Stochastic Multi-Armed-Bandit Problem with Non-stationary Rewards

Omar Besbes
Columbia University
New York, NY
ob2105@columbia.edu

Yonatan Gur
Stanford University
Stanford, CA
ygur@stanford.edu

Assaf Zeevi
Columbia University
New York, NY
assaf@gsb.columbia.edu

【强化学习 92】Non-stationary MAB

张楚珩
清华大学 交叉信息院博士在读

MAB 问题（multi-armed-bandit problem）在 non-stationary reward 设定下的一种算法分析。关于 MAB 问题的具体定义，可以看 Sutton 的书（大概是第二章）。

原文传送门

Besbes, Omar, Yonatan Gur, and Assaf Zeevi. "Stochastic multi-armed-bandit problem with non-stationary rewards." Advances in neural information processing systems. 2014.

特色

MAB 问题一般有两种设定：一种是 stochastic reward，即每个臂上奖励是从一个稳定的分布中 i.i.d. 采样得到的，该问题的 regret 和一直拉任意的一个臂（或者可以认为拉最优的臂）；另一种是 adversary reward，即考虑每个臂上的奖励可以被任意选择（即可能是有一个对手跟你『作对』），该问题的 regret 也是和一直拉任意一个臂相比。

本文研究的设定叫做 non-stationary reward，它可以被看做介于 stochastic 和 adversary 中间的一种状态，即各个臂上 reward 的分布可以变化，但是变化不太大。基于这种设定，其 regret 的基准可以选择的比 adversary setting 下更强一些，即可以和每一轮中最优的那个臂相比较。之前 regret 中基准为一直拉同一个臂，这种情形文章把它叫做 static oracle，现在期望比较的是每一轮中最优的那个臂，这种情形叫做 dynamic oracle。和 dynamic oracle 的比较更为困难。

过程

1. 背景

MAB 问题最原始的设定就是 stochastic setting，之后大家把该问题推广到各种不同的设定下。这篇文章也讲了一下之前的一些关于 reward 变化的 setting。

- 被拉过的臂上的奖励分布会产生变化；

- 被拉过的臂上的奖励分布产生变化，并且该变化的过程已知；
- 给定总的奖励分布变化的次数；
- 给定最优臂的轮换次数；
- Adversary 设定。

2. 问题设定

2.1 基本的 MAB 问题设定

- 臂： $\mathcal{K}=\{1,\cdots,K\}$
- 玩的轮数： $\mathcal{T}=\{1,\cdots,T\}$
- 在 $t\in\mathcal{T}$ 的时候选取 $k\in\mathcal{K}$ ，得到一个奖励 $X_t^k\in[0,1]$
  - 注意到这里把奖励归一化了
  - 按理来说，只有被选中的臂才会产生一个奖励（随机变量），这里便于分析，假设选不选中都有这么一个量
- 奖励分布的均值： $\mu_t^k=\mathbb{E}[X_t^k]$
- 每一轮最优臂上的奖励分布均值： $\mu_t^*=\max_{k\in\mathcal{K}}\{\mu_t^k\}$
- 一个臂上整个游戏过程中的均值： $\mu^k=\{\mu_t^k\}_{t=1}^T$
- 所有臂上整个游戏过程中的均值： $\mu=\{\mu^k\}_{k=1}^K$

2.2 变换缓慢的奖励分布

下面规定奖励分布变化的不要太快（否则就退化为 adversary 设定了）。变化不要太快的标准是每一轮均值变化最大的那个臂的的变化量加起来不要超过一个规定好的 variation budget V_T ，即奖励均值应该属于以下集合

$$\mathcal{V}=\left\{\mu\in[0,1]^{K\times T}\ :\ \sum_{t=1}^{T-1}\sup_{k\in\mathcal{K}}\left|\mu_t^k-\mu_{t+1}^k\right|\le V_T\right\}.$$

同时，文章要求 variation budget 不能超过 $K^{-1}T$ ，即臂是数目越多，允许的变化量越小。

同时可以注意到，这样的 variation budget 只要求整个过程中变化不超过 V_T ，但是对于这个变化发生在整个过程中的什么地方没有限制，即以下两种变化形式，都对应同样的 V_T 。

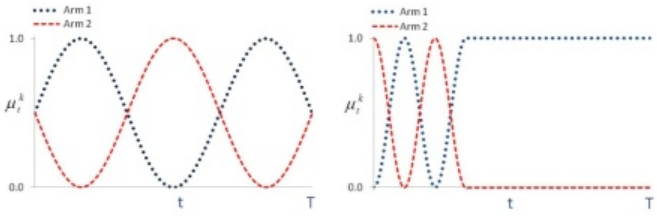

Figure 1: Two instances of variation in the mean rewards: (*Left*) A fixed variation budget (here equal to 3) is "spent" over the whole horizon. (*Right*) The same budget is "spent" in the first third of the horizon.

2.3 策略空间

这里考虑的策略空间是一个 non-anticipating 的策略，即非预测的。大致可以认为策略只与过去的经历有关，但是可以是随机性的策略。

$$\pi_t = \begin{cases} \pi_1(U) & t=1, \\ \pi_t\left(X_{t-1}^{\pi}, \ldots, X_1^{\pi}, U\right) & t=2,3,\ldots, \end{cases}$$

2.4 Regret

这里考虑的 regret 的基准是一个 dynamic oracle，即

$$\mathcal{R}^{\pi}(\mathcal{V}, T) = \sup_{\mu \in \mathcal{V}} \left\{ \sum_{t=1}^{T} \mu_t^* - \mathbb{E}^{\pi} \left[\sum_{t=1}^{T} \mu_t^{\pi} \right] \right\},$$

需要注意

- 这里的 regret 是均值和均值的比较。
- $\mathbb{E}^{\pi}$ 包含策略的随机性（即，策略每次可能按照一定概率分布随机选择一个臂），也包含环境的随机性。后面一点不太好理解，因为这里写的是均值，因此在单步拉臂上的随机性是没有了，但是 X_t^{π} 的随机性能够影响策略，从而最后影响期望。
- 如果 regret 相对于 T 是 sub-linear 的，称相应的策略是 long-run average optimal 的。

把所有策略能产生的最优 regret 记为

$$\mathcal{R}^*(\mathcal{V}, T) = \inf_{\pi \in \mathcal{P}} \mathcal{R}^{\pi}(\mathcal{V}, T)$$

3. 算法性能下界

在上述设定中，最好算法的性能也会有一个下界，这里构造了一个这样的下界。构造下界的方法是找出一个环境『对抗』算法的方式，证明对于任意的算法都能遭受到这种『对抗』方式多大的损失。

Theorem 1 *Assume that rewards have a Bernoulli distribution. Then, there is some absolute constant $C > 0$ such that for any policy $\pi \in \mathcal{P}$ and for any $T \geq 1$, $K \geq 2$ and $V_T \in \left[K^{-1}, K^{-1}T\right]$,*

$$\mathcal{R}^{\pi}(\mathcal{V}, T) \geq C\left(KV_T\right)^{1/3} T^{2/3}.$$

先看结论本身，再讲构造方法。

- 对于 stationary 的情况（stochastic setting），已知最优算法 UCB1 能够达到 $\sqrt{T}$ 的 regret。本文设定下，最优就只有 $T^{2/3}$，这是『price of non-stationary』，即 variation budget 如果是和 T 无关的常数，但是只要奖励分布会变，就会产生一个这样的代价。
- 当奖励会变时，我们可以只利用最近的样本去估计每个臂上的均值，这样相应的 bias 会比较小；也可以把比较远的样本也用上，这样 variance 会比较小。文章把它称作 remembering / forgetting tradeoff，这是和 non-stationary 关联的。
- 与之对应的是 MAB 本身的 tradeoff，即 exploration / exploitation tradeoff。

构造方法如下：

首先把整个游戏过程分为 m 个 batch，每个 batch为

$$\mathcal{T}_j = \left\{ t : (j-1)\tilde{\Delta}_T + 1 \leq t \leq \min\left\{ j\tilde{\Delta}_T, T \right\} \right\}, \quad \text{for all } j = 1, \ldots, m, \tag{2}$$

其中 $\tilde{\Delta}_T$ 为每个 batch 的长度（最后一个 batch 不太一样）。每个 batch 中各个臂的奖励分布均值

都不变，但是每个 batch 开始的时候随机选择一个最优的臂，其均值设定为 $1/2+\epsilon$ ，其余的臂都设置为 $1/2$ 。要满足前述假设，要求 $\epsilon T/\tilde{\Delta}_T \leq V_T$ 。同时，当 $\epsilon \approx 1/\sqrt{\tilde{\Delta}_T}$ 的时候，由于臂之间的均值太过接近，因此没法有效分辨出最优的臂（之前工作的结论），因此每个 batch 会产生大约 $\epsilon\tilde{\Delta}_T \approx \sqrt{\tilde{\Delta}_T}$ 的 regret。把各个 batch 加起来，能产生一个 $T/\sqrt{\tilde{\Delta}_T}$ 的 regret。环境通过选择一个合适的 $\tilde{\Delta}_T$ 使得 regret 最大，即

$$\max T/\sqrt{\tilde{\Delta}_T},\ \ s.t.\ \epsilon\frac{T}{\tilde{\Delta}_T} \leq V_T$$

观察到

$$\epsilon\frac{T}{\tilde{\Delta}_T} \approx \frac{T}{(\tilde{\Delta}_T)^{3/2}} \leq V_T \Rightarrow \tilde{\Delta}_T \geq (T/V_T)^{2/3} \Rightarrow \mathcal{R} = \frac{T}{\sqrt{\tilde{\Delta}_T}} = (V_T)^{1/3}(T)^{2/3}$$

可以得到最后的结论。

4. 算法

Rexp3. Inputs: a positive number γ, and a batch size Δ_T.

1. Set batch index $j = 1$
2. Repeat while $j \leq \lceil T/\Delta_T \rceil$:
   (a) Set $\tau = (j-1)\,\Delta_T$
   (b) Initialization: for any $k \in \mathcal{K}$ set $w_t^k = 1$
   (c) Repeat for $t = \tau+1, \ldots, \min\{T, \tau+\Delta_T\}$:
   - For each $k \in \mathcal{K}$, set

$$p_t^k \;=\; (1-\gamma)\,\frac{w_t^k}{\sum_{k'=1}^{K} w_t^{k'}} + \frac{\gamma}{K}$$

   - Draw an arm k' from $\mathcal{K}$ according to the distribution $\{p_t^k\}_{k=1}^{K}$
   - Receive a reward $X_t^{k'}$
   - For k' set $\hat{X}_t^{k'} = X_t^{k'}/p_t^{k'}$, and for any $k \neq k'$ set $\hat{X}_t^k = 0$. For all $k \in \mathcal{K}$ update:

$$w_{t+1}^k \;=\; w_t^k \exp\left\{\frac{\gamma \hat{X}_t^k}{K}\right\}$$

   (d) Set $j = j+1$, and return to the beginning of step 2

算法把整个游戏过程分为了若干个 batch。在每个 batch 中使用已有的 Exp3 算法，来解决 exploration-exploitation tradeoff，达到 $\sqrt{T}$ 的 regret；通过调整 batch 的长度来解决 remembering-forgetting tradeoff，产生最后的结果。

5. Near-optimal bound

该算法能够达到如下 regret bound。

Theorem 2 *Let π be the Rexp3 policy with a batch size $\Delta_T = \left\lceil (K\log K)^{1/3}(T/V_T)^{2/3} \right\rceil$ and with $\gamma = \min\left\{1, \sqrt{\frac{K\log K}{(e-1)\Delta_T}}\right\}$. Then, there is some absolute constant $\bar{C}$ such that for every $T \geq 1$, $K \geq 2$, and $V_T \in \left[K^{-1}, K^{-1}T\right]$:*

$$\mathcal{R}^{\pi}(\mathcal{V}, T) \;\leq\; \bar{C}\,(K\log K \cdot V_T)^{1/3}\,T^{2/3}.$$

证明思路当然是要利用到 Exp3 的结论，注意到 Exp3 比较的是 static oracle，文章期望分析的时候 dynamic oracle，因此，需要做一个拆分。

$$\mathbb{E}^{\pi}\left[\sum_{t\in\mathcal{T}_j}(\mu_t^* - \mu_t^{\pi})\right] = \underbrace{\sum_{t\in\mathcal{T}_j}\mu_t^* - \mathbb{E}\left[\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j}X_t^k\right\}\right]}_{J_{1,j}} + \underbrace{\mathbb{E}\left[\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j}X_t^k\right\}\right] - \mathbb{E}^{\pi}\left[\sum_{t\in\mathcal{T}_j}\mu_t^{\pi}\right]}_{J_{2,j}}.$$

其中前一项可以利用均值变化缓慢的假设来 bound。

Defining $\mu_{T+1}^k = \mu_T^k$ for all $k \in \mathcal{K}$, we denote the variation in expected rewards along batch $\mathcal{T}_j$ by $V_j = \sum_{t\in\mathcal{T}_j}\max_{k\in\mathcal{K}}\left|\mu_{t+1}^k - \mu_t^k\right|$. We note that:

$$\sum_{j=1}^{m} V_j \;=\; \sum_{j=1}^{m}\sum_{t\in\mathcal{T}_j}\max_{k\in\mathcal{K}}\left|\mu_{t+1}^k - \mu_t^k\right| \;\leq\; V_T. \tag{4}$$

Let k_0 be an arm with best expected performance over $\mathcal{T}_j$: $k_0 \in \arg\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j}\mu_t^k\right\}$. Then,

$$\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j}\mu_t^k\right\} \;=\; \sum_{t\in\mathcal{T}_j}\mu_t^{k_0} \;=\; \mathbb{E}\left[\sum_{t\in\mathcal{T}_j}X_t^{k_0}\right] \;\leq\; \mathbb{E}\left[\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j}X_t^k\right\}\right], \tag{5}$$

and therefore, one has:

$$\begin{aligned} J_{1,j} \;&=\; \sum_{t\in\mathcal{T}_j}\mu_t^* - \mathbb{E}\left[\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j}X_t^k\right\}\right] \overset{(a)}{\leq} \sum_{t\in\mathcal{T}_j}\left(\mu_t^* - \mu_t^{k_0}\right) \\ &\leq\; \Delta_T \max_{t\in\mathcal{T}_j}\left\{\mu_t^* - \mu_t^{k_0}\right\} \overset{(b)}{\leq}\; 2V_j\Delta_T, \end{aligned} \tag{6}$$

for any $\mu \in \mathcal{V}$ and $j \in \{1, \ldots, m\}$, where (a) holds by (5) and (b) holds by the following argument: otherwise there is an epoch $t_0 \in \mathcal{T}_j$ for which $\mu_{t_0}^* - \mu_{t_0}^{k_0} > 2V_j$. Indeed, let $k_1 = \arg\max_{k\in\mathcal{K}}\mu_{t_0}^k$. In such case, for all $t \in \mathcal{T}_j$ one has $\mu_t^{k_1} \geq \mu_{t_0}^{k_1} - V_j > \mu_{t_0}^{k_0} + V_j \geq \mu_t^{k_0}$, since V_j is the maximal variation in batch $\mathcal{T}_j$. This however, contradicts the optimality of k_0 at epoch t, and thus (6) holds.

文章讲的很详细了，大家仔细看一下吧。比较关键的是 (6) 中的 (b)，反映了『某一轮中最优的 arm』相比于『一个 batch 中的 single best arm』之间的约束关系。

第二项可以直接利用已有算法的已有结论。

$$J_{2,j} = \mathbb{E}\left[\max_{k\in\mathcal{K}}\left\{\sum_{t\in\mathcal{T}_j} X_t^k\right\} - \mathbb{E}^\pi\left[\sum_{t\in\mathcal{T}_j}\mu_t^\pi\right]\right] \overset{(a)}{\leq} 2\sqrt{e-1}\sqrt{\Delta_T K\log K}, \tag{7}$$

最后，把不同的 batch 全部加和起来即可得到最后的结论。

Step 3 (Regret throughout the horizon). Summing over $m = \lceil T/\Delta_T\rceil$ batches we have:

$$\begin{aligned}
\mathcal{R}^\pi(\mathcal{V},T) &= \sup_{\mu\in\mathcal{V}}\left\{\sum_{t=1}^T \mu_t^* - \mathbb{E}^\pi\left[\sum_{t=1}^T\mu_t^\pi\right]\right\} \overset{(a)}{\leq} \sum_{j=1}^m\left(2\sqrt{e-1}\sqrt{\Delta_T K\log K} + 2V_j\Delta_T\right)\\
&\overset{(b)}{\leq} \left(\frac{T}{\Delta_T}+1\right)\cdot 2\sqrt{e-1}\sqrt{\Delta_T K\log K} + 2\Delta_T V_T.\\
&= \frac{2\sqrt{e-1}\sqrt{K\log K}\cdot T}{\sqrt{\Delta_T}} + 2\sqrt{e-1}\sqrt{\Delta_T K\log K} + 2\Delta_T V_T, \qquad (8)
\end{aligned}$$

where: (a) holds by (3), (6), and (7); and (b) follows from (4). Finally, selecting $\Delta_T = \left\lceil (K\log K)^{1/3}(T/V_T)^{2/3}\right\rceil$, we establish:

$$\begin{aligned}
\mathcal{R}^\pi(\mathcal{V},T) &\leq 2\sqrt{e-1}\,(K\log K\cdot V_T)^{1/3}T^{2/3}\\
&\quad +2\sqrt{e-1}\sqrt{\left((K\log K)^{1/3}(T/V_T)^{2/3}+1\right)K\log K}\\
&\quad +2\left((K\log K)^{1/3}(T/V_T)^{2/3}+1\right)V_T\\
&\overset{(a)}{\leq} \left(\left(2+2\sqrt{2}\right)\sqrt{e-1}+4\right)(K\log K\cdot V_T)^{1/3}T^{2/3},
\end{aligned}$$

where (a) follows from $T\geq K\geq 2$, and $V_T\in\left[K^{-1},K^{-1}T\right]$. This concludes the proof. ■

这一部分都比较直接。

6. 讨论

以上算法在选定 Δ_T 的时候（即考虑 remembering / forgetting tradeoff 的时候）需要已知 variation budget V_T 。在很多情况下，这个 budget 无法提前知道，只能靠估计。估计可能有一个误差，比如估计的是 $\hat{V}_T = T^\alpha$ 但是真实的是 $V_T = T^{\alpha+\delta}$ ，那么最后的 regret 为 $T^{2/3+\alpha/3+\delta}$ 。

发布于 2019-08-29

强化学习 (Reinforcement Learning)