因加速度调整导致速度分量超出该区间，则需将其截断（clamp）到合法范围内。

2.2 动作与随机噪声

每个时间步，赛车可以执行 9 种动作 (a_x, a_y)，其中

$$a_x, a_y \in \{-1, 0, 1\}.$$

该动作表示对当前速度在水平方向和垂直方向各增加 $-1, 0, +1$。根据题目描述，为增加难度，还需要在每个时间步以 0.1 的概率将加速度置为 $(0, 0)$，即本来想加速也可能失败，从而使赛车不发生速度变化。

2.3 奖励设计

为了鼓励赛车尽快到达终点，并惩罚越界、拖延时间等行为，本实验中设定：

$$r_t = \begin{cases} \text{REWARD\_MOVE} = -1, & \text{若赛车在赛道内正常移动;} \\ \text{REWARD\_FINISH} = 0, & \text{若赛车穿过终点线;} \\ \text{REWARD\_OUT\_OF\_TRACK} = -100, & \text{若赛车越界;} \\ \text{REWARD\_TIMEOUT} = -200, & \text{若回合超过最大步数限制;} \end{cases}$$

其中回合超时指超过 MAX_STEPS_PER_EPISODE = 10000 步，此时会强制结束回合。

3 离策略蒙特卡洛控制原理

3.1 目标策略与行为策略

在离策略（Off-policy）控制中，我们将**目标策略** π 与**行为策略** b 区分开来：

- **目标策略** π：我们想要学习和评估的策略，通常是一个确定性贪心策略，即对动作价值函数 $Q(s, a)$ 取最大值的动作。
- **行为策略** b：实际在环境中产生数据的策略，需要保证对所有可能动作都有非零概率（覆盖性），常见做法是纯随机策略或 ϵ-soft 策略。

在本练习中，π 为" 对当前 Q 贪心" 的确定性策略，而 b 选为纯随机（9 种动作等概率），使得所有动作都能被探索到。这种设置确保了足够的探索，但也可能导致训练效率较低，需要更多回合才能收敛。

3.2 加权重要性采样（Weighted Importance Sampling）

离策略蒙特卡洛方法依赖于重要性采样（*Importance Sampling*）来对目标策略的价值进行无偏估计。本实现中，我们使用了加权重要性采样方法，以降低估计的方差。

设一条经历为

$$S_0, A_0, R_1, S_1, A_1, R_2, \ldots, S_{T-1}, A_{T-1}, R_T, S_T,$$

其在行为策略 b 下发生的概率为

$$P_b(\tau) = \prod_{t=0}^{T-1} b(A_t \mid S_t),$$

而在目标策略 π 下发生的概率则为

$$P_\pi(\tau) = \prod_{t=0}^{T-1} \pi(A_t \mid S_t).$$

对于该条轨迹，我们定义重要性采样比率为

$$W_t \;=\; \prod_{k=0}^{t-1} \frac{\pi(A_k \mid S_k)}{b(A_k \mid S_k)}.$$

在本实验中，b 是 9 种动作均匀随机，即 $b(a \mid s) = \frac{1}{9}$，而 π 对唯一最优动作的概率为 1，其余为 0，则

$$\frac{\pi(A_k \mid S_k)}{b(A_k \mid S_k)} = \begin{cases} 9, & \text{若} A_k = \arg\max_a Q(S_k, a), \\ 0, & \text{否则}. \end{cases}$$

这导致在回溯时，一旦发现行为策略所选动作与目标策略不符，后续权重即为 0，可直接 break 结束回溯。如代码中所实现的：

```
if at != best_a:
    break
W *= 9
```

3.3 价值函数更新

在蒙特卡洛框架下，我们在每条轨迹结束后计算回报。由于本问题中设置折扣因子 $\gamma = 1.0$，回报公式简化为：

$$G_t = R_{t+1} + R_{t+2} + \cdots + R_T.$$

选择 $\gamma = 1.0$ 的原因是：

- 赛车轨道是一个有明确终点的情节式任务，不存在无限累积的问题。
- 我们希望最小化到达终点的总步数，每步都有 REWARD_MOVE $= -1$ 的惩罚，因此总回报实际上就是负的步数（加上可能的越界惩罚）。

- 轨道上的每一步决策都同等重要，不应偏向于短期或长期奖励，$\gamma = 1.0$ 保证了不对未来奖励进行折扣。

然后，使用加权重要性采样对 $Q(S_t, A_t)$ 进行更新：

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \frac{W_t}{C(S_t, A_t)} \big(G_t - Q(S_t, A_t)\big),$$

其中 $C(S_t, A_t)$ 用来累积权重，以实现自适应步长：

$$C(S_t, A_t) \leftarrow C(S_t, A_t) + W_t.$$

这种更新方式使得学习率随着访问次数的增加而自动减小，有助于算法的稳定收敛。用算法描述，这部分逻辑为：

[backgroundcolor=gray!10, linewidth=1pt, roundcorner=10pt]

价值函数更新

$C[S_t, A_t] \leftarrow C[S_t, A_t] + W$
$Q[S_t, A_t] \leftarrow Q[S_t, A_t] + \frac{W}{C[S_t, A_t]} \cdot (G - Q[S_t, A_t])$
由于设置了 GAMMA $= 1.0$，回报更新简化为直接累加：
[backgroundcolor=gray!10, linewidth=1pt, roundcorner=10pt]

回报计算（当 $\gamma = 1.0$ 时）

$G \leftarrow G + R_t$ （没有折扣因子，直接累加奖励）

这意味着我们直接累加所有奖励值，不进行折扣，使得算法更关注于找到到达终点的最短路径。

我们将价值函数的初始值设为 INITIAL_VALUE $= -150$，这是一个稍微悲观的估计，有助于鼓励探索。这个值的设定也考虑到了在无折扣情况下，累积奖励的范围和可能的最差路径长度。

详细的算法和代码参见作业文件中的代码文件，在代码中详细描述了算法思路和细节。

4 实验结果与分析

4.1 两种轨道地图

我们在两种不同的轨道地图上进行了实验：

1. **track_map1**：一个相对简单的 L 形轨道，宽度适中，有一个 90 度转弯。
2. **track_map2**：一个更复杂的 S 形轨道，具有多个弯道和不同宽度的赛道段落。

4.2 价值函数的热图分析

在训练 50,000 回合后，我们得到了两个轨道的价值函数热图，从中可以观察到以下特点：

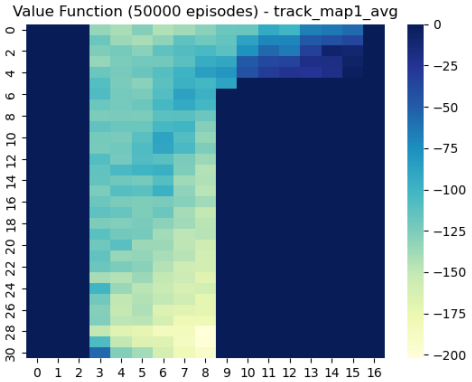

图 1: track_map1 的平均价值函数热图。颜色越浅表示价值越低，深蓝色区域代表赛道边界，而浅黄色至浅蓝色区域显示了赛车可行驶的路径价值分布。

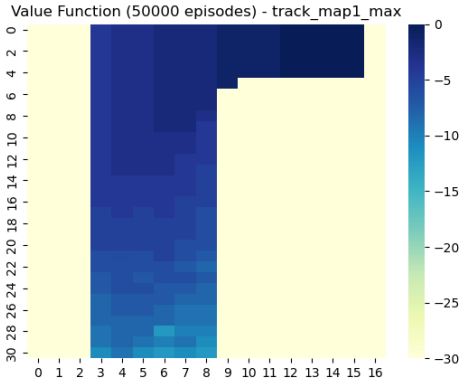

图 2: track_map1 的最大价值函数热图。与平均价值函数相比，最大价值函数显示出更为鲜明的价值对比，深蓝色区域代表最优路径。

这些热图不仅验证了算法的收敛性，也帮助我们理解赛车如何在不同位置选择最优行动。特别是，我们可以观察到：

- 在直道上，价值函数较为平滑，赛车倾向于保持高速。
- 在接近弯道处，价值函数有明显变化，暗示赛车需要提前减速。
- 在终点线附近，价值函数显著提高，反映了成功完成任务的奖励期望。

4.3 轨迹可视化与分析

通过轨迹可视化，我们可以直观地看到学习到的策略在实际执行时的表现。从生成的轨迹图中，我们观察到：

通过分析这些轨迹，我们发现：

1. **track_map1**：
    - 赛车能够根据起点位置选择不同的路线，体现出策略的灵活性。
    - 在接近转弯处时，赛车会提前调整路线，避免碰撞边界。
    - 当有多条可行路径时，学习到的策略倾向于选择最短或风险最低的路径。
2. **track_map2**：

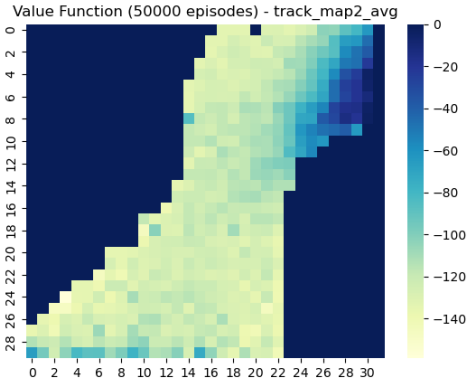

图 3: track_map2 的平均价值函数热图。在这个 S 形赛道上，可以观察到价值分布随着赛道形状的变化而变化，从浅黄色（低价值）逐渐过渡到深蓝色（高价值）区域。

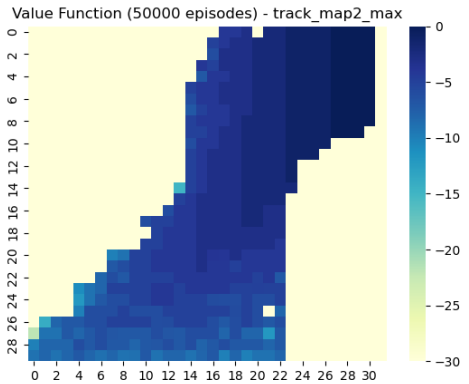

图 4: track_map2 的最大价值函数热图。S 形赛道的最优路径清晰可见，深蓝色区域代表最优策略的价值较高的状态。

- 在较大的赛道上，策略表现出更明显的位置自适应性，能够规划出从不同起点到终点的有效路径。
- 图 7 展示了直线策略，而图 8 展示了对角线策略，表明算法能够发现多种到达终点的有效方式。
- 学习到的策略能够利用赛道特性，在安全的前提下最大化移动效率。

通过比较不同起始位置的轨迹，我们可以确认算法成功学习到了适应不同初始条件的灵活策略，而不仅仅是单一的固定路线。这一特性对于实际应用中的鲁棒性至关重要。

4.4 算法收敛性与参数敏感度

通过实验我们发现，离策略蒙特卡洛控制算法在本问题上表现出良好的收敛性，但也观察到一些值得注意的特点：

1. **训练周期**：虽然设置了 50,000 回合，但实际上在约 30,000 回合后，价值函数和策略已基本稳定，表明算法在此任务上收敛较快。

2. **参数敏感度**：

   - **折扣因子 (GAMMA)**：设为 1.0，这是一个关键选择，它使得回报直接等同于累积奖励和，适合于有明确终止状态的赛车问题。由于每步移动奖励为-1，这实际上使得算法直接优化到达终点的最短路径。如果设置较小的 GAMMA

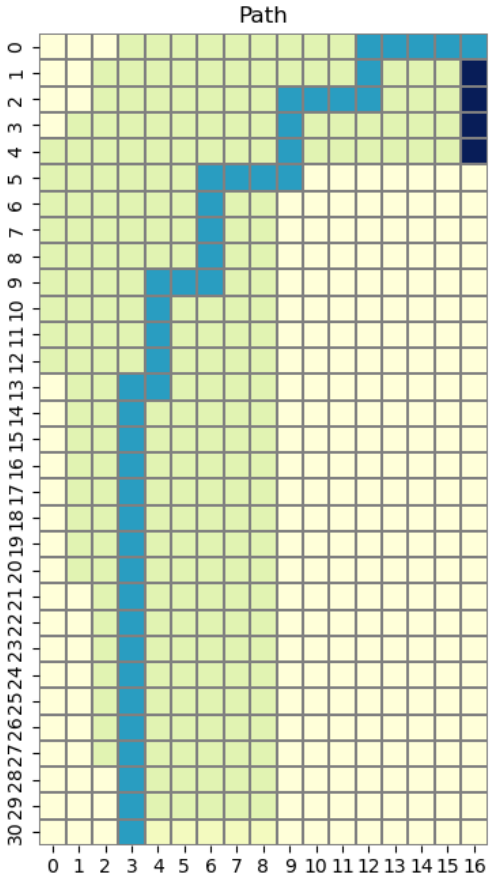

图 5: track_map1 的一条轨迹示例（右侧起点）。蓝色路径显示了赛车从右侧驶向左上方终点的路线，体现了算法学习到的策略在处理直角弯道时的表现。

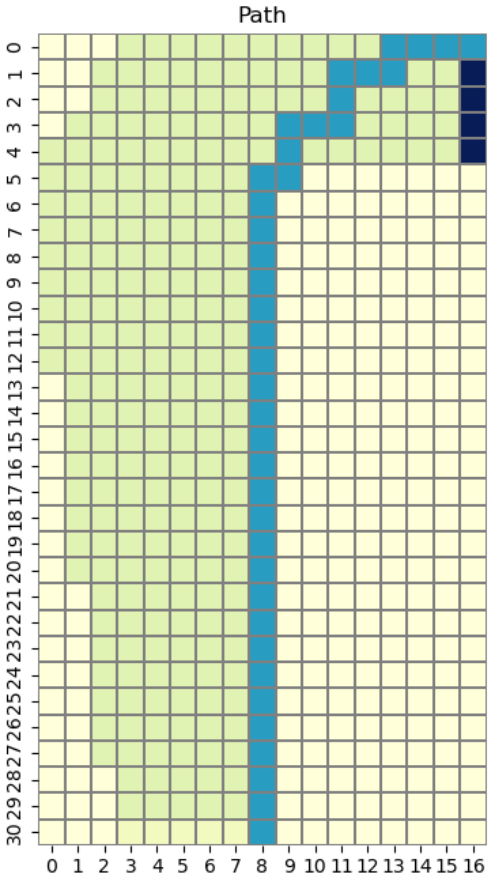

图 6: track_map1 的另一条轨迹示例（右侧另一起点）。与图 5 相比，这条轨迹显示了算法在不同起点位置下的适应性，路线更为平缓。

（如 0.9），算法会更关注近期奖励，可能导致选择次优但前期回报较高的路径。

- **初始值 (INITIAL_VALUE)**：设为 -150 的悲观初始值有助于鼓励探索，若设置过高可能导致过度乐观，影响收敛速度。
- **奖励设计**：越界惩罚 (-100) 和超时惩罚 (-200) 的比例关系影响了策略的风险偏好，当前设置使赛车既不过分冒险也不过分保守。

3. **动作空间影响**：限制加速度在 $\{-1, 0, 1\}$ 范围内是合理的，但这也意味着赛车需要多个步骤才能显著改变速度，从而延长了策略学习的时间。

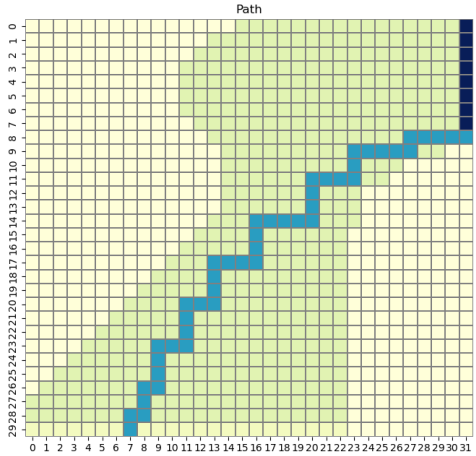

图 7: track_map2 的一条轨迹示例（下方起点）。在这个大型赛道上，蓝色路径显示了赛车如何沿着中间列直线前进，并在顶部转向右侧的终点。

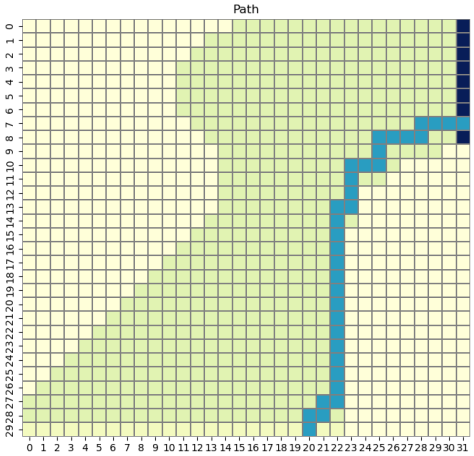

图 8: track_map2 的另一条轨迹示例（下方另一起点）。这条对角线轨迹展示了算法在不同起点下能够学习到更为灵活的策略，通过斜向移动更快地到达终点。

5 结论

通过离策略蒙特卡洛控制算法，我们成功解决了赛车轨道问题，使赛车能够学习到在避免碰撞的前提下，高效地从起点到达终点的驾驶策略。实验结果表明，该方法在不同复杂度的赛道上都能收敛到合理的策略，并且通过价值函数和轨迹的可视化，我们能够直观地理解和验证学习成果。

虽然纯随机的行为策略在探索效率上不尽理想，但结合加权重要性采样和逐格检查等技术，我们的实现仍然取得了令人满意的结果。这也证明了离策略学习在复杂控制任务中的有效性和灵活性。

6 参考文献

- Sutton, R. S., & Barto, A. G. (2018). *Reinforcement Learning: An Introduction* (2nd Edition).
- Precup, D., Sutton, R. S., & Singh, S. P. (2000). Eligibility Traces for Off-Policy Policy Evaluation. *ICML*.
- Rubinstein, R. Y., & Kroese, D. P. (2016). *Simulation and the Monte Carlo Method* (3rd Edition).
- 本课程作业相关说明与提示。

7 附录：Blackjack 问题的蒙特卡洛方法实现

作为额外的工作，我复现了书中关于 Monte Carlo 方法对 Blackjack 游戏的解决方案。Blackjack（二十一点）是一个经典的强化学习测试问题，它具有状态空间大、随机性强的特点，非常适合用蒙特卡洛方法求解。

7.1 Monte Carlo 方法评估固定策略

首先，我在代码文件 blackjack.py 中实现了对一个确定策略的 Monte Carlo 评估。这个实现遵循了书中第 5 章的方法，通过多次采样来估计状态价值函数。

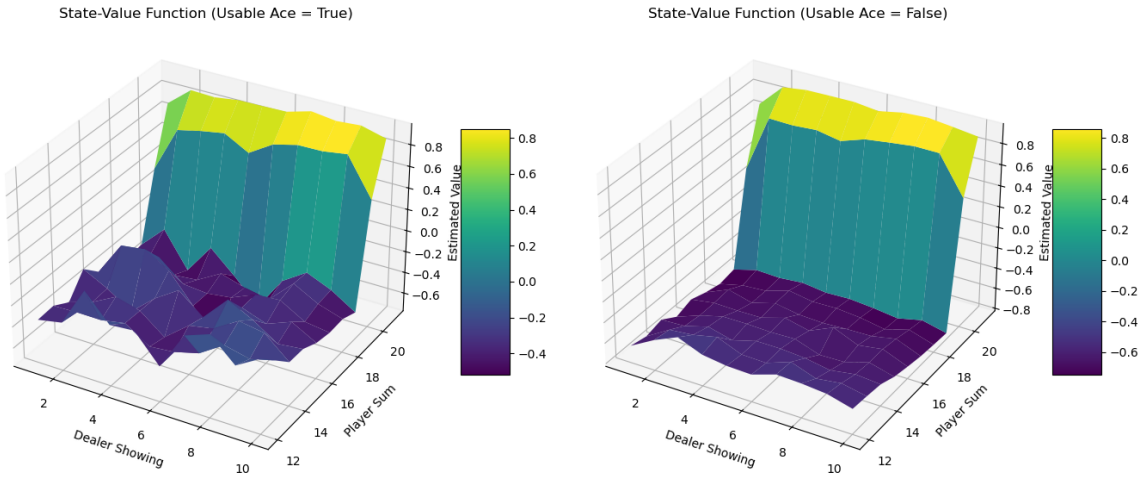

图 9: Blackjack 状态价值函数的 Monte Carlo 评估结果。左图显示玩家可以使用 A 当作 1 或 11 时的状态价值，右图显示玩家只能将 A 当作 1 时的状态价值。纵轴表示估计价值，横轴分别是庄家明牌和玩家手牌总和。

从图9中可以观察到，当玩家手牌总和较高时（接近 21 点），状态价值明显更高（黄色区域），这符合游戏的规则和直觉。同时，拥有可用的 A（可以作为 1 或 11 计算）时，状态价值整体高于无可用 A 的情况，这说明了 A 的灵活性带来的价值。

7.2 Monte Carlo ES 方法学习最优策略

进一步，我基于第 5 章示例 5.3 的思路，在 MC_ES.py 文件中实现了 Monte Carlo Exploring Starts (ES) 方法来学习最优策略。该方法具有以下特点：

- 构建了一个抽象化的 Blackjack 环境（使用无限牌堆）
- 完全依靠蒙特卡洛方法自动学习策略，不包含任何”硬编码”或”手动指定”的策略规则

- 通过充分探索起始状态和动作对（Exploring Starts）保证了对所有状态-动作对的充分访问
- 在足够多的训练回合后自然收敛到最优策略

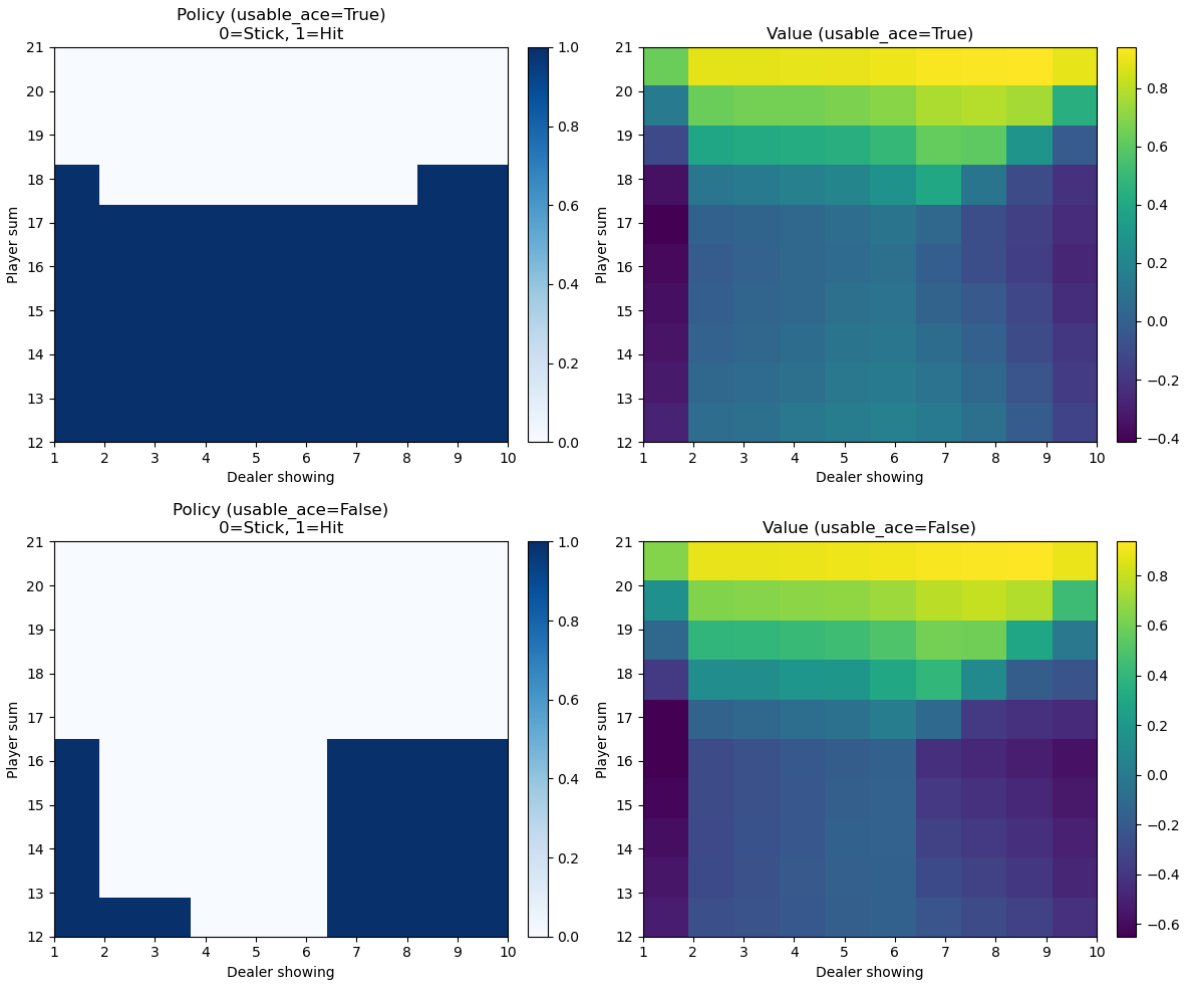

图 10: 使用 Monte Carlo ES 学习的 Blackjack 最优策略和对应的价值函数。左上: 有可用 A 时的最优策略；右上: 有可用 A 时的状态价值；左下: 无可用 A 时的最优策略；右下: 无可用 A 时的状态价值。在策略图中，深蓝色表示选择”Hit”（要牌），浅色表示选择”Stick”（停牌）。

这些学习到的策略与标准的 Blackjack 最优策略（如基本策略表）高度一致，同时也与书中的结果相符，验证了 Monte Carlo 方法在解决这类问题上的有效性。

7.3 与 Racetrack 问题的对比分析

将 Blackjack 问题与本报告主体的 Racetrack 问题对比，我们可以发现一些有趣的共同点和差异：

- **状态空间**：Blackjack 的状态空间相对简单（玩家总和、庄家明牌、是否有可用 A），而 Racetrack 问题的状态空间包含位置和速度，维度更高。
- **随机性**：两个问题都包含随机性，Blackjack 中是卡牌的随机抽取，Racetrack 中是加速度可能以 0.1 的概率被置为零。
- **终止条件**：两个问题都有明确的终止条件，Blackjack 是达到 21 点、爆牌或决定停牌，Racetrack 是到达终点或越界。
- **学习方法**：Blackjack 使用了 Monte Carlo ES（同策略）方法，而 Racetrack 使用了离策略 Monte Carlo 方法。两种方法各有优势：ES 方法在探索空间较小时更高效，而离策略方法在复杂环境中可以更好地利用已有经验。

两个问题的解决过程都展示了蒙特卡洛方法在解决马尔可夫决策过程（MDP）问题上的强大能力，特别是在不需要对环境动态模型的精确知识的情况下，通过采样和经验学习找到最优或近似最优策略。

7.4 总结与思考

Blackjack 问题的求解不仅补充了对蒙特卡洛方法的理解，也为 Racetrack 问题提供了有益的对比参照。这两个问题代表了强化学习中不同类型的挑战：

- Blackjack 是一个相对简单但具有随机性的问题，强调单步决策的风险管理。
- Racetrack 是一个确定性但状态空间更大的问题，强调序列决策和长期规划。

通过这两个问题的对比研究，我们可以更全面地理解蒙特卡洛方法在不同场景下的应用特点和优势。特别地，蒙特卡洛方法不需要对环境动态进行建模，而是直接从经验中学习，这一特性使其在复杂或难以精确建模的环境中特别有用。