强化学习原理

第二章读书笔记

姓名：石若川 学号：2111381 专业：智能科学与技术

2 多臂赌博机

强化学习与其他机器学习方法最大的不同，就在于前者的训练信号是用来**评估**给定动作的好坏的，而不是通过给出正确动作范例来进行直接的**指导**。这使得主动地反复试验以试探出好的动作变得很有必要。

- 评估性反馈：评估性反馈只能表明当前采取的动作的好坏程度，但却无法确定当前采取的动作是不是所有可能性中最好的或者最差的。
- 指导性反馈：应该选择的正确动作是什么，并且这个正确动作和当前实际采取的动作无关，这是有监督学习的基本方式，其被广泛应用于模式分类、人工神经网络和系统辨识等。

上述两种不同的反馈有着很大的不同：评估性反馈依赖于当前采取的动作，即采取不同的动作会得到不同的反馈；而指导性反馈则不依赖于当前采取的动作，即采取不同的动作也会得到相同的反馈。

2.1 k 臂赌博机问题

1. 问题描述

考虑如下的一个学习问题：你要重复地在 k 个选项或动作中进行选择，每次做出选择之后，你都会得到一定数值的收益，收益由你选择的动作决定的平稳概率分布产生。目标是在某一段时间内最大化总收益的期望。

在 k 臂赌博机问题中，进行以下定义：

- 将在时刻 t 时选择的动作记作 A_t，并将对应的收益记作 R_t。
- k 个动作中的每一个在被选择时都有一个期望或者平均收益，称为这个动作的价值。动作 a 的价值记作 $q_*(a)$

$$q_*(a) \doteq \mathbb{E}[R_t \mid A_t = a]$$

由于不能确切知道动作的价值，所以进行估计。将对动作 a 在时刻 t 时的价值的估计记作 $Q_t(a)$，希望它接近 $q_*(a)$。

2. 贪心动作

如果持续对动作的价值进行估计，那么在任一时刻都会至少有一个动作的估计价值是最高的，将这些对应最高估计价值的动作称为 **“贪心”** 的动作。当在这些动作中选择时，称为 **“利用”** 当前你所知道的关于动作的价值的知识。如果不是如此，而是选择非贪心的动作，则称此为 **“试探”**，因为这可以让你改善对非贪心动作的价值的估计。“利用” 对于最大化当前这一时刻的期望收益是正确的做法，但是 “试探” 从长远来看可能会带来总体收益的最大化。

2.2 动作-价值方法

通过计算实际收益的平均值来估计动作的价值的一个简单方法——采样平均方法

$$Q_t(a) \doteq \frac{\text{sum of rewards when } a\text{taken prior to}t}{\text{number of times } a\text{taken prior to}t} = \frac{\sum_{i=1}^{t-1} R_i \cdot \mathbb{1}_{A_i=a}}{\sum_{i=1}^{t-1} \mathbb{1}_{A_i=a}},$$

其中，$\mathbb{1}_{predicate}$ 表示随机变量，当 predicate 为真时其值为 1，反之为 0。当分母为 0 时将 $Q_t(a)$ 定义为某个默认值，比如 $Q_t(a) = 0$。当分母趋向无穷大时，根据大数定律 $Q_t(a)$ 会收敛到 $q_*(a)$。我们将这种估计动作价值的方法称为采样平均方法，因为每一次估计都是相关收益样本的平均。

最简单的动作选择规则是选择具有最高估计值的动作，即贪心动作。如果有多个贪心动作，那就任意选择一个，比如随机挑选。将这种贪心动作的选择方法记作

$$A_t \doteq \underset{a}{\text{argmax}} Q_t(a),$$

其中 $\underset{a}{\text{argmax}}$ 是使得 $Q_t(a)$ 值最大的动作 a。选择的贪心动作总是利用当前的知识最大化眼前的收益。这种方法根本不花时间去尝试明显的劣质动作，看看它们是否真的会更好。贪心策略的一个简单替代策略是大部分时间都表现得贪心，但偶尔 (比如以一个很小的概率 ϵ) 以独立于动作-价值估计值的方式从所有动作中等概率随机地做出选择。将使用这种近乎贪心的选择规则的方法称为 $\epsilon-greedy$ 方法。这类方法的一个优点是，如果时刻可以无限长，则每一个动作都会被无限次采样，从而确保所有的 $Q_t(a)$ 收敛到 $q_*(a)$。这当然也意味着选择最优动作的概率会收敛到大于 $1-\epsilon$，即接近确定性选择。

2.3 10 臂测试平台

为了大致评估贪心方法和 $\epsilon - greedy$ 方法相对的有效性，在 10 臂赌博机问题上对两方法进行定量比较。如图2.1所示，动作的真实价值为 $q_*(a), a = 1, \ldots, 10$，从一个均值为 0 方差为 1 的标准正态 (高斯) 分布中选择。当对应于该问题的学习方法在时刻 t 选择 A_t 时，实际的收益 R_t 则由一个均值为 $q_*(A_t)$ 方差为 1 的正态分布决定。

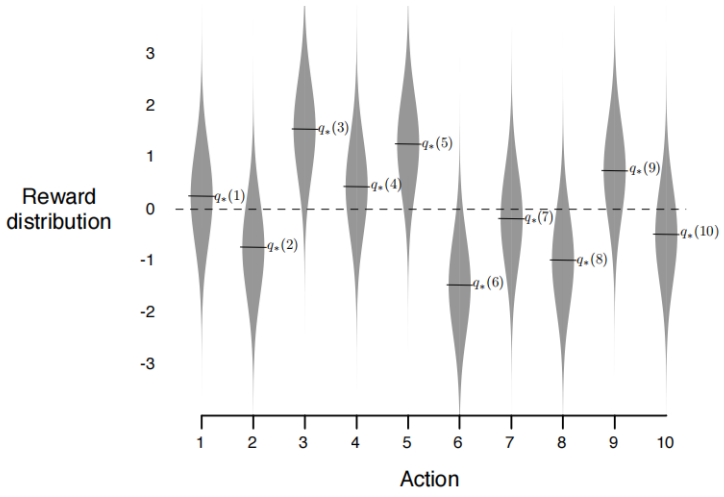
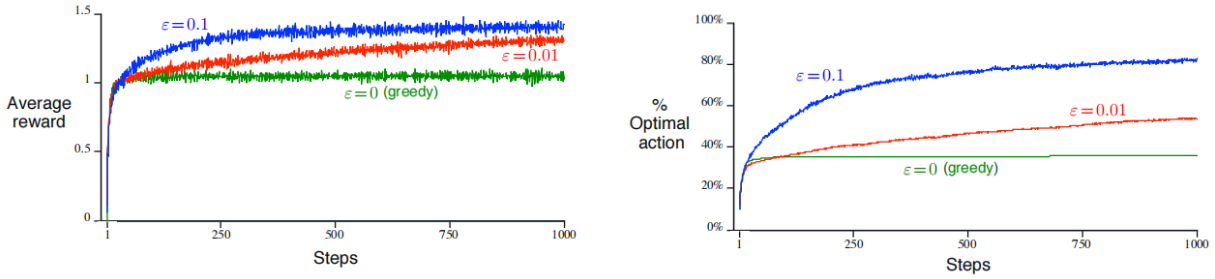


图 2.1: 10 臂赌博机问题测试平台的概率分布

作者在 10 臂测试平台上对贪心方法和 $\epsilon - greedy$ 方法（$\epsilon = 0.01$ 和 $\epsilon = 0.1$）进行测试，测试结果如图2.2所示。左图中，贪心方法在最初增长得略微快一些，但是随后稳定在一个较低的水平。相对

于在这个测试平台上最好的可能收益 1.55，这个方法每时刻只获得了大约 1 的收益。从长远来看，贪心的方法表现明显更糟，因为它经常陷入执行次优的动作。右图中显示贪心方法只在大约三分之一的任务中找到最优的动作。在另外三分之二的动作中，最初采样得到的动作非常不好，贪心方法无法跳出来找到最优的动作。$\epsilon - greedy$ 方法最终表现更好，因为它们持续地试探并且提升找到最优动作的机会。$\epsilon = 0.1$ 的方法试探得更多，通常更早发现最优的动作，但是在每时刻选择这个最优动作的概率却永远不会超过 91%(因为要在 $\epsilon = 0.1$ 的情况下试探)。$\epsilon = 0.01$ 的方法改善得更慢，但是在图中的两种测度下，最终的性能表现都会比 $\epsilon = 0.1$ 的方法更好。为了充分利用高和低的值的优势，随着时刻的推移来逐步减小也是可以的。



(a) 平均收益曲线 (b) 最优动作占比曲线

图 2.2: $\epsilon - greedy$ 方法与贪心方法在 10 臂测试平台上的对比

$\epsilon - greedy$ 方法相对于贪心方法的优点依赖于任务。比方说，假设收益的方差更大，不是 1 而是 10。由于收益的噪声更多，所以为了找到最优的动作需要更多次的试探，而 $\epsilon - greedy$ 方法会比贪心方法好很多。但是，如果收益的方差是 0，那么贪心方法会在尝试一次之后就知道每一个动作的真实价值。在这种情况下，贪心方法实际上可能表现最好，因为它很快就会找到最佳的动作，然后再也不会进行试探。但是，即使在有确定性的情况下，如果弱化一些假设，对试探也有很大的好处。例如，假设赌博机任务是非平稳的，也就是说，动作的真实价值会随着时间而变化。在这种情况下，即使在有确定性的情况下，试探也是需要的，这是为了确认某个非贪心的动作不会变得比贪心动作更好。

2.4 增量式实现

对于新的收益，可以设计增量式公式以常数级计算新的均值。

$$
\begin{aligned}
Q_{n+1} &= \frac{1}{n}\sum_{i=1}^{n} R_i \\
&= \frac{1}{n}\left(R_n + \sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\left(R_n + (n-1)\frac{1}{n-1}\sum_{i=1}^{n-1} R_i\right) \\
&= \frac{1}{n}\Big(R_n + (n-1)Q_n\Big) \\
&= \frac{1}{n}\Big(R_n + nQ_n - Q_n\Big) \\
&= Q_n + \frac{1}{n}\Big[R_n - Q_n\Big]
\end{aligned}
$$

更新公式的一般形式为：

$$\text{新估计值} \leftarrow \text{旧估计值} + \text{步长} \times \left[\text{目标} - \text{旧估计值}\right]$$

表达式 [目标-旧估计值] 是估计值的偏差。误差会随着向“目标”靠近的每一步而减小。

A simple bandit algorithm

Initialize, for $a = 1$ to k:
$Q(a) \leftarrow 0$
$N(a) \leftarrow 0$

Loop forever:
$$A \leftarrow \begin{cases} \operatorname{argmax}_a Q(a) & \text{with probability } 1-\varepsilon \quad \text{(breaking ties randomly)} \\ \text{a random action} & \text{with probability } \varepsilon \end{cases}$$
$R \leftarrow bandit(A)$
$N(A) \leftarrow N(A) + 1$
$Q(A) \leftarrow Q(A) + \frac{1}{N(A)}\left[R - Q(A)\right]$

2.5 跟踪一个非平稳问题

1. 指数近因加权平均（exponential recency-weighted average）

若赌博机的收益概率是随时间变化的，那么给近期收益赋予比过去收益更高的权值就是一种合理的处理方式。一种流行的方法就是使用固定步长。比如，Q_n 的增量规则可以改为

$$Q_{n+1} \doteq Q_n + \alpha\left[R_n - Q_n\right]$$

式中，步长参数 $\alpha \in (0, 1]$ 是一个常数。这使得 Q_{n+1} 成为对过去的收益和初始的估计 Q_1 的加权平均

$$\begin{aligned} Q_{n+1} &= Q_n + \alpha\left[R_n - Q_n\right] \\ &= \alpha R_n + (1-\alpha)Q_n \\ &= \alpha R_n + (1-\alpha)\left[\alpha R_{n-1} + (1-\alpha)Q_{n-1}\right] \\ &= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 Q_{n-1} \\ &= \alpha R_n + (1-\alpha)\alpha R_{n-1} + (1-\alpha)^2 \alpha R_{n-2} + \\ &\quad \cdots + (1-\alpha)^{n-1}\alpha R_1 + (1-\alpha)^n Q_1 \\ &= (1-\alpha)^n Q_1 + \sum_{i=1}^{n} \alpha(1-\alpha)^{n-i} R_i \end{aligned}$$

赋给收益 R_i 的权值 $\alpha(1-\alpha)^{n-i}$ 依赖于它被观测到的具体时刻与当前时刻的差即 $n-i$。$1-\alpha$ 小于 1，因此赋予民的权值随着相隔次数的增加而递减。因此，这个方法有时候也被称为“指数近因加权平均”。

2. 收敛概率为 1 的条件

设 $\alpha_n(a)$ 表示用于处理第 n 次选择动作 a 后收益的步长参数。随机逼近理论中给出了保证收敛概

率概率为 1 所需的条件

$$\sum_{n=1}^{\infty} \alpha_n(a) = \infty \quad \text{and} \quad \sum_{n=1}^{\infty} \alpha_n^2(a) < \infty$$

第一个条件是要求保证有足够大的步长，最终克服任何初始条件或随机波动。第二个条件保证最终步长变小，以保证收敛。

在常数步长参数 $\alpha_n(n) = \alpha$ 中，不满足第二个条件，说明估计无法完全收敛。而这是在非平稳环境中想要的。此外，满足以上两条件的步长参数序列往往收敛得很慢，或需要大量的调试，因此在实际应用中很少用到。

2.6 乐观估计值

初始动作的价值提供了一种简单的试探方式。比如一个 10 臂的测试平台，替换掉原先的初始值 0，将它们全部设为 +5。如前所述，在这个问题中，$q_*(a)$ 是按照均值为 0 方差为 1 的正态分布选择的。因此 +5 的初始值是一个过度乐观的估计. 但是这种乐观的初始估计却会鼓励动作-价值方法去试探。因为无论哪一种动作被选择，收益都比最开始的估计值要小，因此学习器会对得到的收益感到“失望”，从而转向另一个动作。其结果是，所有动作在估计值收敛之前都被尝试了好几次。即使每一次都按照贪心法选择动作，系统也会进行大量的试探。

图2.3展示了在 10 臂测试平台上设定 $Q_1(a) = +5$，并采用贪心算法的结果，并与 $\epsilon - greedy$ 算法（$Q_1(a) = 0$）进行比较。刚开始乐观初始化方法表现得比较糟糕，因为它需要试探更多次，但是最终随着时间的推移，试探的次数减少，它的表现也变得更好。作者把这种鼓励试探的技术叫作乐观初始价值。但它不太适合非平稳问题，因为它试探的驱动力天生是暂时的。如果任务发生了变化，对试探的需求变了，则这种方法就无法提供帮助。事实上，任何仅仅关注初始条件的方法都不太可能对一般的非平稳情况有所帮助。开始时刻只出现一次，因此不应该过多地关注它。对于采样平均法也是如此，它也将时间的开始视为一种特殊的事件，用相同的权重平均所有后续的收益。

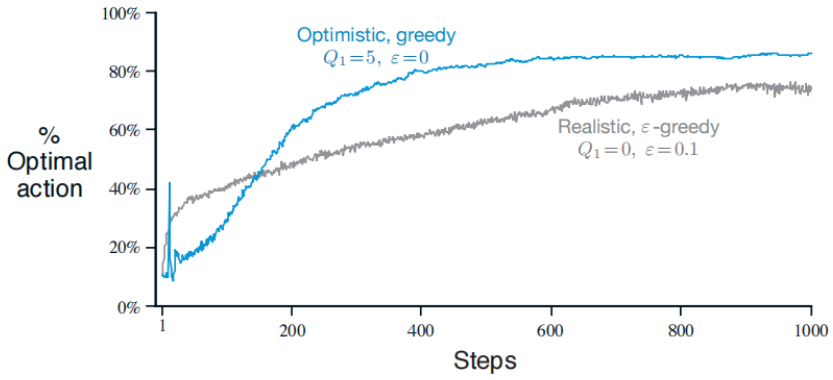


图 2.3: 乐观初始值在 10 臂赌博机测试平台的运行效果

2.7 基于置信度上界的动作选择

贪心动作虽然在当前时刻看起来最好，但实际上其他一些动作可能从长远看更好。$\epsilon - greedy$ 算法会尝试选择非贪心的动作，但是这是一种盲目的选择，因为它不大会去选择接近贪心或者不确定性特别大的动作。在非贪心动作中，最好是根据它们的潜力来选择可能事实上是最优的动作，这就要考虑到它们的估计有多接近最大值，以及这些估计的不确定性。

一个有效的方法是置信度上界 (upper confdence bound, UCB) 的方法，按照以下公式选择动作

$$A_t \doteq \arg\max_a \left[Q_t(a) + c\sqrt{\frac{\ln t}{N_t(a)}} \right]$$

在这个公式里，$N_t(a)$ 表示在时刻 t 之前动作 a 被选择的次数。c 是一个大于 0 的数，它控制试探的程度。如果 $N_t(a) = 0$，则 a 就被认为是满足最大化条件的动作。

这种基于置信度上界的动作选择的思想是，平方根项是对 a 动作值估计的不确定性或方差的度量。因此，最大值的大小是动作 a 的可能真实值的上限，参数 c 决定了置信水平。每次选 a 时，不确定性可能会减小；由于 $N_t(a)$ 出现在不确定项的分母上，因此随着 $N_t(a)$ 的增加，这一项就减小了。另一方面，每次选择 a 之外的动作时，在分子上的 t 增大，而 $N_t(a)$ 却没有变化，所以不确定性增加了。自然对数的使用意味着随着时间的推移，增加会变得越来越小，但它是无限的。所有动作最终都将被选中，但是随着时间的流逝，具有较低价值估计的动作或者已经被选择了更多次的动作被选择的频率较低。

图2.4展示了 UCB 算法在 10 臂赌博机测试平台上的运行效果，并和 $\epsilon - greedy$ 算法对比。如图所示 UCB 算法的表现比 $\epsilon - greedy$ 算法更好。但是与 $\epsilon - greedy$ 算法相比，UCB 更难推广到更一般的强化学习问题。一个困难是在处理非平稳问题时，它需要更复杂的方法。另一个难题是要处理大的状态空间，特别是函数近似问题。

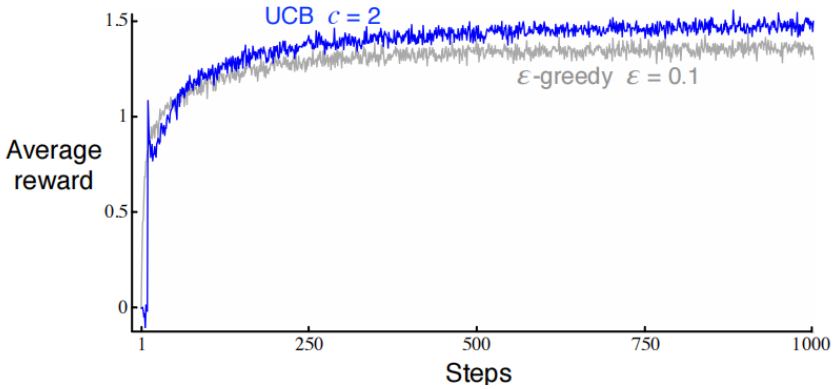


图 2.4: UCB 算法在 10 臂赌博机测试平台上的运行

2.8 梯度赌博机算法

针对每个动作 a 考虑学习一个数值化的偏好函数 $H_t(a)$。偏好函数越大，动作就越频繁地被选择，但偏好函数的概念并不是从“收益”的意义上提出的，而是一个动作对另一个动作的相对偏好。如果给每一个动作的偏好函数都加上 1000，那么对于按照如下 softmax 分布 (吉布斯或玻尔兹曼分布) 确定的动作概率没有任何影响

$$\Pr\{A_t = a\} \doteq \frac{e^{H_t(a)}}{\sum_{b=1}^{k} e^{H_t(b)}} \doteq \pi_t(a)$$

其中，$\pi_t(a)$ 用来表示动作 a 在时刻 t 时被选择的概率。所有偏好函数的初始值都是一样的，所以每个动作被选择的概率是相同的。

基于随机梯度上升的思想，又提出了一种自然学习算法。在每个步骤中，选择动作 A_t 获得收益

R_t 之后，偏好函数会按下式进行更新

$$\begin{aligned} H_{t+1}(A_t) &\doteq H_t(A_t) + \alpha\big(R_t - \bar{R}_t\big)\big(1 - \pi_t(A_t)\big), && \text{and} \\ H_{t+1}(a) &\doteq H_t(a) - \alpha\big(R_t - \bar{R}_t\big)\pi_t(a), && \text{for all a} \neq A_t, \end{aligned} \tag{1}$$

其中，a 是一个大于 0 的数，表示步长。$\bar{R}_t \in \mathbb{R}$ 是在时刻 t 内所有收益的平均值。$\bar{R}_t \in \mathbb{R}$ 项作为比较收益的一个基准项。如果收益高于它，那么在未来选择动作 A 的概率就会增加，反之概率就会降低。未选择的动作被选择的概率上升。

图2.5展示了在一个 10 臂测试平台问题的变体上采用梯度赌博机算法的结果. 在这个问题中，它们真实的期望收益是按照平均值为 +4 而不是 0(方差与之前相同) 的正态分布来选择的。所有收益的这种变化对梯度赌博机算法没有任何影响，因为收益基准项让它可以马上适应新的收益水平。如果没有基准项 (即把 $\bar{R}_t$ 设为常数 0)，那么性能将显著降低。

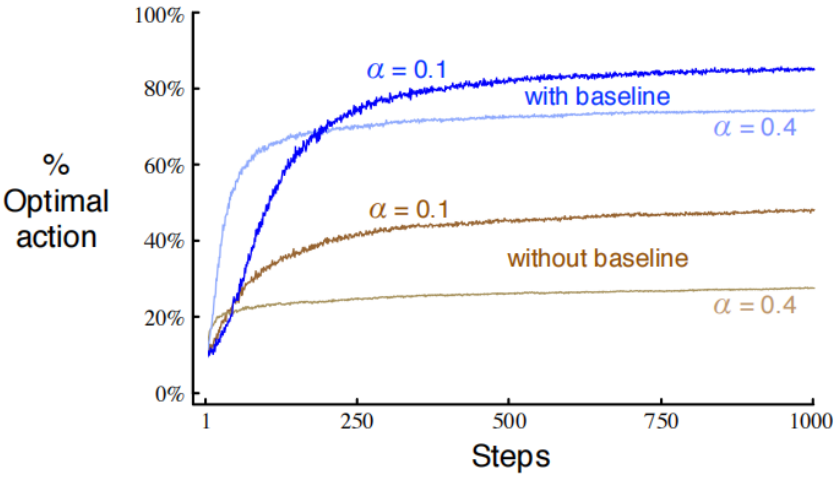


图 2.5: 梯度赌博机算法（包括与未包括基准项）在 10 臂赌博机测试平台上的运行效果

在精确的梯度上升算法中，每个动作的偏好函数 $H_t(a)$ 与增量对性能的影响成正比

$$H_{t+1}(a) \doteq H_t(a) + \alpha \frac{\partial \mathrm{E}\left[R_t\right]}{\partial H_t(a)} \tag{2}$$

此处性能指标定义为总体的期望收益

$$\mathbb{E}\left[R_t\right] = \sum_x \pi_t(x) q_*(x)$$

增量产生的影响为性能指标对动作偏好的偏导数。可以证明公式1和公式2采用期望价值时是等价的，即公式1是随机梯度上升方法的一个实例。首先分析一下精确的性能梯度定义

$$\begin{aligned} \frac{\partial \mathbb{E}\left[R_t\right]}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)}\left[\sum_x \pi_t(x) q_*(x)\right] \\ &= \sum_x q_*(x) \frac{\partial \pi_t(x)}{\partial H_t(a)} \\ &= \sum_x \left(q_*(x) - B_t\right) \frac{\partial \pi_t(x)}{\partial H_t(a)}, \end{aligned}$$

其中，B_t 称为“基准项”，可以是任何不依赖于 x 的标量。因为 $\sum_x \frac{\partial \pi_t(x)}{\partial H_t(a)} = 0$，所以可以将 B_t 加进来。然后将求和公式的每一项都乘上 $\pi_t(x)/\pi_t(x)$

$$\begin{aligned}
\frac{\partial \mathrm{E}[R_t]}{\partial H_t(a)} &= \sum_x \pi_t(x)\big(q_*(x) - B_t\big)\frac{\partial \pi_t(x)}{\partial H_t(a)}/\pi_t(x) \\
&= \mathbb{E}\left[\big(q_*(A_t) - B_t\big)\frac{\partial \pi_t(A_t)}{\partial H_t(a)}/\pi_t(A_t)\right] \\
&= \mathbb{E}\left[\big(R_t - \bar{R}_t\big)\frac{\partial \pi_t(A_t)}{\partial H_t(a)}/\pi_t(A_t)\right] \\
&= \mathbb{E}\left[\big(R_t - \bar{R}_t\big)\pi_t(A_t)\left(\mathbb{1}_{a=A_t} - \pi_t(a)\right)/\pi_t(A_t)\right] \\
&= \mathbb{E}\left[\big(R_t - \bar{R}_t\big)\left(1_{a=A_t} - \pi_t(a)\right)\right]
\end{aligned}$$

在这里选择 $B_t = \bar{R}_t$，并且将 $\bar{R}_t$ 用 $q_*(A)$ 代替。这个选择是可行的，因为 $\mathbb{E}[R_t|A_t] = q_*(A_t)$，而且 R_t(给定 A_t) 与任何其他东西都不相关。很快就可以确定 $\frac{\partial \pi_t(x)}{\partial H_t(a)} = \pi_t(x)\big(\mathbb{1}_{a=x} - \pi_t(a)\big)$，$\mathbb{1}_{a=x}$ 表示如果 $a = x$ 就取 1，否则取 0。

将公式2中性能指标的梯度用一个单独样本的期望值代替，可得

$$H_{t+1}(a) = H_t(a) + \alpha\big(R_t - \bar{R}_t\big)\big(\mathbb{1}_{a=A_t} - \pi_t(a)\big), \quad \text{for all } a$$

这与公式1中的原始公式是一致的。

现在只需要证明 $\frac{\partial \pi_t(x)}{\partial H_t(a)} = \pi_t(x)(\mathbb{1}_{a=x} - \pi_t(a))$

$$\begin{aligned}
\frac{\partial \pi_t(x)}{\partial H_t(a)} &= \frac{\partial}{\partial H_t(a)}\pi_t(x) \\
&= \frac{\partial}{\partial H_t(a)}\left[\frac{e^{H_t(x)}}{\sum_{y=1}^k e^{H_t(y)}}\right] \\
&= \frac{\frac{\partial e^{H_t(x)}}{\partial H_t(a)}\sum_{y=1}^k e^{H_t(y)} - e^{H_t(x)}\frac{\partial \sum_{y=1}^k e^{H_t(y)}}{\partial H_t(a)}}{\left(\sum_{y=1}^k e^{H_t(y)}\right)^2} \qquad \text{(by the quotient rule)} \\
&= \frac{\mathbb{1}_{a=x}e^{H_t(x)}\sum_{y=1}^k e^{H_t(y)} - e^{H_t(x)}e^{H_t(a)}}{\left(\sum_{y=1}^k e^{H_t(y)}\right)^2} \qquad \text{(because } \frac{\partial e^x}{\partial x} = e^x\text{)} \\
&= \frac{\mathbb{1}_{a=x}e^{H_t(x)}}{\sum_{y=1}^k e^{H_t(y)}} - \frac{e^{H_t(x)}e^{H_t(a)}}{\left(\sum_{y=1}^k e^{H_t(y)}\right)^2} \\
&= \mathbb{1}_{a=x}\pi_t(x) - \pi_t(x)\pi_t(a) \\
&= \pi_t(x)\big(\mathbb{1}_{a=x} - \pi_t(a)\big). \qquad \text{Q.E.D.}
\end{aligned}$$

以上已经证明了梯度赌博机算法的期望更新与期望收益的梯度是相等的，因此该算法是随机梯度上升算法的一种。这就保证了算法具有很强的收敛性。注意，对于收益基准项，除了要求它不依赖于所选的动作之外，不需要其他任何的假设。例如，可以将其设置为 0 或 1000，算法仍然是随机梯度上升算法的一个特例。基准项的选择不影响算法的预期更新，但它确实会影响更新值的方差，从而影响收敛速度 (如图2.5所示)。采用收益的平均值作为基准项可能不是最好的，但它很简单，并且在实践中很有效。

2.9 关联搜索（上下文相关的赌博机）

在一般的强化学习任务中，往往有不止一种情境，它们的目标是学习一种策略：一个从特定情境到最优动作的映射。为了进行一般性问题分析，下面简要地探讨从非关联任务推广到关联任务的最简单的方法。

举例说明，假设有一系列不同的 k 臂赌博机任务，每一步都要随机地面对其中的一个。因此，赌博机任务在每一步都是随机变化的。从观察者的角度来看，这是一个单一的、非平稳的 k 臂赌博机任务，其真正的动作价值是每步随机变化的。现在假设，当遇到某一个 k 臂赌博机任务时，会得到关于这个任务的编号的明显线索 (但不是它的动作价值)。那么，现在可以学习一些任务相关的操作策略，把每个任务和该任务下最优的动作直接关联起来。有了这种任务相关的策略，在知道任务编号信息时通常要比不知道任务编号信息时做得更好。

这是一个关联搜索任务的例子，因为它既涉及采用试错学习去搜索最优的动作，又将这些动作与它们表现最优时的情境关联在一起。关联搜索任务现在通常在文献中被称为上下文相关的赌博机。关联搜索任务介于 k 臂赌博机问题和完整强化学习问题之间。它与完整强化学习问题的相似点是，它需要学习一种策略。但它又与 k 臂赌博机问题相似，体现在每个动作只影响即时收益。如果允许动作可以影响下一时刻的情境和收益，那么这就是完整的强化学习问题。

2.10 总结

本章介绍了集中平衡试探与利用的简单方法：

- $\epsilon - greedy$ 方法在一小段时间内进行随机的动作选择。
- UCB 方法虽然采用确定的动作选择，却可以通过在每个时刻对那些具有较少样本的动作进行优先选择来实现试探。
- 梯度赌博机算法则不估计动作价值，而是利用偏好函数，使用 softmax 分布来以一种分级的、概率式的方式选择更优的动作。
- 将收益的初值进行乐观的设置，就可以让贪心方法也能进行显式试探。

图2.6显示了本章各种赌博机算法的性能曲线，展示了各种算法和参数超过 1000 步的平均收益。这些算法的参数都是相当不敏感的，在一个数量级上表现得很好。总的来说，UCB 表现最好。

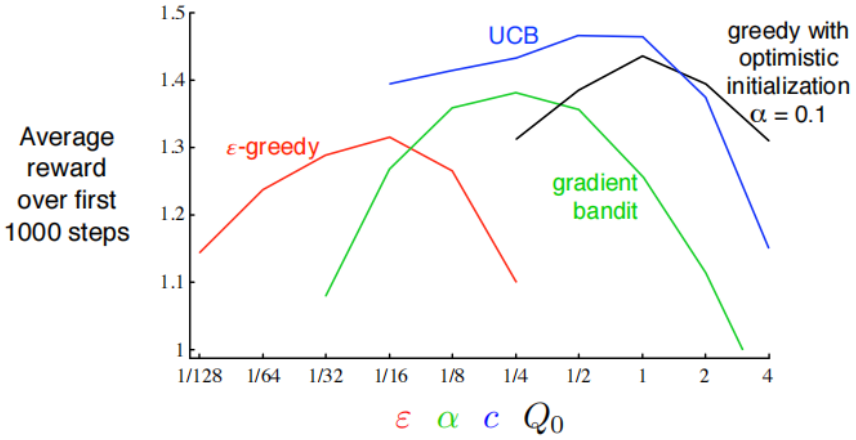


图 2.6: 本章不同赌博机算法的参数调整对比

在 k 臂赌博机问题中，平衡试探与利用的一个经典解决方案是计算一个名为 Gittins 指数的特殊函数。这为一些赌博机问题提供了一个最优的解决方案，比在本章中讨论的方法更具有一般性，但前提是已知可能问题的先验分布。但是，这种方法的理论和可计算性都不能推广到完整强化学习问题。

贝叶斯方法假定已知动作价值的初始分布，然后在每步之后更新分布。一般来说，更新计算可能非常复杂，但对于某些特殊分布 (称为共轭先验) 则很容易。这样，我们就可以根据动作价值的后验概率，在每一步中选择最优的动作。这种方法，有时称为后验采样或 Thompson 采样，通常与我们在本章中提出的最好的无分布方法性能相近。

贝叶斯方法甚至可以计算出试探与利用之间的最佳平衡。对于任何可能的动作，我们都可以计算出它对应的即时收益的分布，以及相应的动作价值的后验分布。这种不断变化的分布成为问题的信息状态。假设问题的视界有 1000 步，则可以考虑所有可能的动作，所有可能的收益，所有可能的下一个动作，所有下一个收益，等等，依此类推到全部 1000 步。有了这些假设，可以确定每个可能的事件链的收益和概率，并且只需挑选最好的。但可能性树会生长得非常快，即使只有两种动作和两种收益，树也会有 2^{2000} 个子节点完全精确地进行这种庞大的计算通常是不现实的，但可能可以有效地近似。贝叶斯方法有效地将赌博机问题转变为完整强化学习问题的一个实例。最后，我们可以使用近似强化学习方法来逼近最优解。