SI252 Reinforcement Learning: Homework #04

Due on April 20, 2025 at 11:59 p.m.(CST)

Name: **Zhou Shouchen**
Student ID: 2021533042

Part I: Analysis of Bandit Algorithms

(1) Reproduce the proof for Regret Decomposition Lemma
(2) Given k actions, we define a preference function $H(\cdot):\{1,\dots,k\}\to\mathcal{R}$. Then for actions $x\in\{1,\dots,k\}$ and $y\in\{1,\dots,k\}$, we define a soft-max function

$$\pi(x)=\frac{e^{H(x)}}{\sum_{y=1}^{k}e^{H(y)}}$$

Please show the following result: for any action $a\in\{1,\dots,k\}$, we have

$$\frac{\partial\pi(x)}{\partial H(a)}=\pi(x)\left(\mathbb{I}_{(x=a)}-\pi(a)\right)$$

where $\mathbb{I}_A$ is an index function of events, being 1 when event A is true and being 0 otherwise.
(3) Reproduce the proof for gradient bandit algorithm.
(4) (Bouns): show the proof for the regret upper bound of UCB1 algorithm
(5) (Bonus): show the proof for the regret upper bound of Thompson sampling algorithm (Beta-Bernoulli bandit only)

Solution

(1) Let a_τ be a random variable representing to the τ-th action, and r_τ be the reward, which is also a random variable. Since a_τ is a random variable, so $Q(a_\tau)=\mathbb{E}_{r_\tau}[r_\tau|a_\tau]$ is also a random variable of a_τ, and according to Adam's rule, we can get that:

$$\mathbb{E}_{a_\tau}[Q(a_\tau)]=\mathbb{E}_{a_\tau}[\mathbb{E}_{r_\tau}[r_\tau|a_\tau]]=\mathbb{E}[r_\tau]$$

Define the total expected reward $S_t=\sum\limits_{\tau=1}^{t}Q(a_\tau)$, then we have:

$$\mathbb{E}(S_t)=\mathbb{E}\left[\sum_{\tau=1}^{t}Q(a_\tau)\right]=\sum_{\tau=1}^{t}\mathbb{E}[Q(a_\tau)]=\sum_{\tau=1}^{t}\mathbb{E}[r_\tau]=\mathbb{E}\left[\sum_{\tau=1}^{t}r_\tau\right]$$

Since for each step, an arm must be pulled, so let $\mathcal{A}$ be the action space, then $\forall\tau$, $\sum\limits_{a\in\mathcal{A}}\mathbb{I}_{a_\tau=a}=1$, thus:

$$\begin{aligned}\mathbb{E}(S_t)&=\mathbb{E}\left[\sum_{\tau=1}^{t}r_\tau\right]\\&=\mathbb{E}\left[\sum_{\tau=1}^{t}\sum_{a\in\mathcal{A}}r_\tau\mathbb{I}_{a_\tau=a}\right]\\&=\sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t}\mathbb{E}[r_\tau\mathbb{I}_{a_\tau=a}]\end{aligned}\tag{1}$$

On the other hand, for t steps, the arms are pulled totally t times, thus

$$\begin{aligned}&\sum_{\tau=1}^{t}\sum_{a\in\mathcal{A}}\mathbb{I}_{a_\tau=a}=\sum_{\tau=1}^{t}1=t\\\Rightarrow\ &\mathbb{E}\left[\sum_{\tau=1}^{t}\sum_{a\in\mathcal{A}}\mathbb{I}_{a_\tau=a}\right]=t\\\Rightarrow\ &\sum_{\tau=1}^{t}\sum_{a\in\mathcal{A}}\mathbb{E}[\mathbb{I}_{a_\tau=a}]=t\\\Rightarrow\ &\sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t}\mathbb{E}[\mathbb{I}_{a_\tau=a}]=t\end{aligned}\tag{2}$$

Define V^* be the expected reward of the best action, i.e. $V^* = \max_{a\in\mathcal{A}} Q(a)$, and according to the definition of each action's regret, we have

$$L_\tau = \mathbb{E}_{a_\tau}\left[V^* - Q(a_\tau)\right]$$

So the total regret is

$$\begin{aligned}
L_t &= \sum_{\tau=1}^{t} \mathbb{E}_{a_\tau}\left[V^* - Q(a_\tau)\right] \\
&= tV^* - \mathbb{E}\left[\sum_{\tau=1}^{t} Q(a_\tau)\right] \\
&= tV^* - \mathbb{E}(S_t) \\
&\overset{(1),(2)}{=} \sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t} \mathbb{E}\left[\mathbb{I}_{a_\tau=a}\right] V^* - \sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t} \mathbb{E}\left[r_\tau \mathbb{I}_{a_\tau=a}\right] \\
&= \sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t} \mathbb{E}\left[(V^* - r_\tau)\mathbb{I}_{a_\tau=a}\right]
\end{aligned} \tag{3}$$

Let Δ_a be the gap between the expected reward between the best action and action a, i.e. $\Delta_a = V^* - Q(a)$, then we have:

$$\begin{aligned}
&\mathbb{E}\left[(V^* - r_\tau)\mathbb{I}_{a_\tau=a}|a_\tau\right] \\
=& \mathbb{I}_{a_\tau=a}\mathbb{E}\left[(V^* - r_\tau)|a_\tau\right] \\
=& \mathbb{I}_{a_\tau=a}(V^* - Q(a_\tau)) \qquad \text{(Definition of } Q(a_\tau)) \\
=& \mathbb{I}_{a_\tau=a}(V^* - Q(a)) \\
=& \mathbb{I}_{a_\tau=a}\Delta_a
\end{aligned}$$

Using Adam's low, we can get that

$$\mathbb{E}\left[(V^* - r_\tau)\mathbb{I}_{a_\tau=a}\right] = \mathbb{E}_{a_\tau}\left[\mathbb{E}\left[(V^* - r_\tau)\mathbb{I}_{a_\tau=a}|a_\tau\right]\right] = \mathbb{E}_{a_\tau}\left[\mathbb{I}_{a_\tau=a}\Delta_a\right] \tag{4}$$

Define the number of action a is selected after t steps as $N_t(a)$, and let π be the policy, then the total regret is:

$$\begin{aligned}
L_t &\overset{(3)}{=} \sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t} \mathbb{E}\left[(V^* - r_\tau)\mathbb{I}_{a_\tau=a}\right] \\
&\overset{(4)}{=} \sum_{a\in\mathcal{A}}\sum_{\tau=1}^{t} \mathbb{E}\left[\mathbb{I}_{a_\tau=a}\Delta_a\right] \\
&= \sum_{a\in\mathcal{A}} \mathbb{E}\left[\sum_{\tau=1}^{t} \mathbb{I}_{a_\tau=a}\right]\Delta_a \\
&= \sum_{a\in\mathcal{A}} \mathbb{E}_\pi\left[N_t(a)\right]\Delta_a
\end{aligned}$$

So above all, we have proved the regret decomposition lemma:

$$L_t = \sum_{a\in\mathcal{A}} \mathbb{E}_\pi\left[N_t(a)\right]\Delta_a$$

(2) Since the distribution is the softmax function

$$\pi(x) = \frac{e^{H(x)}}{\sum_{y=1}^{k} e^{H(y)}}$$

Then we can get that

$$\begin{aligned}
\frac{\partial \pi(x)}{\partial H(a)} &= \frac{\partial \frac{e^{H(x)}}{\sum_{y=1}^{k} e^{H(y)}}}{\partial H(a)} \\
&= \frac{\frac{\partial H(x)}{\partial H(a)} \cdot \left(\sum_{y=1}^{k} e^{H(y)}\right) - e^{H(x)} \cdot e^{H(a)}}{\left(\sum_{y=1}^{k} e^{H(y)}\right)^2} \\
&= \frac{\mathbb{I}_{x=a} e^{H(x)} \cdot \left(\sum_{y=1}^{k} e^{H(y)}\right) - e^{H(x)} \cdot e^{H(a)}}{\left(\sum_{y=1}^{k} e^{H(y)}\right)^2} \\
&= \frac{e^{H(x)}}{\sum_{y=1}^{k} e^{H(y)}} \left(\mathbb{I}_{x=a} - \frac{e^{H(a)}}{\sum_{y=1}^{k} e^{H(y)}}\right) \\
&= \pi(x) \left(\mathbb{I}_{x=a} - \pi(a)\right)
\end{aligned} \tag{5}$$

So above all, we have proved that

$$\frac{\partial \pi(x)}{\partial H(a)} = \pi(x) \left(\mathbb{I}_{x=a} - \pi(a)\right)$$

(3) The objective function is max $\mathbb{E}_{R_t}[R_t]$. Using Adam's law and LOTE, we can get that:

$$\begin{aligned}
\mathbb{E}_{R_t}[R_t] &= \mathbb{E}_{A_t}[\mathbb{E}_{R_t}[R_t|A_t]] && \text{(Adam's Law)} \\
&= \sum_x \mathbb{E}_{R_t}[R_t|A_t = x] P(A_t = x) && \text{(LOTE)} \\
&= \sum_x Q(x)\pi_t(x)
\end{aligned}$$

where $Q(a) = \mathbb{E}_{R_t}[R_t|A_t = a]$ is fixed, and $\pi_t(x) \propto e^{H_t(x)}$.
Since $\sum\limits_x \pi_t(x) = 1$, so

$$\frac{\partial \sum_x \pi_t(x)}{\partial H_t(a)} = 0 \Rightarrow \sum_x \frac{\partial \pi_t(x)}{\partial H_t(a)} = 0 \Rightarrow \sum_x B_t \cdot \frac{\partial \pi_t(x)}{\partial H_t(a)} = 0 \tag{6}$$

Thus a baseline B_t, which is independent with x could be added to ensure a better convergence and lower variance. Thus the derivative of $\mathbb{E}_{R_t}[R_t]$ becomes:

$$\begin{aligned}
\frac{\partial \mathbb{E}_{R_t}[R_t]}{\partial H_t(a)} &= \sum_x Q(x) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} \\
&\overset{(6)}{=} \sum_x \left(Q(x) \frac{\partial \pi_t(A_t)}{\partial H_t(a)} - B_t \cdot \frac{\partial \pi_t(x)}{\partial H_t(a)}\right) \\
&\overset{(5)}{=} \sum_x \left(Q(x) - B_t\right) \pi_t(x) \left(\mathbb{I}_{A_t=a} - \pi_t(a)\right) \\
&= \mathbb{E}_{A_t \sim \pi_t}\left[\left(Q(A_t) - B_t\right)\left(\mathbb{I}_{A_t=a} - \pi_t(a)\right)\right]
\end{aligned}$$

According to the definition $Q(A_t) = \mathbb{E}_{R_t}[R_t|A_t]$ is a random variable, we can get that:

$$\begin{aligned}
&\mathbb{E}\left[Q(A_t)(\mathbb{I}_{A_t=a} - \pi_t(a))\right] \\
&= \mathbb{E}_{A_t \sim \pi_t}\left[\mathbb{E}_{R_t}[R_t|A_t] \left(\mathbb{I}_{A_t=a} - \pi_t(a)\right)\right] \\
&= \mathbb{E}_{A_t \sim \pi_t}\left[\mathbb{E}_{R_t}[R_t(\mathbb{I}_{A_t=a} - \pi_t(a))|A_t]\right] \\
&= \mathbb{E}_{A_t \sim \pi_t}\left[R_t(\mathbb{I}_{A_t=a} - \pi_t(a))\right] && \text{(Adam's Law)}
\end{aligned}$$

So we can maximum $\mathbb{E}[R_t]$ by maximizing $H_t(x)$, which could use the gradient ascent methods. To save computation, using the stochastic gradient ascent method, for each iteration with step size α, we have

$$\begin{aligned}
H_{t+1}(a) \leftarrow H_t(a) + \alpha \frac{\partial \mathbb{E}_{R_t}[R_t]}{\partial H_t(a)} \\
&= H_t(a) + \alpha \mathbb{E}\left[(Q(A_t) - B_t)(\mathbb{I}_{A_t=a} - \pi_t(a))\right] \\
&= H_t(a) + \alpha \mathbb{E}\left[(R_t - B_t)(\mathbb{I}_{A_t=a} - \pi_t(a))\right] \\
&= H_t(a) + \alpha (R_t - B_t)(\mathbb{I}_{A_t=a} - \pi_t(a)) \qquad \text{(Stochastic gradient ascent)}
\end{aligned}$$

Where R_t is the reward after doing the t-th action. So above all, we have shown that after each time of action, we update the value of

$$H_{t+1}(a) \leftarrow H_t(a) + \alpha (R_t - B_t)(\mathbb{I}_{A_t=a} - \pi_t(a))$$

where B_t is a baseline, it may update if we select $B_t = \bar{R}_t$.
Additionally, if the perference function is generated by a function with parameter θ, i.e. $H_t(a) \propto f_\theta(a)$, which means that the actions are generated by the policy $A_t \sim f_\theta$, then the objective funtion is to maximize $\mathbb{E}_{R_t}[R_t] = \mathbb{E}_{A_t \sim f_\theta}[Q(A_t)]$. So the gradient of the objective function is

$$\begin{aligned}
\nabla_\theta \mathbb{E}_{A_t \sim f_\theta}[Q(A_t)] &= \nabla_\theta \sum_{a \in \mathcal{A}} Q(a) f_\theta(a) \\
&= \sum_{a \in \mathcal{A}} Q(a) \nabla_\theta f_\theta(a) \\
&= \sum_{a \in \mathcal{A}} Q(a) \frac{\nabla_\theta f_\theta(a)}{f_\theta(a)} f_\theta(a) \\
&= \sum_{a \in \mathcal{A}} Q(a) \left(\nabla_\theta \log f_\theta(a)\right) f_\theta(a) \\
&= \mathbb{E}_{A_t \sim f_\theta}[Q(A_t) \nabla_\theta \log f_\theta(a)] \\
&\approx \frac{1}{n} \sum_{i=1}^{n} Q(a_i) \nabla_\theta \left(\log f_\theta(a_i)\right)
\end{aligned}$$

Where $a_1, \ldots, a_n$ are the samples for the update, and $\nabla_\theta \log f_\theta(x)$ is generally regarded as the 'score funtion' of the preference function.
(4) Suppose that we are considering a K-arm bandit problem, let a^* be the arm with maximum probability, l be any constant, then we have

$$\begin{aligned}
N_t(a) &= \sum_{\tau=1}^{t} \mathbb{I}_{a_\tau = a} \\
&= 1 + \sum_{\tau=K+1}^{t} \mathbb{I}_{(a_\tau = a, N_{\tau-1}(a) \geq l) \cup (a_\tau = a, N_{\tau-1}(a) < l)} \\
&= 1 + \sum_{\tau=K+1}^{t} \mathbb{I}_{a_\tau = a, N_{\tau-1}(a) \geq l} + \sum_{\tau=K+1}^{t} \mathbb{I}_{a_\tau = a, N_{\tau-1}(a) < l} \\
&\leq 1 + \sum_{\tau=K+1}^{t} \mathbb{I}_{a_\tau = a, N_{\tau-1}(a) \geq l} + (l-1) \qquad \text{(prove by contradiction)} \\
&= l + \sum_{\tau=K+1}^{t} \mathbb{I}_{a_\tau = a, N_{\tau-1}(a) \geq l}
\end{aligned}$$

Since the $\hat{Q}_\tau(a)$ is the average of the reward of action a after τ steps, actually only the number of steps that a is selected could update $\hat{Q}$, so we can replace it as a function of selected times $N_t(a)$, i.e. $\mu_{N_t(a)}(a)$. If the τ-th action selected action a, which means that

$$\forall a' \in \mathcal{A}, \hat{Q}_\tau(a') + \sqrt{\frac{2\log\tau}{N_\tau(a')}} \leq \hat{Q}_\tau(a) + \sqrt{\frac{2\log\tau}{N_\tau(a)}} \Rightarrow \mu_{N_\tau(a^*)}(a^*) + \sqrt{\frac{2\log\tau}{N_\tau(a^*)}} \leq \mu_{N_\tau(a)}(a) + \sqrt{\frac{2\log\tau}{N_\tau(a)}}$$

$$\begin{aligned}
\Rightarrow \qquad N_t(a) &\leq l + \sum_{\tau=K+1}^{t} \mathbb{I}_{a_\tau=a, N_{\tau-1}(a)\geq l} \\
&\leq l + \sum_{\tau=K+1}^{t} \mathbb{I}\left\{\mu_{N_\tau(a^*)}(a^*) + \sqrt{\frac{2\log\tau}{N_\tau(a^*)}} \leq \mu_{N_\tau(a)}(a) + \sqrt{\frac{2\log\tau}{N_\tau(a)}}, N_{\tau-1}(a) \geq l\right\} \\
&\leq l + \sum_{\tau=K+1}^{t} \mathbb{I}\left\{\bigcup_{1\leq s<t}\bigcup_{l\leq s'<t} \mu_s(a^*) + \sqrt{\frac{2\log s}{N_s(a^*)}} \leq \mu_{s'}(a) + \sqrt{\frac{2\log s'}{N_{s'}(a)}}\right\} \\
&\leq l + \sum_{\tau=1}^{t}\sum_{s=1}^{\tau-1}\sum_{s'=l}^{\tau-1} \mathbb{I}\left\{\mu_s(a^*) + \sqrt{\frac{2\log s}{N_s(a^*)}} \leq \mu_{s'}(a) + \sqrt{\frac{2\log s'}{N_{s'}(a)}}\right\}
\end{aligned}$$

It could be proved by contradiction that if the inequality in the indicator holds, then at least one of the following inequalities holds:

1. The optimal action's estimated value is too low: $\mu_s(a^*) \leq Q(a^*) - \sqrt{\frac{2\log t}{s}}$
2. The action a's estimated value is too high: $\mu_{s'}(a) \geq Q(a) + \sqrt{\frac{2\log t}{s'}}$
3. The action a's expoilation is too high: $Q(a^*) < Q(a) + \sqrt{\frac{2\log t}{s'}}$

Using Hoffding's Inequality, we can get that:

$$\begin{aligned}
P\left(\mu_s(a^*) \leq Q(a^*) - \sqrt{\frac{2\log t}{s}}\right) &\leq \exp\left(\frac{-2s^2\left(\sqrt{\frac{2\log t}{s}}\right)^2}{s(0-(-1)^2)}\right) \\
&= t^{-4} \\
P\left(\mu_{s'}(a) \geq Q(a) + \sqrt{\frac{2\log t}{s'}}\right) &\leq \exp\left(\frac{-2s'^2\left(\sqrt{\frac{2\log t}{s'}}\right)^2}{s'(0-(-1)^2)}\right) \\
&= t^{-4}
\end{aligned}$$

It could also be proved with contradiction that if we select $l = \left\lceil \frac{8\log t}{\Delta_a^2} \right\rceil$, then the inequality 3. must not hold using $\Delta_a = V^* - Q(a) = Q(a^*) - Q(a)$.

Combine all information above, we can get that

$$
\begin{aligned}
\mathbb{E}\left[N_t(a)\right] &\leq \mathbb{E}\left[l + \sum_{\tau=1}^{t}\sum_{s=1}^{\tau-1}\sum_{s'=l}^{\tau-1}\mathbb{I}\left\{\mu_s(a^*) + \sqrt{\frac{2\log s}{N_s(a^*)}} \leq \mu_{s'}(a) + \sqrt{\frac{2\log s'}{N_{s'}(a)}}\right\}\right] \\
&\leq l + \sum_{\tau=1}^{t}\sum_{s=1}^{\tau-1}\sum_{s'=l}^{\tau-1} P\left(\mu_s(a^*) \leq Q(a^*) - \sqrt{\frac{2\log t}{s}}\right) + P\left(\mu_{s'}(a) \geq Q(a) + \sqrt{\frac{2\log t}{s'}}\right) \\
&\leq l + 1 + \sum_{\tau=1}^{t}\sum_{s=1}^{\tau}\sum_{s'=1}^{\tau} t^{-4} + t^{-4} \\
&\leq \frac{8\log t}{\Delta_a^2} + 1 + 2\sum_{\tau=1}^{t} t\cdot t\cdot t^{-4} \\
&\leq \frac{8\log t}{\Delta_a^2} + 1 + \frac{\pi^2}{3} \qquad \left(\sum_{n=1}^{\infty}\frac{1}{n^2} = \frac{\pi}{6}\right)
\end{aligned}
$$

Thus, according to the regret decomposition lemma, we can get that:

$$
L_t = \sum_{a|\Delta_a>0}\mathbb{E}_\pi\left[N_t(a)\right]\Delta_a \leq \sum_{a|\Delta_a>0}\left(\frac{8\log t}{\Delta_a^2} + \left(1+\frac{\pi^2}{3}\right)\right)\Delta_a = 8\log t\sum_{a|\Delta_a>0}\frac{1}{\Delta_a} + \left(1+\frac{\pi^2}{3}\right)\sum_{a|\Delta_a>0}\Delta_a
$$

As $t\to\infty$, the constant form could be ignored, so above all, we have proved that the upper bound of the regret is

$$
\lim_{t\to\infty} L_t \leq 8\log t\sum_{a|\Delta_a>0}\frac{1}{\Delta_a}
$$

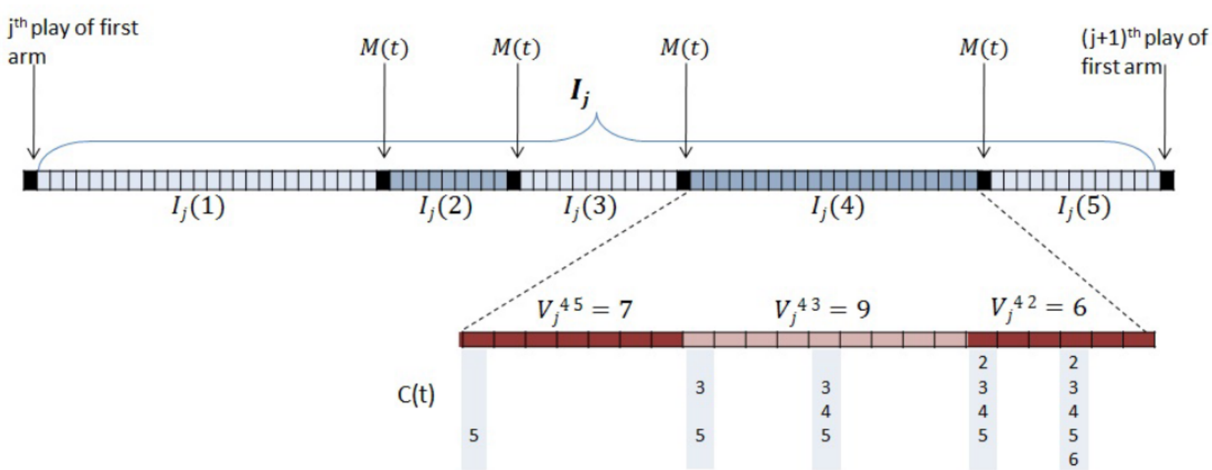
However, this is somehow a little bit different from the solution in slides(Lecture 4: Bandit Learning's Page 72):

$$
\lim_{t\to\infty} L_t \leq 8\log t\sum_{a|\Delta_a>0}\Delta_a
$$

It may be a small mistake in slides? Since the key step to prove the inequality 3. requires the selection $l = \left\lceil \frac{8\log t}{\Delta_a^2}\right\rceil$.

Reference: *Finite-time Analysis of the Multiarmed Bandit Problem*

(5) The interval between two adjacent selection for the best arm a^* from the j-th selection of a^* to the $(j+1)$-th selection I_j are devided into intervals $I_j(1)\dots, I_j(\gamma_j)$ as follows:



Where the 'first arm' in the figure representing to the best arm a^*. The saturated set $C(\tau)$ is defined as at time τ, if an arm $a \neq a^*$ has been selected more than $l_i = \dfrac{24\log t}{\Delta_a^2}$, then $a \in C(\tau)$, otherwise $a \notin C(\tau)$.
The seperate points are defined as an event $M(\tau)$. If $M(\tau)$ holds, then time τ is a seperate point. $\theta_a(\tau)$ is the sample for action a at time τ from the Beta distribution, and the seperate event is defined as

$$M(\tau) = \mathbb{I}\left\{\theta_{a^*}(\tau) \max_{a\in C(\tau)} Q(a) + \frac{\Delta_a}{2}\right\}$$

The seperate points seperate the trials into intervals $I_j(1), \ldots, I_j(\gamma_j)$. Where γ_j is the number of intervals. Since a saturated arm a can be played at step τ only if $\theta_a(\tau) > \theta_{a^*}(\tau)$. Saturated arm a can be played at a time step $\tau \notin I_j(\ell), \forall \ell, j$ (i.e., at a time step τ where $M(\tau)$ holds) only if $\theta_a(\tau) > Q(a) + \dfrac{\Delta_a}{2}$. Thus an event $E(\tau)$ is defined as

$$E(\tau) = \mathbb{I}\left\{\theta_a(\tau) \in \left[Q(a) - \frac{\Delta_a}{2}, Q(a) + \frac{\Delta_a}{2}\right], \forall a \in C(\tau)\right\}$$

So the number of plays of saturated arms in interval I_j is at most

$$\sum_{\ell=1}^{\gamma_j+1} |I_j(\ell)| + \sum_{\tau\in I_j} I(\overline{E(\tau)})$$

Also, define the number of saturated arm is played at each time step: let $V_j^{\ell,a}$ denote the number of steps in $I_j(\ell)$, for which a is the best saturated arm:

$$V_j^{\ell,a} = \left|\left\{t \in I_j(\ell) : Q(a) = \max_{a'\in C(\tau)} Q(a')\right\}\right|$$

And it could be proved that the expected regret due to playing saturated arms in interval I_j is bounded as

$$\begin{aligned}
\mathbb{E}\left[\mathcal{R}(I_j)\right] &\leq \mathbb{E}\left[\sum_{\ell=1}^{\gamma_j+1} \sum_{a|\Delta_a>0} 3\Delta_a V_k^{\ell,a}\right] + 2\mathbb{E}\left[\sum_{\tau\in I_j} \mathbb{I}(\overline{E_\tau})\right] \\
&\leq \mathbb{E}\left[\sum_{\ell=1}^{\gamma_j+1} |I_j(\ell)| \cdot \Delta_{\max}\right] + 2\sum_{\tau} P(\overline{E_\tau})
\end{aligned}$$

Where $\Delta_{\max} = \max\limits_{a|\Delta_a>0} \Delta_a$ and $\Delta_{\min} = \min\limits_{a|\Delta_a>0} \Delta_a$. And N is the number of arms of the bandit.
As $t \to \infty$, the constant form could be ignored, so above all, we have proved that the upper bound of the regret is

$$\begin{aligned}
\lim_{t\to\infty} L_t &= \sum_j \mathbb{E}\left[\mathcal{R}(I_j)\right] \\
&\leq 1152\log t\left(\sum_{a|\Delta_a>0} \frac{1}{\Delta_a^2}\right)^2 + 288\log t \sum_{a|\Delta_a>0} \frac{1}{\Delta_a^2} + 48\log t \sum_{a|\Delta_a>0} \frac{1}{\Delta_a} + 192N \sum_{a|\Delta_a>0} \frac{1}{\Delta_a^2} + 96(N-1) + 8(N-1) \\
&\leq \log t \cdot \frac{\Delta_{\max}}{\Delta_{\min}^3} \cdot \left(\sum_{a|\Delta_a>0} \frac{1}{\Delta_a^2}\right)^2 \qquad \text{(Ignore constants)} \\
\Rightarrow \lim_{t\to\infty} L_t &\leq \log t \cdot \frac{\Delta_{\max}}{\Delta_{\min}^3} \cdot \left(\sum_{a|\Delta_a>0} \frac{1}{\Delta_a^2}\right)^2
\end{aligned}$$

Reference: *Analysis of Thompson Sampling for the Multi-armed Bandit Problem*

Part II: Performance Evaluation of Classical Bandit Algorithms

You are required to use the Jupyter Notebook (Formerly known as the IPython Notebook) to submit your work.

- **Basic Setting**

  We consider a time-slotted bandit system ($t = 0, 1, 2, \ldots$) with three arms. We denote the arm set as $\{1, 2, 3\}$. Pulling each arm $j (j \in \{1, 2, 3\})$ will obtain a reward r_j, which satisfies a Bernoulli distribution with mean θ_j ($\text{Bern}(\theta_j)$), i.e.,

  $$r_j = \begin{cases} 1, & \text{w.p. } \theta_j \\ 0, & \text{w.p. } 1 - \theta_j \end{cases}$$

  where θ_j are parameters within $(0, 1)$ for $j \in \{1, 2, 3\}$.

  Now we run this bandit system for $N (N \gg 3)$ time slots. At each time slot t, we choose one and only one arm from these three arms, which we denote as $I(t) \in \{1, 2, 3\}$. Then we pull the arm $I(t)$ and obtain a reward $r_{I(t)}$. Our objective is to find an optimal policy to choose an arm $I(t)$ at each time slot t such that the expectation of the aggregated reward is maximized, i.e.,

  $$\max_{I(t), t=1, \ldots N} \mathbb{E}\left[\sum_{t=1}^{N} r_{r(t)}\right]$$

  If we know the values of $\theta_j, j \in \{1, 2, 3\}$, this problem is trivial. Since $r_{T(t)} \sim \text{Bern}\left(\theta_{I(t)}\right)$,

  $$\mathbb{E}\left[\sum_{t=1}^{N} r_{I(t)}\right] = \sum_{t=1}^{N} \mathbb{E}\left[r_{I(t)}\right] = \sum_{t=1}^{N} \theta_{I(t)}$$

  Let $I(t) = I^* = \underset{j \in \{1,2,3\}}{\arg\max} \theta_j$ for $t = 1, 2, \ldots, N$, then

  $$\max_{I(t), t=1, \ldots, N} \mathbb{E}\left[\sum_{i=1}^{N} r_{I(t)}\right] = N \cdot \theta_{I^*}$$

  However, in reality, we do not know the values of $\theta_j, j \in \{1, 2, 3\}$. We need to estimate the values θ_j via empirical samples, and then make the decisions at each time slot. Next we introduce three classical bandit algorithms: ϵ-greedy, UCB and Thompson sampling.

- **Bandit Algorithms**

  1. ϵ-greedy Algorithm ($0 \leq \epsilon \leq 1$)

Algorithm 1 ϵ-greedy Algorithm

Initialize $\hat{\theta}(j) \leftarrow 0, \text{count}(j) \leftarrow 0, j \in \{1, 2, 3\}$

1: **for** $t = 1, 2, 3, \ldots, N$ **do**

2: $\quad I(t) \leftarrow \begin{cases} \underset{j \in \{1,2,3\}}{\text{argmax}}\, \hat{\theta}_j, & \text{w.p. } 1 - \epsilon \\ \text{randomly choose from} \{1, 2, 3\}, & \text{w.p. } \epsilon \end{cases}$

3: $\quad \text{count}(I(t)) \leftarrow \text{count}(I(t)) + 1$

4: $\quad \hat{\theta}(I(t)) \leftarrow \hat{\theta}(I(t)) + \frac{1}{\text{count}(I(t))}\left[r_{I(t)} - \hat{\theta}(I(t))\right]$

5: **end for**

  2. UCB (Upper Confidence Bound) Algorithm

  Note: c is a positive constant with a default value of 1.

Algorithm 2 UCB Algorithm

Initialize $\hat{\theta}(j) \leftarrow 0, \text{count}(j) \leftarrow 0, j \in \{1, 2, 3\}$

for $t = 1, 2, 3$ **do**
  $I(t) \leftarrow t$
  $\text{count}(I(t)) \leftarrow 1$
  $\hat{\theta}(I(t)) \leftarrow r_{I(t)}$
end for
for $t = 4, \ldots, N$ **do**
  $$I(t) \leftarrow \underset{j \in \{1,2,3\}}{\text{argmax}} \left(\hat{\theta}(j) + c \cdot \sqrt{\frac{2\log(t)}{\text{count}(j)}} \right)$$
  $\text{count}(I(t)) \leftarrow \text{count}(I(t)) + 1$
  $\hat{\theta}(I(t)) \leftarrow \hat{\theta}(I(t)) + \frac{1}{\text{count}(I(t))} \left[r_{I(t)} - \hat{\theta}(I(t)) \right]$
end for

3. Thompson sampling (TS) Algorithm
Recall that $\theta_j, j \in \{1, 2, 3\}$, are unknown parameters over (0, 1). From the Bayesian perspective, we assume their priors are Beta distributions with given parameters (α_j, β_j).

Algorithm 3 TS Algorithm

Initialize Beta parameter $(\alpha_j, \beta_j), j \in \{1, 2, 3\}$

for $t = 1, 2, 3, \ldots, N$ **do**
  # Sample method
  for $j \in \{1, 2, 3\}$ **do**
    Sample $\hat{\theta}(j) \sim \text{Beta}(\alpha_j, \beta_j)$
  end for
  # Select and pull the arm
  $I(t) \leftarrow \underset{j \in \{1,2,3\}}{\text{argmax}}\, \hat{\theta}(j)$
  # Update the distribution
  $\alpha_{I(t)} \leftarrow \alpha_{I(t)} + r_{I(t)}$
  $\beta_{I(t)} \leftarrow \beta_{I(t)} + 1 - r_{I(t)}$
end for

- **Simulation**
(1) Now suppose we obtain the Bernoulli distribution parameters from an oracle, which are shown in the following table below. Choose $N = 10000$ and compute the theoretically maximized expectation of aggregate rewards over N time slots. We call it the oracle value. Note that these parameters $\theta_j, j = 1, 2, 3$ and oracle values are unknown to all bandit algorithms.

Arm j	1	2	3
θ_j	0.9	0.8	0.7

(2) Implement classical bandit algorithms with following settings:
- $N = 5000$
- ϵ-greedy with $\epsilon = 0.1, 0.5, 0.9$.
- UCB with $c = 1, 5, 10$.
- Thompson Sampling with
$\{(\alpha_1, \beta_1) = (1, 1), (\alpha_2, \beta_2) = (1, 1), (\alpha_3, \beta_3) = (1, 1)\}$ and

$\{(\alpha_1, \beta_1) = (601, 401), (\alpha_2, \beta_2) = (401, 601), (\alpha_3, \beta_3) = (2, 3)\}$
- Gradient bandit with baseline $b = 0, 0.8, 5, 20$.
- Parameterized gradient bandit with constant parameter $\beta = 0.2, 1, 2, 5$
- Parameterized gradient bandit with time-varying parameters (you need to design a time-varying rule)

(3) Each experiment lasts for $N = 5000$ turns, and we rum each experiment 1000 times. Results are averaged over these 1000 independent rums.

(4) Please report three performance metrics
- The total regret accumulated over the experiment.
- The regret as a function of time.
- The percentage of plays in which the optimal arm is pulled.

(5) Compute the gaps between the algorithm outputs and the oracle value. Compare the numerical results of ϵ-greedy, UCB, Thompson Sampling and gradient bandit. Which one is the best? Then discuss the impacts of ϵ, C, and α_j, β_j, b, and β respectively.

(6) What is the role of baseline in gradient bandit algorithm? Show your answer with simulation result.

(7) Give your understanding of the exploration-exploitation trade-off in bandit algorithms.

(8) Give your understanding of the adoption of sublinear regret as the performance threshold between good bandit algorithms and bad bandit algorithms.

Solution

(1) Since each arm's parameter is oracled, so we just need to choose the arm with the largest parameter to have the maximum expectation of aggregate rewards over N time slots.
Since $\theta_1 = 0.9, \theta_2 = 0.8, \theta_3 = 0.7$, so we choose arm 1 everytime.
i.e.

$$\forall t, I(t) = I^* = \underset{j\in\{1,2,3\}}{\operatorname{argmax}}\, \theta_j = 1$$
$$\hat{\theta}_{I(t)} = \theta_1 = 0.9$$
$$r_{I(t)} \sim \text{Bern}(\theta_{I(t)}) \Rightarrow \mathbb{E}(r_{I(t)}) = \hat{\theta}_{I(t)}$$

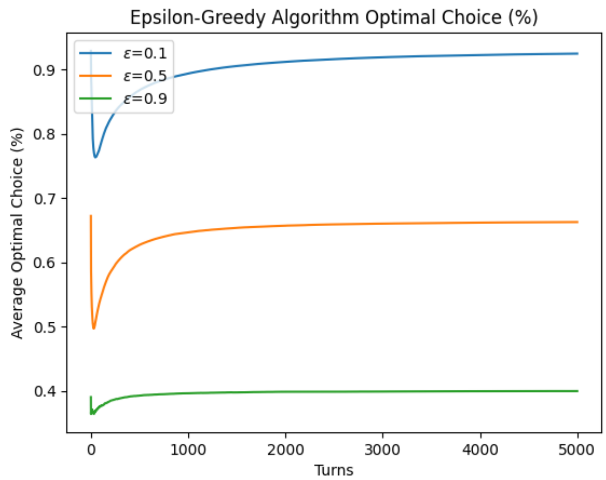
So the maximum expected value is

$$\begin{aligned}
&\max_{I(t), t=1,2,\cdots,N} \mathbb{E}\left[\sum_{t=1}^{N} r_{I(t)}\right] \\
=& \max_{I(t), t=1,2,\cdots,N} \sum_{t=1}^{N} \mathbb{E}\left[r_{I(t)}\right] \\
=& N \cdot \hat{\theta}_{I(t)} \\
=& 10000 \times 0.9 \\
=& 9000
\end{aligned}$$

So above all, with the given oracle parameters, the maximum expected value is 9000.
(2), (3) The implementation codes are in the 'hw4_code.ipynb' file.
(4) 1. The epsilon-greedy algorithm

parameter	Average Regret	Optimal Percentage
$\epsilon = 0.1$	54.35	92.48%
$\epsilon = 0.5$	251.34	66.30%
$\epsilon = 0.9$	449.43	39.95%

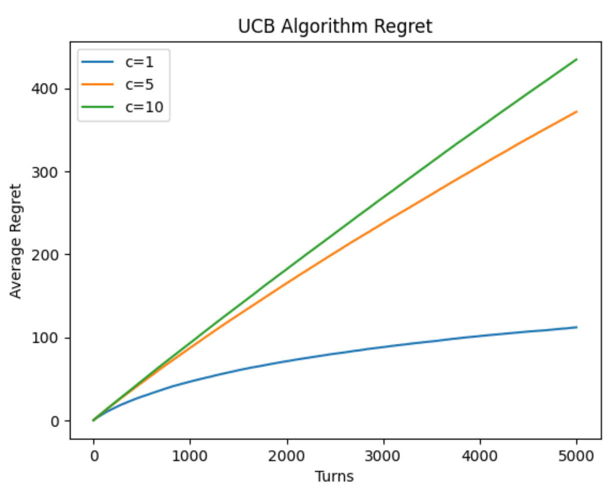
The regret and optimal percentage for ϵ-greedy algorithm as a function of time are shown in the following figures:

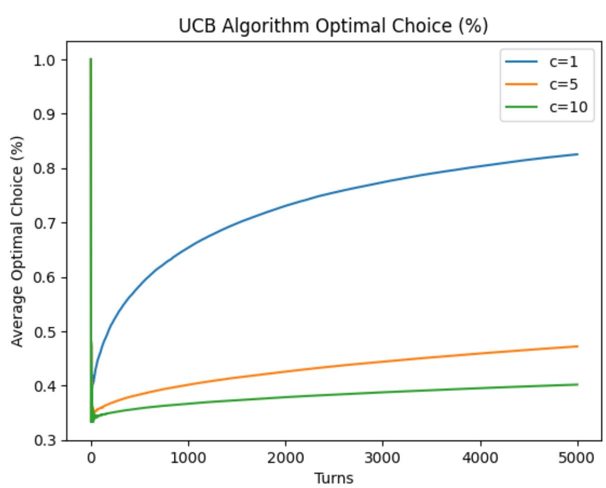


2. The UCB algorithm

parameter	Average Regret	Optimal Percentage
$c = 1$	112.75	82.51%
$c = 5$	372.65	47.20%
$c = 10$	436.05	40.18%

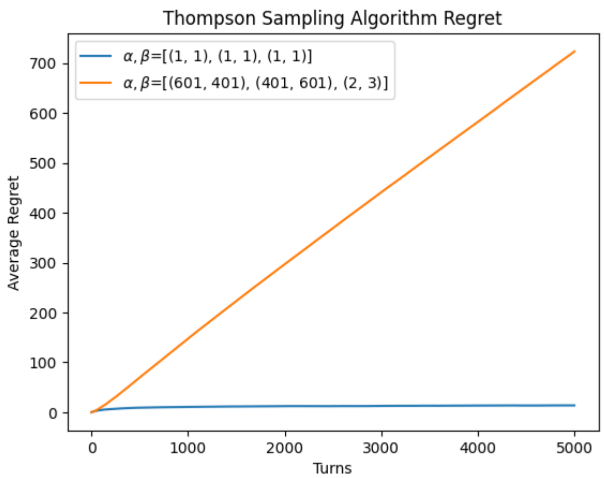
The regret and optimal percentage for UCB algorithm as a function of time are shown in the following figures:

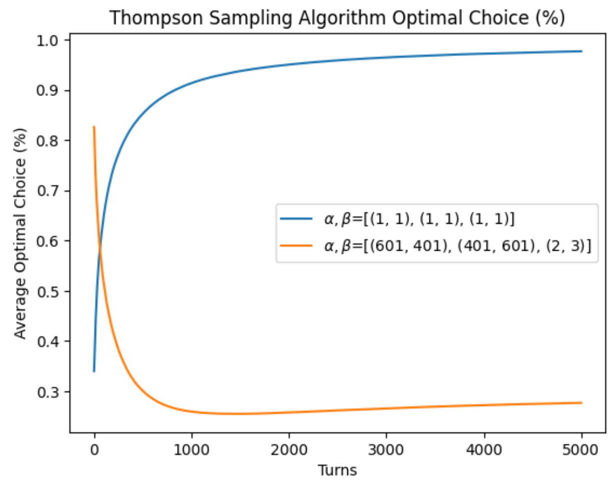




3. The Thompson Sampling algorithm

parameter	Average Regret	Optimal Percentage
$\alpha, \beta = [(1, 1), (1, 1), (1, 1)]$	14.36	97.58%
$\alpha, \beta = [(601, 401), (401, 601), (2, 3)]$	708.15	29.07%





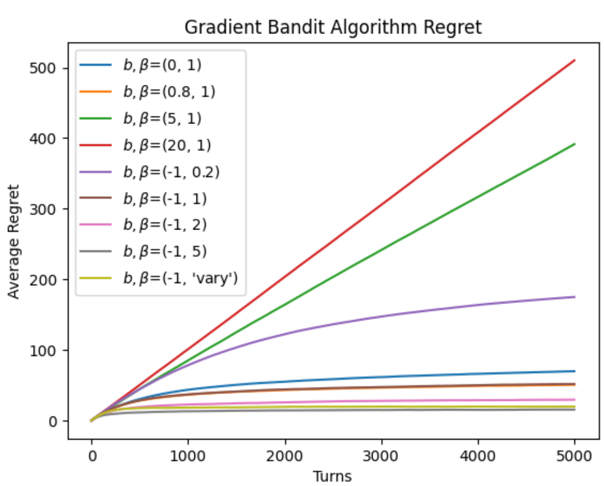
4. The Gradient Bandit algorithm

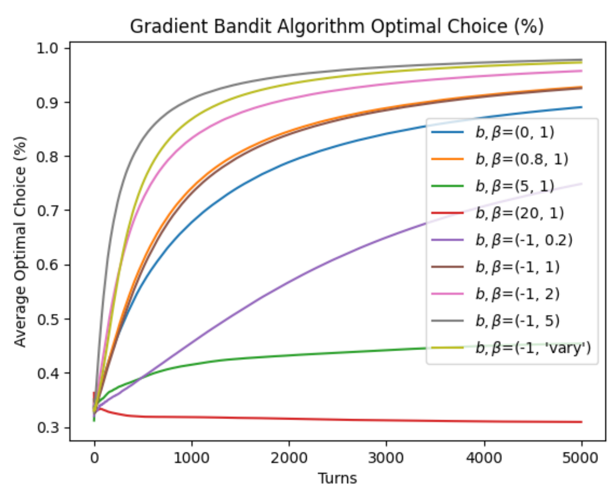
The step size is set to be $\alpha = 0.1$. For the time-varient case, to have a smooth varying, set $\beta_0 = 0.1, \beta_T = 10$:

$$\beta(t) = \beta_0 + \left(\frac{\log\left(1 + 9 \cdot \frac{t}{T}\right)}{\log(10)}\right) \cdot (\beta_T - \beta_0) = 0.1 + \left(\frac{\log\left(1 + 9 \cdot \frac{t}{T}\right)}{\log(10)}\right) \cdot (10 - 0.1)$$

And the baseline is set to be $B_t = \bar{R}_t$ is $b = -1$, otherwise it is set to be $B_t = b$.

parameter	Average Regret	Optimal Percentage
$b = 0$	71.36	88.72%
$b = 0.8$	50.37	92.72%
$b = 5$	381.22	44.73%
$b = 20$	490.65	33.18%
$\beta = 0.2$	176.42	74.77%
$\beta = 1$	49.97	92.60%
$\beta = 2$	30.06	95.67%
$\beta = 5$	16.57	97.55%
time-varying	19.22	97.22%





(5) The gap's comparison between the algorithms with different parameters is as follows:

best | second best

Algorithm	**Parameter**	**Gap**
ϵ-Greedy	$\epsilon = 0.1$	54.35
	$\epsilon = 0.5$	251.34
	$\epsilon = 0.9$	449.43
UCB	$c = 1$	112.75
	$c = 5$	372.65
	$c = 10$	436.05
TS	$\alpha, \beta = [(1,1),(1,1),(1,1)]$	14.36 (best)
	$\alpha, \beta = [(601,401),(401,601),(2,3)]$	708.15
Gradient	$b = 0$	71.36
	$b = 0.8$	50.37
	$b = 5$	381.22
	$b = 20$	490.65
	$\beta = 0.2$	176.42
	$\beta = 1$	49.97
	$\beta = 2$	30.06
	$\beta = 5$	16.57 (second best)
	time-varying	19.22

Comparing all rewards among the experiments we have done, we could find that the Thompson Sampling algorithm with parameter $\alpha, \beta = [(1,1),(1,1),(1,1)]$ is the best one, and gradient bandit algorithm with parameter $\beta = 5$ is the second best one.
Then we can discuss the impacts of ϵ, c, and α_j, β_j, b, β respectively.

- ϵ-greedy algorithm
  We have a parameter ϵ, in our experiments, lower ϵ has a lower regret, however, it is still linear to the time.

- the UCB algorithm
  We choose the arm by the metric $\hat{\theta}(j) + c \cdot \sqrt{\frac{2\log(t)}{\text{count}(j)}}$. In our experiments, we could find that the regret of the UCB algorithm with $c = 1$ is the lowest. And the regret is sublinear to the time. Larger c has a higher regret, and the regret looks like linear to the time when $c = 5, 10$.

- the Thompson Sampling algorithm
  In the Beta-Bernoulli Thompson Sampling algorithm, we have a parameter α and β to decide $\hat{\theta}_j$ as it $\sim \text{Beta}(\alpha, \beta)$, in our experiences, we could discover that $\alpha, \beta = [(1,1),(1,1),(1,1)]$ is the best one. This is because the prior brief is closer to the true distribution of the reward, and it is not in a wrong direction. $\alpha, \beta = [(601,401),(401,601),(2,3)]$ could be regarded as doing some of the experiments, but the experiment is in a wrong distribution with the oracle distribution, so a worse prior lead to a worse performance.

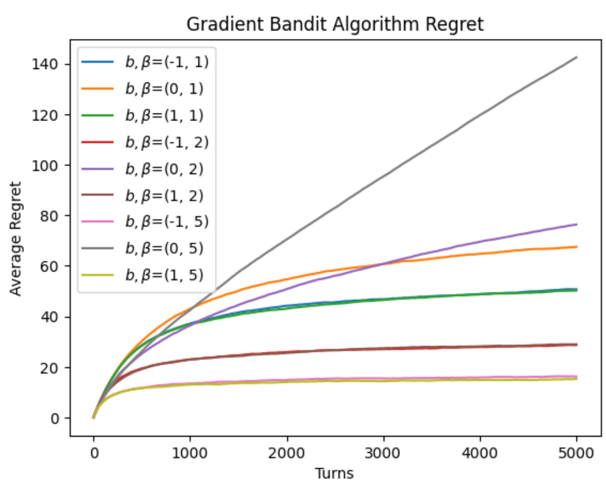
- the Gradient Bandit algorithm
  The motivation of introducting baseline in the policy gradient algorithm is mainly used to reduce variance and accelerate convergence to make the algorithm more stable. The parameter β adjust the weights for the perference of the arm.

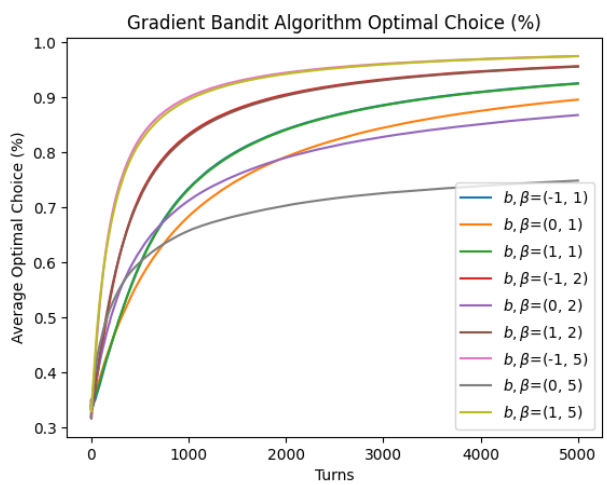
In ours experiments, baseline $b = 0.8$ have a better performance all bs. This is because $b = 0.8$ is closer to the best arm's oracle distribution $\theta_1 = 0.9$, so it has a better performance. And larger β means more important to the perference of the arm so $\beta = 5$ has a better performance. And in theoretically, β should be small at beginning for exploration, and be larger for exploitation latter. However, in our experiments, the time-varying β is worse than $\beta = 5$. This may because the time-varying function is not well selected.

(6) The baseline in the gradient bandit algorithm is a reference point for the reward. It is used to reduce the variance of the reward and make the algorithm more stable. We can test the performance of the gradient bandit algorithm with different baseline and different β. $b = -1$ representing the baseline is the average reward $\bar{R}_t$.

best second best

parameter	Average Regret	Optimal Percentage
$b = -1, \beta = 1$	51.38	92.57%
$b = 0, \beta = 1$	73.54	88.41%
$b = 1, \beta = 1$	50.44	92.65%
$b = -1, \beta = 2$	29.04	95.66%
$b = 0, \beta = 2$	76.34	86.75%
$b = 1, \beta = 2$	30.16	95.75%
$b = -1, \beta = 5$	16.07	97.57%
$b = 0, \beta = 5$	150.11	73.57%
$b = 1, \beta = 5$	16.59	97.33%





And we can see that β is the most significant parameter. Setting the baseline to be the average reward $\bar{R}_t$ has a better performance in ours experiments. If the baseline set not suitable, even a better β could lead to a worse performance. For example, in ours experiment $b = 0, \beta = 5$ has a much larger average regret. And with baseline $B_t = \bar{R}_t, \beta = 5$ has a much better performance. $b = 0$ means without baseline, and we can see that in all settings of β, $b = 0$, i.e. without baseline always has the highest regret.

(7) Exploration-Exploitation is a basic and popular topic in Reinforcement Learning. And it is also a very important topic in the bandit algorithm. At the beginning of playing with the bandit, we know nothing about the bandit. So we have to explore the bandit, gaining data from the previous decisions and feedbacks. Then after getting some information about the bandit, we can exploit them.

However, there must have a trade-off between exploration and exploitation. That is we have no idea how many times to explore, and when to start to exploit. If we always explore, we will never exploit to obtain better reward. And if we always exploit, we do not explore, and we may miss the best decision, go along the wrong way further and further. So we need to find a balance between exploration and exploitation. And this is the exploration-exploitation trade-off.

- ϵ-greedy algorithm
  We have a parameter ϵ to decide the probability of exploration and exploitation. If we set ϵ to be a small value, we will have a high probability to exploit the best arm we have found. And we will have a low probability to explore other arms. If we set ϵ to be a large value, we will have a high probability to explore other arms. And we will have a low probability to exploit the best arm we have found. So theoretically, the most suitable ϵ is time-varying: a larger value at the beginnig, and gradually decrease to a small value.

- the UCB algorithm
  We choose the arm by the metric $\hat{\theta}(j) + c \cdot \sqrt{\frac{2\log(t)}{\text{count}(j)}}$. Where $\hat{\theta}(j)$ could be regard as exploitation, and $c \cdot \sqrt{\frac{2\log(t)}{\text{count}(j)}}$ could be regard as exploration.

  As for c, it is the parameter the decribe the degree of exploration. As c increase, It turns to be more likely to explore. Correspondingly, as c decrease, it more likely to exploitation.

  So theoretically, the most suitable c is time-varying: a larger value at the beginnig, and gradually decrease to a small value.

- the Thompson Sampling algorithm
  In the Beta-Bernoulli Thompson Sampling algorithm, we have a parameter α and β to decide $\hat{\theta}_j \sim \text{Beta}(\alpha, \beta)$. The Thompson Sampling algorithm is somehow more like a Bayesian method. We have a prior belief of the distribution of the reward. And we update the belief according to the reward we get.

  The initial parameters α and β are the prior belief of the distribution of the reward. And we update the parameters according to the reward we get with the Beta-Binomial conjugate. If we set α_j and β_j to be a small value, we can regard that the prior tests time are less. i.e. with less prior tests, also less exploration. If we set α_j and β_j to be a large value, we can regard that the prior tests time are more. i.e. with more prior tests, also more exploration.

- the Gradient Bandit algorithm
  The parameter β adjust the weights for the perference of the arm, smaller β means more exploration, larger β means more exploitation. In our experiments, we could find that the gradient bandit algorithm with $\beta = 5$ is the best one. In a suitable range, larger β has a better performance.

  So theoretically, the most suitable β is time-varying: a larger value at the beginnig, and gradually decrease to a small value.

(8) Intuitively, linear regret means that the algorithm failed to learn effectively: its average regret per step is a constant, instread of decreasing over time, indicating that the algorithm cannot improve decisions through historical experience.

But for sublinear regret, which means that $\lim\limits_{t\to+\infty} \frac{L_t}{t} = 0$, where L_t is the total regret at time t. Indicating that the average regret per step of the algorithm tends towards zero. The algorithms gradually learn to choose actions that are close to optimal, reflecting effective exploration and utilization of the environment. Theoretically, it is proved that the asymptotic total regret's lower bound has

$$\lim_{t\to+\infty} L_t \geq \log t \sum_{a|\Delta_a>0} \frac{\Delta_a}{KL(\mathcal{R}^a \| \mathcal{R}^{a*})}$$

Which is a sublinear regret. So if a algorithm has a sublinear regret, it reach the theoretical asymptotic optimal, and it is a good algorithm.

Part III: Design for Modern Bandit Algorithms

(1) design a UCB style algorithm for graph bandits in the format of pseudocode, and explain how you utilize the additional structure information.
(2) design a UCB style algorithm for dueling bandits in the format of pseudocode, and explain how you utilize the additional structure information.
(3) design a UCB style algorithm for combinatorial bandits in the format of pseudocode, and explain how you utilize the additional structure information.
(4) design a UCB style algorithm for neural bandits in the format of pseudocode, and explain how you utilize the additional structure information.

Solution

(1) Graph bandit is a special case of multi-armed bandit problem, where the reward between arms has the correlation, which could be represented as a graph structure.
Base on the assumption that the graph's structure is known, i.e. the degree matrix D and adjacency matrix A are known, thus the Laplacian matrix $L = D - A$ is also known, and it is a positive semi-definite matrix, which could be diagonalized as $L = Q\Lambda Q^\top$. Thus each node has a spectral embedding x_v in the space of Q, which is the corresponding eigenvector of the Laplacian matrix. The eigenvalues for Λ is defined as $0 < \lambda = \lambda_1 \le \lambda_2 \le \cdots \le \lambda_N$.
The reward function is set to be a linear combination of the eigenvectors. For a set of weights α, the reward is defined as

$$f_\alpha(v) = \sum_{k=1}^{N} \alpha_k (q_k)_v = x_v^\top \alpha$$

It is as constructed that for each time t, after selecting the node v, the recieved reward r_t is affected by a noise ϵ_t, i,e,

$$r_t = x_v^\top \alpha^* + \epsilon_t$$

Where the noise has a upper bound R.
Thus, for each time, we the following UCB algorithm for graph bandit to select v_t, which represents to the x_{v_t}-th row of Q.
Reference: *Spectral Bandits for Smooth Graph Functions with Applications in Recommender Systems.*
The pseudocode of the UCB algorithm for graph bandits is as follows:

Algorithm 4 UCB Algorithm for graph bandits

Input: Number of vertices N, total rounds T. Spectral basis $\{\Lambda_L, Q\}$ of graph Laplacian L. Cconfidence parameters δ. Upper bounds: R (noise), C (norm of true parameter vector)

1: $\Lambda \leftarrow \Lambda_L + \lambda I$
2: $d \leftarrow \max\{d : (d-1)\lambda_d \le T/\log(1+T/\lambda)\}$ ▷ Select effective dimension
3: **for** $t = 1$ **to** T **do**
4: $X_t \leftarrow [x_1^\top, x_2^\top, \dots, x_{t-1}^\top]^\top$ ▷ Past feature vectors
5: $r \leftarrow [r_1, r_2, \dots, r_{t-1}]^\top$ ▷ Past observed rewards
6: $V_t \leftarrow X_t^\top X_t + \Lambda$ ▷ Design matrix with regularization
7: $\hat{\alpha}_t \leftarrow V_t^{-1} X_t^\top r$ ▷ Ridge regression estimator
8: $c_t \leftarrow 2R\sqrt{d\log(1+t/\lambda) + 2\log(1/\delta)} + C$ ▷ Confidence radius
9: Choose node v_t as:

$$v_t \leftarrow \underset{v\in\{1,\dots,N\}}{\operatorname{argmax}} \left(f_{\hat{\alpha}}(v) + c_t \cdot \|x_v\|_{V_t^{-1}}\right)$$

▷ UCB rule using spectral embedding
10: Observe reward r_t at node v_t
11: **end for**

---

(2) Dueling bandit is a special case of multi-armed bandit problem, instead of directly getting the reward, but recieve the pairwise comparison between two arms.
So we can use a matrix W to record the recieved pairwise comparison between two arms. Suppose we have K arms, since it may have cases that arms are not compared, so define $\frac{x}{0} = 1$.
Reference: *Relative Upper Condence Bound for the K-Armed Dueling Bandit Problem.*
The pseudocode of the UCB algorithm for dueling bandits is as follows:

Algorithm 5 UCB Algorithm for graph bandits dueling bandits

Input: c, T, K

$\mathbf{W} = [w_{ij}] \leftarrow \mathbf{0}_{K\times K}$ ▷ Matrix of wins: w_{ij} is the number of times a_i beat a_j
for $t = 1, \ldots, T$ **do**
  Compute matrix $\mathbf{U} = [u_{ij}]$ as: $u_{ij} \leftarrow \begin{cases} \frac{w_{ij}}{w_{ij}+w_{ji}} + \sqrt{\frac{c\ln t}{w_{ij}+w_{ji}}} & \text{if } i \neq j \\ \frac{1}{2} & \text{if } i = j \end{cases}$
  Select arm a_c satisfying $u_{cj} \geq \frac{1}{2}, \forall j \neq c$; if no such c exists, pick c uniformly at random
  Select opponent a_d as $d \leftarrow \operatorname*{argmax}_{j\neq c} u_{jc}$ ▷ Arm with highest UCB against a_c
  Compare arms a_c and a_d, observe winner
  if a_c wins **then**
    $w_{cd} \leftarrow w_{cd} + 1$
  else
    $w_{dc} \leftarrow w_{dc} + 1$
  end if
end for
Return: Arm a_c with the largest number of arms j such that: $\frac{w_{cj}}{w_{cj}+w_{jc}} > \frac{1}{2}$

---

(3) Combinatorial bandit is a special case of multi-armed bandit problem, for each time, the selection is a set of arms, and the reward is a function of the selected arms.
Each time we estimate each arm's reward $\bar{\mu}_i$, and solve a combinatorial problem by selecting the best combination from all valid selections $\mathcal{S}$, which is named as 'Oracle'.
Reference: *Combinatorial Multi-Armed Bandit and Its Extension to Probabilistically Triggered Arms.*
The pseudocode of the UCB algorithm for combinatorial bandits is as follows:

Algorithm 6 Combinatorial UCB (CUCB)

Input: Total number of base arms m, oracle access to action set $\mathcal{S} \subseteq 2^{[m]}$

for each base arm $i \in [m]$ **do**
  $T_i \leftarrow 0$ ▷ Number of times base arm i has been played
  $\hat{\mu}_i \leftarrow 1$ ▷ Initial empirical mean reward for base arm i
end for
for do$t = 1, \cdots, T$
  $t \leftarrow t + 1$
  for each base arm $i \in [m]$ **do**
    $\bar{\mu}_i \leftarrow \min\left\{\hat{\mu}_i + \sqrt{\frac{3\ln t}{2T_i}},\ 1\right\}$
  end for
  $S_t \leftarrow \text{Oracle}(\bar{\mu}_1, \bar{\mu}_2, \ldots, \bar{\mu}_m)$ ▷ Solve combinatorial problem using optimistic estimates
  Play action S_t, observe rewards $X_{i,t}$ for each $i \in S_t$
  for each $i \in S_t$ **do**
    $T_i \leftarrow T_i + 1$
    $\hat{\mu}_i \leftarrow \hat{\mu}_i + \frac{1}{T_i}(X_{i,t} - \hat{\mu}_i)$ ▷ Update empirical mean
  end for
end for

---

(4) Neural bandit is a special case of multi-armed bandit problem, each time we can recieve an arm's contextual information $\{x_{t,a}\}_{a=1}^{K}$ with total K arms, and use neural network to predict the reward $\hat{r}_{t,a}$. As for the exploration part of UCB, γ_t is computed through a theoritial bound with the Neural Tangent Kernel, which could be detailedly found in the paper. Z could be regarded as the covariance matrix of the parameters of the neural network. $g(x_t;\theta)$ is the gradient of the neural network $f(x_t;\theta)$. And after each steps, the neural network is updated through a J step's gradient descent with step size η with criterion of minimizing the L2-loss between the estimated reward and the true reward, with regularization term $m\lambda\|\theta-\theta_0\|^2$. Additionally, m is the width of the network to ensure the NTK is well-defined.
Reference: *Neural contextual bandits with ucb-based exploration.*
The pseudocode of the UCB algorithm for Neural bandits is as follows:

Algorithm 7 NeuralUCB

Input: Number of rounds T, regularization parameter λ, exploration parameter ν, confidence parameter δ, norm bound S, step size η, number of gradient steps J, network width m, depth L

Randomly initialize network parameters θ_0
Initialize $Z_0 \leftarrow \lambda I$
for $t = 1$ to T **do**
  Observe context vectors $\{x_{t,a}\}_{a=1}^{K}$
  for $a = 1$ to K **do**
    Compute predicted reward: $\hat{r}_{t,a} \leftarrow f(x_{t,a};\theta_{t-1})$
    Compute uncertainty:

$$U_{t,a} \leftarrow \hat{r}_{t,a} + \gamma_{t-1} \cdot \sqrt{\frac{1}{m} g(x_{t,a};\theta_{t-1})^\top Z_{t-1}^{-1} g(x_{t,a};\theta_{t-1})}$$

  end for
  Select action: $a_t \leftarrow \operatorname*{argmax}_{a\in[K]} U_{t,a}$
  Play arm a_t and observe reward r_{t,a_t}
  Update covariance matrix:

$$Z_t \leftarrow Z_{t-1} + \frac{1}{m} g(x_{t,a_t};\theta_{t-1}) g(x_{t,a_t};\theta_{t-1})^\top$$

  Train $\theta_t \leftarrow \mathrm{TrainNN}(\lambda, \eta, J, m, \{x_{i,a_i}\}_{i=1}^{t}, \{r_{i,a_i}\}_{i=1}^{t}, \theta_0)$
  $\gamma_t \leftarrow$ computed theoritial bound based on NTK
end for