



Reinforcement Learning: An Introduction

强化学习导论第二版习题解答

作者：吕昀璘

组织：UESTC

时间：March 24, 2020



Victory won't come to us unless we go to it. — M. Moore

目录

1	介绍	1
1.1	强化学习	1
1.2	例子	1
1.3	强化学习的要素	1
1.4	局限和范围	1
1.5	拓展例子：井字游戏	1
1.6	总结	2
1.7	强化学习的早期历史	2
	第一部分 表格解决方法	3
2	多臂赌博机	4
2.1	k 臂赌博机问题	4
2.2	动作值方法	4
2.3	10 臂试验	4
2.4	渐增实现	5
2.5	非平稳问题	5
2.6	乐观初始值	5
2.7	置信上限动作选择	6
2.8	梯度赌博机算法	6
2.9	关联搜索（上下文赌博机）	7
2.10	总结	7
3	有限马尔可夫决策过程	8
3.1	Agent-环境接口	8
3.2	目标和奖励	9
3.3	回报和 episode	9

第一章 介绍

1.1 强化学习

1.2 例子

1.3 强化学习的要素

1.4 局限和范围

1.5 拓展例子：井字游戏

🔥 **练习 1.1** : *Self-Play* 假设上面描述的强化学习算法不是与随机对手对战，而是与自身对战，双方都在学习。你认为在这种情况下会发生什么？它会学习一个不同的策略来选择动作吗？ □

解 当与自身对战时：

- 比起固定的对手，与自身对战将学习不同的策略，因为在这种情况下，对手也会有所变化。
- 由于对手也在不断变化，因此可能无法学习最佳策略。
- 可能卡在循环中。因为与自身博弈，自身策略和对手策略都在优化。
- 策略可以保持静态，因为就平均而言，通过每次迭代它们处于平局。

🔥 **练习 1.2** : *Symmetries* 许多井字游戏的位置看起来不同，但由于对称性实际上是一样的。我们如何修改上述学习过程来利用这一点？这种变化会在哪些方面改善学习过程？现在再想想。假设对手没有利用对称性。那样的话，我们应该吗？那么，对称相等的位置必然具有相同的值，这是真的吗？ □

解 我们可以将状态标记为对称的唯一状态，这样我们的搜索空间更小，这样我们就可以更好地估计最佳玩法。

如果我们面对的对手在比赛时没有考虑对称性，那么我们也不应将状态标记为相同。因为对手也是环境的一部分，而环境给出的这些状态并不一致。

🔥 **练习 1.3** : *Greedy Play* 假设强化学习玩家是贪婪的，也就是说，他总是做出让他达到最佳位置的移动。它会比不贪婪的玩家学得更好或更差吗？可能会发生什么问题？ □

解 贪婪的玩家不会探索，因此通常会比非贪婪的玩家表现更差。

如果贪婪的玩家对状态的价值有一个完美的估计，那它将更好。


🔥 **练习 1.4** : *Learning from Exploration* 假设学习更新发生在所有移动之后，包括探索移动。如果随时间逐步减小步长参数（而不是探索的趋势），则状态值将收敛到一组不同的概率。当我们从或者不从探索性动作中学习时对应的两组概率是什么（概念上）？假设我们

确实在继续进行探索移动，那么哪一组概率可能更好学习？哪个会带来更多胜利？ □

解 如果我们不从探索性动作中学习，那么所学到的状态概率将是随机的，因为我们不会更新在给定状态下采取给定动作时会发生的情况。

如果我们从探索性动作中进行学习，那么我们的极限概率应该是状态和动作选择的期望分布。

显然，由于玩家更好地理解正在玩的“游戏”，因此对概率密度的更全面的了解应该会带来更好的玩法。

 **练习 1.5: Other Improvements** 你还能想出其他方法来提高强化学习玩家吗？你能想出更好的办法来解决所提出的井字游戏问题吗？ □

解 一种可能的方法是持有已保存的玩法库。例如，当在一组已知状态中，始终执行库中所对应的移动。这有点像国际象棋游戏，其中有很多“开场”位置被专家玩家认为是好的。这可以加快整个学习过程，或至少改善强化学习玩家的初期发挥。

由于井字游戏是如此简单，我们可以使用递归解决此问题，并计算所有可能的对手移动，并在每一步中选择能最大化我们获胜机会的移动。

1.6 总结

1.7 强化学习的早期历史


第一部分

表格解决方法

第二章 多臂赌博机

2.1 k 臂赌博机问题


2.2 动作值方法

 **练习 2.1** 在 ε -greedy 动作选择中，对于两个动作和 $\varepsilon = 0.5$ 的情况，选择贪婪动作的概率是多少？ □

解 设动作集合中总共具有 n 个动作。在 ε -greedy 方法中，agent 有 ε 的概率机会从动作集合中随机选择，有 $1 - \varepsilon$ 的概率机会选择贪婪动作。已知 $\varepsilon = 0.5$ 和 $n = 2$ ，那么选择贪婪动作的概率为：

$$\frac{1}{n} \times \varepsilon + (1 - \varepsilon) = \frac{1}{2} \times 0.5 + (1 - 0.5) = 0.75$$


2.3 10 臂试验

 **练习 2.2** : *Bandit example* 考虑一个具有 $k = 4$ 个动作的 k 臂赌博机问题，分别表示为 1、2、3 和 4。考虑对该问题应用赌博机算法，该算法使用 ε -greedy 动作选择，样本平均动作值估计和对于所有 a ， $Q_1(a) = 0$ 。假设动作和奖励的初始序列为 $A_1 = 1, R_1 = -1, A_2 = 2, R_2 = 1, A_3 = 2, R_3 = -2, A_4 = 2, R_4 = 2, A_5 = 3, R_5 = 0$ 。在某些时间步上， ε 情况可能已经发生，导致随机选择一个动作。这肯定发生在哪些时间步？在哪些时间步这可能已经发生？ □

解 根据题意列出每一步的动作值，已选择的动作，和选择该动作的原因如下：

时间步	动作值	已选择的动作	选择原因	原因说明				
1	<table><tr><td>0</td><td>0</td><td>0</td><td>0</td></tr></table>	0	0	0	0	1	贪婪或随机	所有动作值相等，都为 0
0	0	0	0					
2	<table><tr><td>-1</td><td>0</td><td>0</td><td>0</td></tr></table>	-1	0	0	0	2	贪婪或随机	2、3、4 的动作值相等
-1	0	0	0					
3	<table><tr><td>-1</td><td>1</td><td>0</td><td>0</td></tr></table>	-1	1	0	0	2	贪婪	2 的动作值最大
-1	1	0	0					
4	<table><tr><td>-1</td><td>-1/2</td><td>0</td><td>0</td></tr></table>	-1	-1/2	0	0	2	随机	2 的动作值最小
-1	-1/2	0	0					
5	<table><tr><td>-1</td><td>1/3</td><td>0</td><td>0</td></tr></table>	-1	1/3	0	0	3	随机	2 的动作值最大
-1	1/3	0	0					


由表可知， ε 情况肯定在 A_4 和 A_5 发生，可能在 A_1 和 A_2 发生。

 **练习 2.3** 在图 2.2 所示的比较中，就累积奖励和选择最佳动作的概率而言，哪种方法在长期内表现最好？它会好多少？量化地表达你的答案。 □

解 $\varepsilon = 0.01$ 将有更好的表现，因为在两种情况下，当 $t \rightarrow \infty$ 时，我们都有 $Q_t \rightarrow q_*$ 。因此，在这种情况下，总奖励和选择最佳行动的可能性将比 $\varepsilon = 0.1$ 大 10 倍。


2.4 渐增实现

2.5 非平稳问题

 **练习 2.4** 如果步长参数 α_n 不恒定，则估计值 Q_n 是先前接收的奖励的加权平均值，其权重与 (2.6) 给出的权重不同。就步长参数的序列而言，对于一般情况，类似于 (2.6)，每个先前奖励的权重是多少？ □


解 推导过程与 (2.6) 类似：

$$\begin{aligned}
 Q_{n+1} &= Q_n + \alpha_n[R_n - Q_n] \\
 &= \alpha_n R_n + (1 - \alpha_n)Q_n \\
 &= \alpha_n R_n + (1 - \alpha_n)[\alpha_{n-1}R_{n-1} + (1 - \alpha_{n-1})Q_{n-1}] \\
 &= \alpha_n R_n + (1 - \alpha_n)\alpha_{n-1}R_{n-1} + (1 - \alpha_n)(1 - \alpha_{n-1})Q_{n-1} \\
 &= \alpha_n R_n + (1 - \alpha_n)\alpha_{n-1}R_{n-1} + (1 - \alpha_n)(1 - \alpha_{n-1})\alpha_{n-2}R_{n-2} + \cdots + \\
 &\quad (1 - \alpha_n)(1 - \alpha_{n-1})(1 - \alpha_{n-2}) \cdots (1 - \alpha_2)(1 - \alpha_1)Q_1 \\
 &= \left(\prod_{i=1}^n (1 - \alpha_i) \right) Q_1 + \sum_{i=1}^n \alpha_i R_i \prod_{k=i+1}^n (1 - \alpha_k)
 \end{aligned}$$

 **练习 2.5** (编程) 设计并进行实验，以证明样本平均方法对于解决非平稳问题的困难。使用 10 臂试验的修改版本，其中起初所有 $q_*(a)$ 均相等，然后进行独立的随机游走（比如在每一步对所有 $q_*(a)$ 加上均值为零且标准差为 0.01 的正态分布增量）。绘制类似图??所示的图，为使用样本平均值进行增量计算的动作值方法，和另一使用恒定步长参数 $\alpha = 0.1$ 的动作值方法去准备图。使用 $\varepsilon = 0.1$ 和更长的运行时间，比如 10,000 步。 □

解 见 `exercise-programming/exercise2.5.py`。

2.6 乐观初始值

 **练习 2.6: Mysterious Spikes** 图 2.3 中显示的结果应该是相当可靠的，因为它们是 2000 多个单独的、随机选择的 10 臂赌博机任务的平均值。那么，为什么乐观方法的曲线的早期部分会有振荡和尖峰呢？换句话说，是什么让这种方法在特定的早期步骤上表现得更好或更差呢？

解 在步骤 10 之后的某个时刻，agent 将找到最优值。然后它将贪婪地选择此值。小步长参数（相对于初始值 5 较小）意味着最优值的估计值将朝着其真实值缓慢收敛。

该真实值可能小于 5。这意味着，由于步长较小，其中一个次优动作的值仍接近 5。因此，在某个时刻，agent 又开始次优动作。

 **练习 2.7: Unbiased Constant-Step-Size Trick** 在本章的大部分内容中，我们使用样本平均来估计动作值，因为样本平均不会产生恒定步长所产生的初始偏差（参见导致 (2.6) 的

分析)。然而，样本平均并不是一个完全令人满意的解决方案，因为它们在非平稳问题上的表现可能很差。是否有可能避免固定步长的偏差，同时保持它们在非平稳问题上的优势？一种方法是使用步长为

$$\beta_n \doteq \alpha / \bar{o}_n, \quad (2.1)$$

来处理特定动作的第 n 次奖励，其中 $\alpha > 0$ 是常规的恒定步长，而 \bar{o}_n 是从 0 开始的跟踪：

$$\bar{o}_n \doteq \bar{o}_{n-1} + \alpha(1 - \bar{o}_{n-1}), \quad \text{对于 } n \geq 0, \quad \bar{o}_0 \doteq 0. \quad (2.2)$$

进行类似 (2.6) 中的分析，表明 Q 值是没有初始偏差的指数近期加权平均。

解 考虑练习 2.4 的答案。由于 $\beta_1 = 1$ ，因此对于 $k > 1$ ， Q_k 与 Q_1 无关。现在有迹象表明，随着我们往前看，剩余总和中的权重会降低。即


$$\omega_i = \beta_i \prod_{k=i+1}^n (1 - \beta_k)$$

对于固定的 n 随 i 而增加。为此，观测到

$$\frac{\omega_{i+1}}{\omega_i} = \frac{\beta_{i+1}}{\beta_i(1 - \beta_{i+1})} = \frac{1}{1 - \alpha} > 1$$

如果假定 $\alpha < 1$ 时。如果 $\alpha = 1$ ，那么对于 $\forall t$ ， $\beta_t = 1$ 。


2.7 置信上限动作选择

 **练习 2.8: UCB Spikes** 在图 2.4 中，UCB 算法在第 11 步显示了明显的峰值性能。这是为什么？请注意，为了让你的答案完全令人满意，必须解释为什么奖励在第 11 步增加，为什么在随后的步减少。提示：如果 $c = 1$ ，则尖峰不那么突出。□

解 在前 10 个步中，agent 会循环执行所有动作，因为当 $N_t(a) = 0$ 时， a 被认为是最大值。然后，在第 11 步，agent 通常会贪婪地选择。agent 将继续贪婪地选择，直到 $\ln t$ 超过 $N_t(a)$ 进行其他动作之一为止，在这种情况下，agent 将开始再次探索，从而减少了奖励。

请注意，从长远来看， $N_t = O(t)$ 和 $\ln(t)/t \rightarrow 0$ 。因此，该 agent 是“渐近贪婪的”。

2.8 梯度赌博机算法


 **练习 2.9** 证明了在两种动作情况下，soft-max 分布与统计学和人工神经网络中常用的 logistic 函数或 sigmoid 函数的分布相同。□

解 令这两个动作分别用 0 和 1 表示。现在

$$\Pr\{A_t = 1\} = \frac{e^{H_t(1)}}{e^{H_t(0)} + e^{H_t(1)}} = \frac{1}{1 + e^{-x}}$$

其中 $x = H_t(1) - H_t(0)$ 是 1 相对于 0 的相对偏好。


2.9 关联搜索（上下文赌博机）

 **练习 2.10** 假设你面对的是一个 2 臂赌博机任务，其真实动作值随时间步而随机变化。具体地说，假设对于任何时间步，动作 1 和 2 的真值分别为 0.1 和 0.2，概率为 0.5（情况 A），以及 0.9 和 0.8，概率为 0.5（情况 B）。如果你在任何一步都不能说出你面对的是哪种情况，你能达到的最好的成功期望是什么，你应该如何行动来实现它？现在假设在每一步中都被告知您面对的是情况 A 还是情况 B（尽管您仍然不知道真实的动作值）。这是一个关联搜索任务。在这个任务中，你能达到的最好的成功期望是什么？你应该如何行动才能达到这个目标？ □

解 假定奖励平稳。应该选择具有最高期望奖励的动作。对于第一种情形，动作 1 和 2 的期望值都为 0.5，即 $0.5 \times (0.1 + 0.9) = 0.5$ ， $0.5 \times (0.2 + 0.8) = 0.5$ 。因此这两个动作可以随机进行选择。

针对第二种情形，应该对每种颜色分别运行正常的赌博机方法。在每种情况下确定最佳动作的期望奖励为 0.55，即 $0.5 \times 0.2 + 0.5 \times 0.9 = 0.55$ 。

2.10 总结

 **练习 2.11**（编程）对于练习 2.5 中概述的非平稳情况，制作类似于图 2.6 的图。包括 $\alpha = 0.1$ 的步长不变的 ϵ -greedy 算法。使用 200,000 步的运行，并作为每个算法和参数设置的性能度量，使用最近 100,000 步的平均奖励。 □

解 见 `exercise-programming/exercise2.11.py`。

第三章 有限马尔可夫决策过程

3.1 Agent-环境接口

- 练习 3.1 设计三个符合 MDP 框架的您自己的示例任务，为每个任务确定状态、动作和奖励。尽可能使这三个例子各不相同。该框架是抽象且灵活的，可以以多种不同的方式应用。在你的至少一个例子中，以某种方式扩展它的限制。 □

解 (1) 网格迷宫：状态为格子编号，动作为东西南北移动，奖励为到达出口；

(2) 棋类游戏：状态为棋子在棋盘上的位置，动作为棋子的移动，奖励为游戏结果；

(3) 自动驾驶：状态为周围环境、视觉、雷达等传感器信息，动作为转向、加速、刹车等，奖励为到达目的地、避免撞车；

(4) Atari 游戏：状态为屏幕像素输入，动作为键盘/鼠标移动，奖励为游戏增加分数。

- 练习 3.2 MDP 框架是否足以有效地代表所有以目标为导向的学习任务？你能想出任何明确的例外吗？ □

解 不足以。当我们没有足够的计算能力去定义状态和奖励时，例如围棋，必须通过深度学习框架来解决。还比如射击游戏，由于视线限制，agent 没有关于其他玩家的直接信息，但状态会受到队友和对手的影响，使 agent 无法确定先前的动作对当前情况的影响，这使得不是 MDP。

- 练习 3.3 考虑一下驾驶问题。您可以根据油门、方向盘和制动器来定义动作，即身体与机器接触的地方。或者您可以将它们定义为，例如橡胶与道路相接的地方，将您的动作视为轮胎扭矩。或者您还可以定义它们为，例如大脑与身体接触的地方，这些动作是肌肉抽搐来控制您的四肢。或者您可以提高到一个很高的层次，并说您的动作是您选择开车前往的地方。在 agent 与环境之间划清界限的正确层次和正确地方是什么？在什么基础上线路的一个位置优于另一个位置？有没有什么基本的理由让你更喜欢一个位置，或者这是一个自由的选择？ □

解 这个问题是要求正确的线路来定义环境和 agent。正确的划分界限应该可以观察到 agent 的动作对状态的影响。


- 练习 3.4 针对 $p(s', r|s, a)$ 给出一个类似于例 3.3 中的表。它应该具有 s, a, s', r 和 $p(s', r|s, a)$ 的列，以及 $p(s', r|s, a) > 0$ 的每个 4 元组的行。

解 表格如下：

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


3.2 目标和奖励

3.3 回报和 episode

 **练习 3.5** 3.1 节中的方程式适用于连续的情况，需要进行修改（非常轻微）以适用于回合任务。通过给出 (3.3) 的修改版本，表明您知道所需的修改。 □

解

$$\sum_{s' \in \mathcal{S}^+} \sum_{r \in \mathcal{R}} p(s', r|s, a) = 1, \quad \text{for all } s \in \mathcal{S}^+, a \in \mathcal{A}(s) \text{ and } \mathcal{S}^+ \doteq \{\text{all states plus terminal state}\}$$

 **练习 3.6** 假设您将杆子平衡视为回合任务，且使用折扣，除了失败奖励设为-1，所有其他奖励设为 0。那么每个时间的回报是什么？这个回报与这个任务的折扣的、连续的形式有什么不同？ □

解 对于回合任务，回报为：

$$G_t \doteq R_{t+1} + R_{t+2} + R_{t+3} + \cdots + R_T$$


使用折扣后，回报为：

$$G_t \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots + \gamma^{T-t-1} R_T = \sum_{k=0}^{T-t-1} \gamma^k R_{t+k+1}$$

由于失败时奖励为-1，其他情况奖励设置为 0，所以每个时间的回报为：


$$G_t = -\gamma^{T-t-1}$$

该回报实际上与折扣的、连续的情况下的回报 $-\gamma^K$ 相同，其中 K 是失败之前的时间步长。

 **练习 3.7** 想象一下，您正在设计一个运行迷宫的机器人。您决定逃脱迷宫时给予它 +1 的奖励，在其他所有时间给予零的奖励。这项任务似乎自然地分解为 episodes，即连续穿过迷宫，因此您决定将其视为回合性任务，目标是最大化期望的总奖励 (3.7)。在运行学习 agent 一段时间后，您发现它在逃离迷宫方面没有任何改善。这出了什么问题？您是否已有效地向 agent 传达了您想要实现的目标？ □


解 如果您不使用 γ 来执行折扣，则无论 agent 花费多长时间，最大的回报始终为 +1。与

agent 进行沟通的正确方法是在逃离前的每个时间步加-1 的惩罚或增加折扣。

 **练习 3.8** 假设 $\gamma = 0.5$, 以及收到 $T = 5$ 的如下奖励序列, $R_1 = -1, R_2 = 2, R_3 = 6, R_4 = 3, R_5 = 2$ 。那么 G_0, G_1, \dots, G_5 是什么? 提示: 向后工作。 □


解

$$\begin{aligned} G_5 &= 0 \\ G_4 &= R_5 + \gamma G_5 = 2 + 0.5 \times 0 = 2 \\ G_3 &= R_4 + \gamma G_4 = 3 + 0.5 \times 2 = 4 \\ G_2 &= R_3 + \gamma G_3 = 6 + 0.5 \times 4 = 8 \\ G_1 &= R_2 + \gamma G_2 = 2 + 0.5 \times 8 = 6 \\ G_0 &= R_1 + \gamma G_1 = -1 + 0.5 \times 6 = 2 \end{aligned}$$

 **练习 3.9** 假设 $\gamma = 0.9$, 以及奖励序列为 $R_1 = 2$, 然后是 7s 的无限序列。那么 G_1 和 G_0 是什么? □

解

$$\begin{aligned} G_1 &= 7 \times \sum_{k=0}^{\infty} \gamma^k R_{k+2} = 7 \times \frac{1}{1-\gamma} = 7 \times \frac{1}{1-0.9} = 70 \\ G_0 &= R_1 + \gamma \times 7 \times \sum_{k=1}^{\infty} \gamma^k R_{k+1} = 2 + 7 \times \frac{\gamma}{1-\gamma} = 2 + 7 \times \frac{0.9}{1-0.9} = 2 + 7 \times 90 = 65 \\ G_0 &= R_1 + \gamma G_1 = 2 + 0.9 \times 70 = 65 \end{aligned}$$

 **练习 3.10** 证明 (3.10) 中的第二个等式。 □

解 因为奖励为不为 0 的常数, 以及 $\gamma < 1$, 则:

$$\begin{aligned} S_N &\doteq \sum_{k=0}^N \gamma^k \\ \gamma S_N - S_N &= \gamma^{N+1} - 1 \\ S_N &= \frac{1 - \gamma^{N+1}}{1 - \gamma} \\ G_t &\doteq \lim_{N \rightarrow \infty} S_N = \frac{1}{1 - \gamma} \end{aligned}$$