Reproducing the BaSE Algorithm for Batched Multi-Armed Bandit Problem

Florentine M. Eloundou Nekoul
RAND Corporation
Santa Monica, CA 90401
feloundo@rand.org

1 Introduction

In this paper, we aim to reproduce **Batched Multi-armed Bandits Problem** by Gao, Han, Ren and Zhou, accepted as an oral presentation at NeurIPS 2019. [3]

The multi-armed bandit problem is a multi-round single agent game framing the expected outcomes of the exploration vs exploitation dilemma. In practical settings where outcomes are only observed after a number of arm pulls, or a "batch", the optimal policy choice is made only after observing a small number of batches. In his exploration phase, the agent incurs some loss, which is formalized in the literature by a regret function. Gao et. al (2019) recognize that the regret function in such cases has already been fully characterized, but propose a framework for choosing the regret-minimizing number of arms in the batched version of the paper. They then propose the BaSE (Batched Successive Elimination) algorithm, and explore its bounds under different batching frameworks, or *gris*. The literature for the multi-armed bandit problem is rich, and many of the analytic results are inherited from earlier work, most heavily from Perchet et. al (2016). [2]

The goal for this paper is to reproduce the original empirical results, and perform an ablation of the described BaSE algorithm using and modifying the source code found at: `https://github.com/Mathegineer/batched-bandit`. we do not re-derive the analytical results in this paper, but instead focus on the problem parameters and experimental results. we faithfully reproduce most of the original results and display some extensions. Given that there are very limited features to ablate, we proceed to provide some additional experimental results for discussion.

2 Setting the Game

Recall the game as defined by the original authors. Let there be a stochastic bandit with arms $K \geq 2$ to define the space of arm pulls i such that

$$i \in we = \{1, 2, \ldots, K\}$$

Let the rewards for each pull i be distributed i.i.d. and drawn from a distribution $\nu^{(i)}$ with mean $\mu^{(i)}$. For the entirely of the paper, without loss of generality, the authors assume

$$\nu^{(i)} = \mathcal{N}(\mu^{(i)}, 1)$$

.

Let $\mu^* = max_{i \in [K]} \mu^{(i)}$ denote the maximum reward possible for any given pull, i.e., the reward from pulling from the best arm, and define the gap between rewards from other arms and maximum reward as

$$\Delta_i = \mu^* - \mu^{(i)} \geq 0$$

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

Define the time horizon T and let it split into M batches represented by a grid $\mathcal{T} = \{t_1, \ldots, t_M\}$ where the grid is either **static** —fixed ahead of time —or **adaptive**, where the grid value t_j may be determined after observing rewards up to time t_{j-1} and with some external randomness. In this setting, **batch learning** corresponds to $M = 1$ and **online learning** corresponds to $M = T$.

Denote a sampling policy $\pi = (\pi_t)_{t=1}^T$, s.t. $\pi_t \in [K]$ indicates which arm to pull at time $t \in [T]$, informed by reward observations up to time t_{j-1}. If we define regret as the sum of $\Delta_t \; \forall t \in \pi_t$, the optimization problem is the agent minimizing expected regret, $\mathcal{E}[R_T(\pi)]$ s.t.

$$R_T(\pi) \triangleq \sum_{t=1}^{T} (\mu^* - \mu^{(i)}) = T\mu^* - \sum_{t=1}^{T} \mu^{(\pi_t)}$$

3 The BaSE policy

The paper adapts the Successive Elimination algorithm from previous literature to created the Batched Successive Elimination. Given a **static** grid $\mathcal{T} = \{t_1, \ldots, t_M\}$, the agent explores during the first $M - 1$ batches, and exploits the best arm during the last M batch at time t_M. Meanwhile, after each exploration batch, the agent permanently removes arms that performs worse than the (theretofore) best arm. As such, it is a subclass of the previously defined Explore-then-Commit algorithms.

4 Experimental Results

In this section, we explore the experimental results found with the provided code on the BaSE policy, taking care to graphically display the algorithm's performance under the three different grids: **minimax**, **geometric** and **arithmetic**.

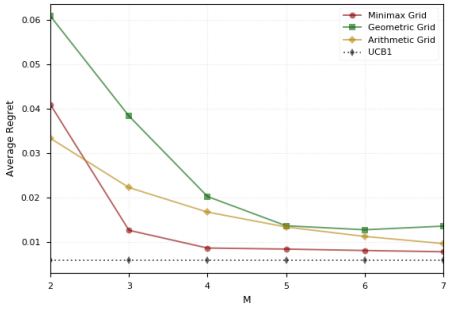
4.1 The Base BaSE Case

For the initial replication, we use the same parameters published by Gao et. al (2019) in their Experimental Results section. Unless otherwise stated, these parameters are the default for our subsequent graphs.

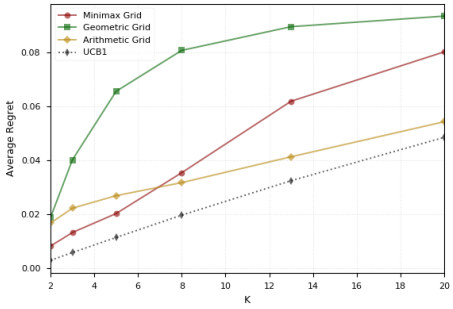
Table 1: Parameters

Symbol	Description	Default Value
T	Time Horizon	5×10^4
K	Bandit Arms	3
M	Batches	3
γ	Tuning Parameter	1
μ^*	Mean Reward for optimal arm	0.6
μ	Mean Reward for other arms	0.5
t_j	Arithmetic Grid value at time j	$j * T/M = j$

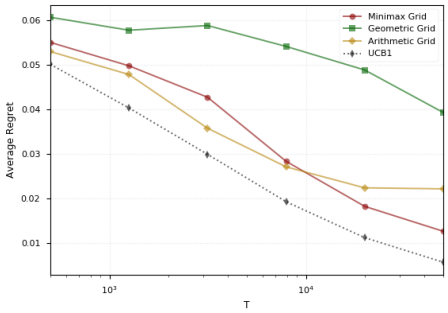
In Figure 1, we see that the results from the paper are replicated exactly, except the ETC Geometric Grid results. In fact, this particular graph negates the assertion in the submitted paper that BaSE always achieves lower regret than ETC. Here we observe that using a geometric grid, ETC sometimes achieves lower regret than BaSE.



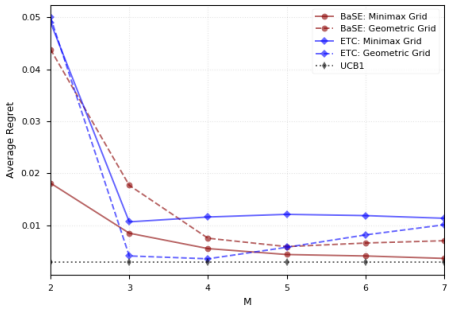
(a) Average regret vs. M



(b) Average regret vs. K



(c) Average regret vs. T



(d) Comparison of BaSE and ETC

Figure 1: Original Results

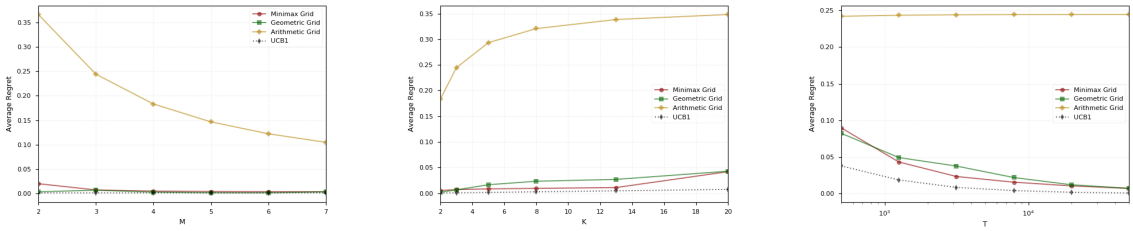
4.2 Large Delta

In defining the minimax and problem-dependent regret functions, Gao et. al (2019) explicitly set the condition that $\Delta_i \leq \sqrt{(K)}$, which is more relaxed than the classical condition that $\Delta_i \in [0, 1]$. In this scenario, we examine outcomes with parameters satisfying the former, but not the latter.

In Figure 2, we can see that all adaptive grids perform very well in this case, as successively elimination occurs faster. The arithmetic grid is much slower and does not converge in this case.

Table 2: $\Delta = 1.1$

Symbol	Description	Value
μ^*	Mean Reward for optimal arm	1.6
μ	Mean Reward for other arms	0.5



(a) Average regret vs. M (b) Average regret vs. K (c) Average regret vs. T

Figure 2: $\Delta = 1.1$

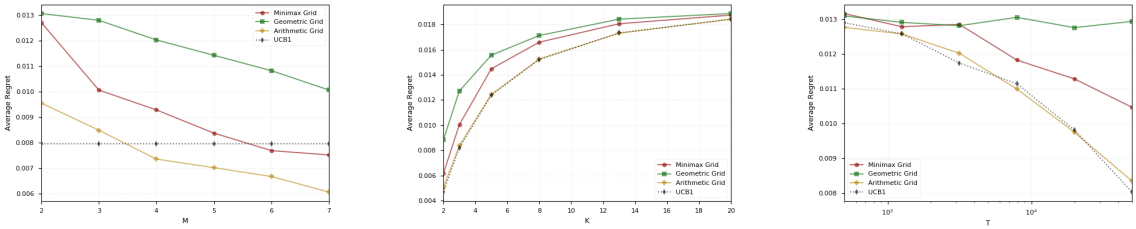
4.3 Small Delta

Exploration requires more batches Δ decreases, so we chose to explore two scenarios in which $\Delta == 0.02$.

In the first, we hold fixed μ^* and increase rewards from non-optimal arms.

Table 3: Small Delta, High Reward

Symbol	Description	Value
μ^*	Mean Reward for optimal arm	0.6
μ	Mean Reward for other arms	0.58



(a) Average regret vs. M (b) Average regret vs. K (c) Average regret vs. T

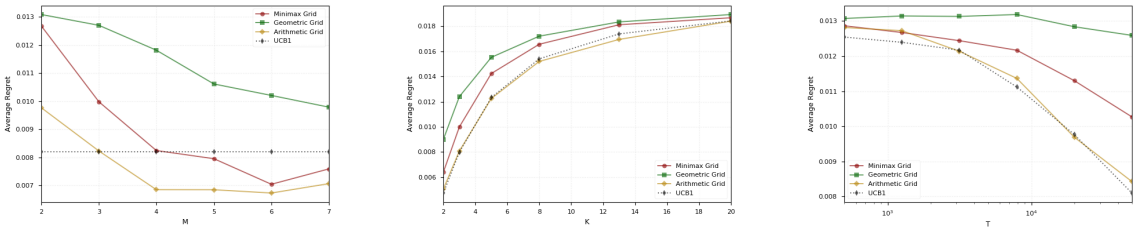
Figure 3: Small Delta, High Reward

In the second case, we decrease both μ and μ^*.

As Gao et. al (2019) demonstrate analytically, translations of the reward distribution (Gaussian or not), does not change the bounds of the regret functions. When Δ is low, regardless of the level of

Table 4: Small Delta, Low Reward

Symbol	Description	Value
μ^*	Mean Reward for optimal arm	0.12
μ	Mean Reward for other arms	0.1



(a) Average regret vs. M (b) Average regret vs. K (c) Average regret vs. T

Figure 4: Small Delta, Low Reward

the reward, the arithmetic grid approximates UCB, and the geometric grid performs the worst, with the minimax grid in the middle.

5 Conclusion

Gao et. al (2019) show that their BaSE algorithm, paired with adaptive grids — especially the minimax grid — is successful and performs well in minimizing regret in batched learning. Its innovation of successive removal of "low"-performing arms works by progressively narrowing the exploration space. In this brief review, we reproduced most of their results, and extended the experimental analysis to include some interesting outcomes of their analytic approach. Esfandiari, Karbasi, Mehrabian and Mirrokni (2019) were able to improve the successive elimination algorithm for batched learning to get optimal results.[1] Time allowing, we would add their algorithm to visualize the differences.

References

[1] Hossein Esfandiari, Amin Karbasi, Abbas Mehrabian, and Vahab Mirrokni. Batched multi-armed bandits problem with Optimal Regret. arXiv preprint arXiv:1910.04959, 2019.

[2] Vianney Perchet, Philippe Rigollet, Sylvain Chassang, and Erik Snowberg. Batched bandit problems. The Annals of Statistics, 44(2):660–681, 2016.

[3] Zijun Gao, Yanjun Han, Zhimei Ren, and Zhengqing Zhou. Batched multi-armed bandits problem. arXiv preprint arXiv:1904.01763, 2019. Accepted in NeurIPS 2019.