强化学习经典算法解读：蒙特卡洛方法

szq (根据 PPT 内容整理)

2025 年 7 月 27 日

目录

1 蒙特卡洛（Monte Carlo）方法介绍

根据课程的规划，我们将逐步学习以下几种基于蒙特卡洛（MC）的强化学习算法：

1. 启发性例子 (Motivating example)
2. 最简单的基于 MC 的强化学习算法 (MC Basic)
3. 更高效地利用数据 (MC Exploring Starts)
4. 无需“探索性开端”的 MC 算法 (MC ε-Greedy)

本文档将按照此顺序，逐步深入解读这些算法。

1.1 核心思想：从“需要模型”到“模型无关”

在强化学习中，我们的目标是评估一个策略的好坏，通常通过计算其价值函数。动作价值函数 $q_\pi(s,a)$ 有两种主要的表达形式。

- **表达式一（需要模型）：**依赖于环境的动态模型 $p(s'|s,a)$ 和奖励模型 $p(r|s,a)$。

$$q_{\pi_k}(s,a) = \sum_r p(r|s,a)r + \gamma \sum_{s'} p(s'|s,a)v_{\pi_k}(s')$$

- **表达式二（无需模型）：**将动作价值定义为回报（Return）的期望。

$$q_{\pi_k}(s,a) = \mathbb{E}[G_t|S_t = s, A_t = a]$$

实现模型无关（Model-Free）RL 的核心思想： 我们可以利用表达式二，直接基于与环境交互产生的**数据**（样本或经验），来估计动作价值函数 $q_{\pi_k}(s,a)$。

1.2 动作价值的蒙特卡洛估计流程

蒙特卡洛方法正是利用大数定律，通过采样来估计期望值。

- 从一个指定的“状态-动作”对 (s,a) 出发，遵循当前策略 π_k 生成一个完整的幕（episode）。
- 这个幕的回报 $g(s,a)$ 就是总回报 G_t 的一个**样本**。
- 通过生成 N 个幕并取回报的平均值，我们可以近似期望值：

$$q_{\pi_k}(s,a) = \mathbb{E}[G_t|S_t = s, A_t = a] \approx \frac{1}{N}\sum_{i=1}^{N} g^{(i)}(s,a)$$

基本思想：当模型未知时，我们可以用**数据（data）**进行估计。

2 最简单的 MC 强化学习算法 (MC Basic)

2.1 算法描述

给定初始策略 π_0，在第 k 次迭代中，算法包括两个步骤：

Step 1: **策略评估 (Policy Evaluation)**: 对每个 (s,a)，从其出发运行足够多的幕，用回报的平均值来近似 $q_{\pi_k}(s,a)$。

Step 2: **策略改进 (Policy Improvement)**: 使用贪心策略来改进策略，$\pi_{k+1}(s) = \text{argmax}_a\, q_{\pi_k}(s,a)$。

2.2 与策略迭代（Policy Iteration）的对比

MC Basic 算法的流程与**策略迭代算法**几乎完全相同，区别在于**策略评估**：MC Basic 通过**采样求平均**来估计 $q_{\pi_k}(s,a)$，而非求解贝尔曼方程。

3 提高数据利用效率：First-Visit 与 Every-Visit MC

MC Basic 算法为了评估 $q(s,a)$，需要从该 (s,a) 出发生成大量幕，数据效率低下。一个幕实际上可以用来更新其中所有出现过的状态-动作对。

3.1 数据利用问题与改进思路

考虑一个幕：$s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_4} s_1 \xrightarrow{a_2} s_2 \xrightarrow{a_3} s_5 \to \dots$

- **MC Basic 的做法 (Initial-visit method)**: 只用这个幕来评估 $q_\pi(s_1,a_2)$。
- **缺点**: 浪费了幕中包含的 $(s_2,a_4),(s_2,a_3)$ 等其他信息。

3.2 更高效的数据利用方法

当一个 (s,a) 在一个幕中被访问时，我们可以利用其后续的回报来更新 $q(s,a)$。当 (s,a) 被多次访问时，有两种主流方法：

- **首次访问蒙特卡洛法 (First-visit MC)**: 仅使用该幕中 (s,a) **第一次**出现时所得到的回报来更新。
- **每次访问蒙特卡洛法 (Every-visit MC)**: 使用该幕中 (s,a) **每一次**出现时所得到的回报来更新。

4 MC Exploring Starts 算法

4.1 探索性开端 (Exploring Starts) 的概念与挑战

问题：为什么需要“探索性开端”？

- **理论上的必要性**: 为了找到最优动作，必须对每个状态下的**所有**动作的价值都有准确估计。如果某个动作从未被探索，就可能错过最优解。

为此，我们引入**探索性开端的假设**：假设任意 (s, a) 都有非零概率成为幕的起点。

实践中的挑战: 这个假设在物理世界等很多应用中难以满足，造成了理论与实践的鸿沟。

5 无需探索性开端的 MC 算法：MC ε-Greedy

为了移除“探索性开端”的强假设，我们引入“软策略”的概念。

5.1 软策略 (Soft Policies) 的引入

- **定义**: 一个策略 π 如果对任意状态下的任意动作，选择的概率都为正（即 $\pi(a|s) > 0$ for all s, a），那么它就被称为**软策略 (soft policy)**。
- **引入原因**: 使用软策略，我们只需生成足够长的幕，就有机会访问到每一个状态-动作对。这样就不再需要强制让幕从每一个状态-动作对开始。

5.2 ε-贪心策略 (ε-Greedy Policies)

我们将使用的具体软策略是 **ε-贪心策略**。

$$\pi(a|s) = \begin{cases} 1 - \varepsilon + \frac{\varepsilon}{|\mathcal{A}(s)|} & \text{对于贪心动作 (greedy action)} \\ \frac{\varepsilon}{|\mathcal{A}(s)|} & \text{对于其他} |\mathcal{A}(s)| - 1 \text{ 个动作} \end{cases}$$

其中 $\varepsilon \in [0, 1]$。它在**探索 (exploration)** 和**利用 (exploitation)** 之间取得了平衡。

5.3 MC ε-Greedy 算法描述

我们将 ε-贪心策略嵌入到算法中，通过修改**策略改进**步骤。

原始的策略改进（确定性） $\pi_{k+1}(a|s) = 1$ if $a = \text{argmax}_{a'} q_{\pi_k}(s, a')$, and 0 otherwise.

新的策略改进（ε-Greedy） 新的策略 π_{k+1} 对于当前的价值函数 q_{π_k} 是 ε-贪心的：

$$\pi_{k+1}(a|s) = \begin{cases} 1-\varepsilon+\frac{\varepsilon}{|\mathcal{A}(s)|}, & \text{if } a = a_k^* \\ \frac{\varepsilon}{|\mathcal{A}(s)|}, & \text{if } a \neq a_k^* \end{cases} \quad \text{其中} a_k^* = \underset{a}{\operatorname{argmax}}\, q_{\pi_k}(s,a)$$

算法总结

- **MC ε-Greedy 算法**与 MC Exploring Starts 算法基本相同，区别在于策略改进步骤使用了 ε-贪心策略。
- 它**不再需要探索性开端**的假设，但仍通过软策略的随机性来保证对所有状态-动作对的持续探索。