RovingBandit Algorithm Reference

This document provides a detailed mathematical exposition of the multi-armed bandit algorithms implemented in `rovingbandit`.

1. Problem Setup and Notation

We consider a stochastic multi-armed bandit problem with K arms (actions), indexed by $i \in \{0, \dots, K-1\}$.

- **Time**: The process proceeds in discrete time steps $t = 1, 2, \dots, T$.
- **Rewards**: At each step t, the agent selects an arm $A_t \in \{0, \dots, K-1\}$ and observes a reward r_t.
- **Distribution**: For each arm i, rewards are drawn i.i.d. from an unknown probability distribution ν_i with mean $\mu_i = \mathbb{E}[r \mid A_t = i]$.
- **Optimal Arm**: Let $\mu^* = \max_i \mu_i$ be the maximal mean reward, and $i^* = \text{argmax}_i \mu_i$ be the optimal arm.
- **Gap**: The sub-optimality gap for arm i is $\Delta_i = \mu^* - \mu_i$.

Objectives

1. **Regret Minimization**: Minimize the expected cumulative regret over horizon T:

   $$R(T) = \mathbb{E}\left[\sum_{t=1}^{T} (\mu^* - \mu_{A_t})\right] = \sum_{i=0}^{K-1} \Delta_i \mathbb{E}[N_i(T)]$$

   where $N_i(T)$ is the number of times arm i was pulled up to time T.

2. **Best-Arm Identification (BAI)**: Identify the optimal arm i^* with high probability using the fewest number of samples, or maximize the probability of identifying i^* given a fixed budget.

3. **Variance Minimization**: Allocate samples {N_i}_{i=1}^K$ to minimize the variance of an estimator (e.g., Average Treatment Effect), often subject to constraints.

---

2. Regret Minimization Policies

2.1 Random Policy

Class: `RandomPolicy`

A baseline policy that selects an arm uniformly at random at each step.

$$P(A_t = i) = \frac{1}{K} \quad \forall i \{0, \dots, K-1\}$$

- **Regret**: Linear, $O(T)$.
- **Use case**: Baseline for comparison; collecting unbiased data for offline evaluation. This is an RCT.

2.2 Greedy Policy

Class: `GreedyPolicy`

Always selects the arm with the highest empirical mean reward.

$$A_t = \text{argmax}_i \hat{\mu}_{i,t-1}$$

where $\hat{\mu}_{i,t-1}$ is the average reward of arm i up to time $t-1$.

- **Regret**: Linear, $O(T)$ (fails to explore).
- **Failure Mode**: Can get stuck selecting a suboptimal arm if early samples from the optimal arm are poor.

2.3 ϵ-Greedy Policy

Class: `EpsilonGreedy`

Balances exploration and exploitation by selecting a random arm with probability ϵ and the greedy arm with probability $1-\epsilon$.

$$P(A_t = i) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{K} & \text{if } i = \text{argmax}_j \hat{\mu}_{j,t-1} \\ \frac{\epsilon}{K} & \text{otherwise} \end{cases}$$

Decaying ϵ-Greedy: Ideally, exploration should decrease as information is gathered. A common schedule is:

$$\epsilon_t = \min\left(1, \frac{C}{t}\right)$$

With properly tuned decay, asymptotic regret is $O(\log T)$.

2.4 Explore-First (EtC)

Class: `ExploreFirst`

Divided into two distinct phases:

1. **Exploration**: Pull each arm N times (total $K \times N$ steps).
2. **Exploitation**: Commit to the arm with the highest empirical mean for the remaining steps.

The parameter `exploration_fraction` (γ) determines the split. If horizon is T, we explore for γT steps.

- **Regret**: $O(T^{2/3})$ generally, or $O(\sqrt{T})$ with optimal tuning.
- **Limitation**: Requires knowledge of horizon T. Suboptimal for any specific instance compared to adaptive strategies.

2.5 UCB1 (Upper Confidence Bound)

Class: `UCB1`

Follows the principle of "Optimism in the Face of Uncertainty". It constructs a confidence interval for each arm's mean and selects the arm with the highest upper bound.

Algorithm: At time t, select:

$$A_t = \text{argmax}_i \left[\hat{\mu}_{i,t-1} + \sqrt{\frac{2 \ln t}{N_i(t-1)}} \right]$$

- **Term 1**: Exploitation (empirical mean).
- **Term 2**: Exploration (confidence width). Grows with time ($\log t$), shrinks with samples (N_i).
- **Regret**: $O(\log T)$. Guarantees near-optimality uniformly over time.
- **Reference**: Auer, P., Cesa-Bianchi, N., & Fischer, P. (2002).

2.6 Thompson Sampling

Class: `ThompsonSampling`

A Bayesian randomized algorithm. It maintains a posterior distribution over the mean reward of each arm and samples an action according to the probability that it is optimal.

Model (Bernoulli Rewards):

- Prior: $\mu_i \sim \text{Beta}(\alpha_0, \beta_0)$.
- Likelihood: $r \sim \text{Bernoulli}(\mu_i)$.
- Posterior after S_i successes and F_i failures:

$$\mu_i \mid \mathcal{H}_t \sim \text{Beta}(\alpha_0 + S_i, \beta_0 + F_i)$$

Algorithm:

1. For each arm i, sample $\theta_i \sim \text{Beta}(\alpha_{i,t}, \beta_{i,t})$.
2. Select $A_t = \text{argmax}_i \theta_i$.

- **Regret**: $O(\log T)$. Often outperforms UCB empirically.
- **Reference**: Thompson (1933); Chapelle & Li (2011).

---

3. Best-Arm Identification Policies

3.1 Top-Two Thompson Sampling (TTTS)

Class: `TopTwoThompson`

Designed to adapt Thompson Sampling for Best-Arm Identification. Standard TS allocates too many samples to the best arm and not enough to the "challengers" necessary to verify optimality.

Algorithm: At each step t:

1. Sample $\theta \sim$ Posterior. Let $I = \text{argmax}_i \theta_i$ (the "leader").
2. With probability ψ (e.g., 0.5), play arm I.
3. Else (probability $1 - \psi$), resample θ' until $\text{argmax}_j \theta'_j \neq I$. Let this challenger be J. Play arm J.

This forces the algorithm to explore the second-best arm (the most likely challenger), improving the convergence rate of the posterior ratio.

- **Reference**: Russo, D. (2016). *Simple Bayesian algorithms for best arm identification.*

---

4. Extension Policies

4.1 Budgeted UCB

Class: `BudgetedUCB`

Handles cases where arms have different costs c_i. The goal is to maximize total reward subject to a total budget B, rather than a fixed number of steps T.

Algorithm: Selects arm maximizing the "bang-for-buck" index (also known as "Fractional KUBE"):

$$A_t = \text{argmax}_i \frac{\hat{\mu}_{i,t-1} + \sqrt{\frac{2 \ln t}{N_i(t-1)}}}{c_i}$$

- **Note**: Our implementation allows dynamic cost updates if costs are stochastic.
- **Reference**: Tran-Thanh et al. (2012). *Knapsack based optimal policies for budget-limited multi-armed bandits.*

4.2 Budgeted Thompson Sampling

Class: `BudgetedThompsonSampling`

Adapts Thompson Sampling for budget-constrained settings where arms have different costs. Standard Thompson Sampling fails in this setting as it ignores costs, often selecting high-reward but expensive arms.

Algorithm:

1. Sample $\theta_i \sim \text{Beta}(\alpha_{i,t}, \beta_{i,t})$ for each arm.
2. Select arm maximizing the ratio of sampled mean to cost:

$$A_t = \text{argmax}_i \frac{\theta_i}{c_i}$$

- **Properties**: Achieves logarithmic regret $O(\ln B)$ where B is the budget. Outperforms standard TS in budgeted settings.
- **Reference**: Lal, A. (2022). *Multi-armed Bandits for Budget-Constrained Data Collection*. (Also: Xia et al., 2015).

---

5. Planned Algorithms

The following algorithms are specified in the roadmap but not yet implemented.

5.1 LUCB (Lower-Upper Confidence Bound)

Target: Best-Arm Identification. Maintains confidence intervals $[L_i, U_i]$ for each arm. Stops when the lower bound of the best arm is higher than the upper bound of any other arm: $\max_{j \neq \hat{i}^*} U_j < L_{\hat{i}^*}$. Samples the arms maximizing the confidence overlap.

5.2 Kasy-Sautmann (Welfare-Constrained Variance Minimization)

Target: Experimental Design / Variance Minimization. Minimize the variance of the treatment effect estimator (allocating samples proportional to standard deviation, Neyman allocation) while ensuring that the experimental units are not subjected to excessively poor treatments (welfare constraint). Solves a constrained optimization problem at each step to determine sampling probabilities.

5.3 Contextual Bandits (LinUCB)

Target: Contextual Regret Minimization. Assumes expected reward is a linear function of a context vector $x_t \in \mathbb{R}^d$: $\mathbb{E}[r_t] = x_t^\top \theta^*$. Mains a ridge regression estimate of θ^* and a confidence ellipsoid. Selects arm maximizing $x_{t,a}^\top \hat{\theta}_t + \alpha\sqrt{x_{t,a}^\top A_t^{-1} x_{t,a}}$.