Best arm identification in multi-armed bandits with delayed feedback

Aditya Grover[*1], **Todor Markov**[1], **Peter Attia**[1], **Norman Jin**[1], **Nicholas Perkins**[1], **Bryan Cheong**[1], **Michael Chen**[1], **Zi Yang**[2], **Stephen Harris**[3], **William Chueh**[1], **Stefano Ermon**[1]
[1]Stanford University [2]University of Michigan [3]Lawrence Berkeley National Laboratory

Abstract

We propose a generalization of the best arm identification problem in stochastic multi-armed bandits (MAB) to the setting where every pull of an arm is associated with *delayed* feedback. The delay in feedback increases the effective sample complexity of standard algorithms, but can be offset if we have access to *partial* feedback received before a pull is completed. We propose a general framework to model the relationship between partial and delayed feedback, and as a special case we introduce efficient algorithms for settings where the partial feedback are biased or unbiased estimators of the delayed feedback. Additionally, we propose a novel extension of the algorithms to the *parallel* MAB setting where an agent can control a batch of arms. Our experiments in real-world settings, involving policy search and hyperparameter optimization in computational sustainability domains for fast charging of batteries and wildlife corridor construction, demonstrate that exploiting the structure of partial feedback can lead to significant improvements over baselines in both sequential and parallel MAB.

1 INTRODUCTION

Intelligent agents often need to interact with the environment and make rational decisions that optimize for a suitable objective. One such setting that commonly arises is the best arm identification problem in stochastic multi-armed bandits [Bubeck et al., 2009, Audibert and Bubeck, 2010]. In a multi-armed bandit (MAB) problem, an agent is given a set of n finite actions (or arms), each associated with a reward drawn from an arm-specific probability distribution. In a pure exploration setting, the goal is to reliably identify the top-k arms while minimizing the exploration cost. This problem has numerous applications, including optimal experimental design.

We consider a new variant of this problem where the feedback rewards are received after a delay. Delayed feedback is common in the real-world. For instance, hypothesis testing in science and engineering often suffers from delayed feedback since they involve expensive, time-consuming experiments. In one of the motivating applications of this work we want to search over fast-charging policies for electrochemical batteries to maximize lifetime, overcoming the difficulties posed due to lengthy experiments. Even within the field of machine learning, finding the best hyperparameter settings for a given learning algorithm and dataset can be modeled as a best arm identification problem involving a non-trivial delay [Jamieson and Talwalkar, 2016].

However, many scenarios of interest are not complete black-boxes during the intermediate time steps before receiving a delayed feedback reward. Depending on the application, we often have access to side-information in the form of *partial feedback* that can aid decision making. These could be extra measurements such as temperature and remaining capacity while charging batteries in the aforementioned scenario, or learning curves for hyperparameter optimization.

In this work, we propose a general-purpose framework for modeling delayed feedback in MAB, and take a deeper dive into several practically relevant instantiations. In particular, we design and analyze algorithms for best arm identification in the fixed confidence setting where the partial feedback are biased or unbiased estimators of the delayed feedback. Our proposed algorithms adaptively tune the mean and confidence estimates wherever the partial feedback reduces the overall uncertainty. We also extend these algorithms to the parallel MAB setting where we are allowed to pull a batch of arms at every time step [Jun et al., 2016].

[*] primary author. Email: `adityag@cs.stanford.edu`

Proceedings of the 21^{st} International Conference on Artificial Intelligence and Statistics (AISTATS) 2018, Lanzarote, Spain. PMLR: Volume 84. Copyright 2018 by the author(s).

Finally, we empirically validate the proposed algorithms on simulated data and real world datasets drawn from two domains. The first corresponds to experimental design for finding the optimal charging policy for a battery that maximizes overall lifetime [Moura et al., 2017]. In the second domain, we perform hyperparameter optimization for finding the best cut strategy for a standard mixed integer programming solver with performance tested on a benchmark set of problem instances drawn from computational sustainability [Gomes et al., 2008]. Our experiments demonstrate that accounting for partial feedback can reduce the delayed sample complexity on average by 15.6% and 80.8% for sequential MAB over baselines for the two application scenarios respectively. The corresponding average savings over baselines for parallel MAB are 20.7% and 87.6% respectively.

2 BACKGROUND & MODELING FRAMEWORK

The chief workhorse of our analysis will be the law of iterated logarithms (LIL) that analyzes the limiting behavior of random walks (sequence of pulls for a given arm in our case) defined over sub-Gaussian random variables [Darling and Robbins, 1967]. Several finite LIL bounds have been proposed in the literature; we consider the one proposed by Zhao et al. [2016] which has been shown to outperform others empirically while retaining the same asymptotic behavior. Alternate bounds, such as the one by Jamieson et al. [2014], could also be used with no effect on the theoretical analysis of this work.

Lemma 1. *Let $X^{(1)}, X^{(2)}, \ldots$ be i.i.d. sub-Gaussian random variables with scale parameter σ and mean μ. Let τ be any random variable with domain $\mathbb{N}$. For any $c > 1, 2a > c, b > 0$, the following holds with probability at least $1 - 2\zeta(^{2a}/_{c})e^{-2b/c}$:*

$$\left| \frac{1}{\tau} \sum_{l=1}^{\tau} X^{(l)} - \mu \right| \leq \sigma \sqrt{\frac{a \log(\log_c \tau + 1) + b}{\tau}}$$

where ζ denotes the Riemannian zeta function. The constants in Lemma 1 are chosen such that the lemma holds for a target confidence. To simplify the notation, we denote the the error probability by δ' and the right hand side of Lemma 1 by $C(\sigma, \tau, \delta')$ such that the following holds with probability $1 - \delta'$ for any $\tau \in \mathbb{N}$:

$$\left| \frac{1}{\tau} \sum_{l=1}^{\tau} X^{(l)} - \mu \right| \leq C(\sigma, \tau, \delta'). \tag{1}$$

We consider a stochastic multi-armed bandit (MAB) problem characterized by a set of n arms, indexed by $i = 1, \ldots, n$. Each arm is associated with a fixed, unknown probability distribution with means $\{\mu_i\}_{i=1}^n$. We assume that the means are unique. Without loss of generality, assume that the arm indices are sorted as per the means, such that $\mu_1 > \mu_2 > \ldots > \mu_n$.

We are interested in the pure exploration setting, also known as the best arm identification problem, where the goal of an agent is to identify the top-k arms (with the highest means) with a target confidence $1 - \delta$ while minimizing the total time spent on exploration. Exploration in our setting, however, is not the same across the pulls of a given arm. In particular, we assume that each pull of an arm is associated with an unknown (stochastic) delay that contributes to the total exploration time. The presentation in this section assumes a *sequential* MAB setting where the agent can pull/run only one arm at a given time step; the alternate *parallel* MAB setting where an agent can control a "batch" of arms at once is discussed in Section 4 [Perchet et al., 2015, Wu et al., 2015, Jun et al., 2016].

Formally, the stochastic data generating process with delayed feedback can be described as follows. At any given start time t_s:

1. Agent chooses an arm i.
2. Nature samples a delay $D_s \geq 1$ from an (unknown) arm specific delay distribution.
3. Nature samples a sequence of partial feedback, $(Y_{i,t_s+1}, \ldots, Y_{i,t_s+D_s}) \mid D_s$ jointly. The joint distribution of the partial feedback depends on μ_i.

   In general, the delay and partial feedback sequence are unknown to the agent at time t_s.

At time $t_s + \Delta$ where $\Delta \in [1, D_s]$,

4. Nature reveals $Y_{i,t_s+\Delta}$ to the agent.
   If $\Delta = D_s$, the agent goes to step 1. Otherwise, the agent decides whether to continue the current pull (step 4) or start another pull (step 1) in which case any remaining partial feedback for the current pull will not be observed.

The agent and nature continue to play the above game until the agent has selected a set of candidate top-k arms. The delay D_s can contribute significantly to the total time spent on exploration. Under appropriate assumptions however, we can exploit the structure in the partial feedback to significantly reduce the overall exploration cost of delayed feedback. The data generating process described above is very general and one can make many natural assumptions on the distribution of the partial feedback $(Y_{i,t_s+1}, \cdots, Y_{i,t_s+D_s}) \mid D_s$.

For instance, we can model the following scenarios:

- **Full delayed feedback**: The partial feedback at the last delay, Y_{i,t_s+D_s} is sub-Gaussian with mean μ_i and scale parameter σ_i. For the intermediate time steps, $\Delta \in [1, D_s - 1]$, we have $Y_{i,t_s+\Delta} = 0$, and hence, we receive no information about μ_i at these time steps.
- **Incremental partial feedback**: The set of partial feedback $Y_{i,t_s+\Delta}$ for every time step $\Delta \in [1, D_s]$ consists of mutually independent, sub-Gaussian random variables with mean ${}^{\mu_i}/{}_{D_s}$ and scale parameter ${}^{\sigma_i}/{}_{\sqrt{D_s}}$. Hence, the cumulative partial feedback $\sum_{\Delta=1}^{D_s} Y_{i,t_s+\Delta}$ is also sub-Gaussian with mean μ_i and scale parameter σ_i.
- **Unbiased noisy partial feedback:** The partial feedback at the last delay, Y_{i,t_s+D_s} is sub-Gaussian with mean μ_i and scale parameter σ_i. For the intermediate time steps, $\Delta \in [1, D_s - 1]$, the set of partial feedback $Y_{i,t_s+\Delta} \mid Y_{i,t_s+D_s} - Y_{i,t_s+D_s}$ consists of mutually independent, sub-Gaussian random variables with zero mean and scale parameter $\sigma_i^{(p)}$.
- **Biased noisy partial feedback:** The partial feedback at the last delay, Y_{i,t_s+D_s} is sub-Gaussian with mean μ_i and scale parameter σ_i. For the intermediated time steps, $\Delta \in [1, D_s - 1]$, the set of partial feedback $Y_{i,t_s+\Delta} \mid Y_{i,t_s+D_s} - Y_{i,t_s+D_s}$ consists of mutually independent, sub-Gaussian random variables with mean b_i and scale parameter $\sigma_i^{(p)}$. Here, b_i is a fixed, but unknown bias associated with the partial feedback for the arm.

Note that the standard MAB setting where we observe the feedback at the immediate next time step is a special case of the full delayed feedback with a constant delay $D_s = 1$ for every pull. In fact, the algorithms for best arm identification in the *full delayed* and *incremental partial feedback* settings can be derived naturally from the standard MAB algorithms with no delays. Specifically, the agent can simply chose to ignore the time instants at which delayed feedback is unavailable for the full delayed feedback setting. The sample complexity of any such algorithm is hence the number of arm pulls required in the standard MAB setting weighted by the delay of every pull. These settings are still interesting for parallel MAB where information can be shared across arms; we discuss this case in Section 4.

Algorithm 1 RacingSubroutines

```
function UpdateArmSets(arm sets $A$, $R$, $S$, top $k$, confidence bounds $\{LCB_i, UCB_i\}_{i \in S}$)
    Initialize $k_t \leftarrow k - |A|$.
    Update $A \leftarrow A \cup \{i \in S \mid LCB_i > \max_{j \in S}^{(k_t+1)} UCB_j\}$.
    Update $R \leftarrow R \cup \{i \in S \mid UCB_i < \max_{j \in S}^{(k_t)} LCB_j\}$.
    Update $S \leftarrow S \backslash \{R \cup A\}$.
    return $A$, $R$, $S$.
end function

function GetBatchArms(surviving arms $S$, counts $\{N_i, a_i\}_{i \in S}$, effective batch size $e$, limit $r$)
    Initialize new arm pulls $\mathbf{m} \leftarrow \mathbf{0} \in \mathbf{R}^n$.
    for slot $s = \{1, \cdots, \min(e, |S|r)\}$ do
        Least pulled arm $j \leftarrow \arg\ \min_{i \in S: a_i \leq r} N_i$
        Update $a_j \leftarrow a_j + 1$.
        Update $m_j \leftarrow m_j + 1$.
        Update $N_j \leftarrow N_j + 1$.
    end for
    return $\mathbf{m}, \{N_i\}_{i \in S}, \{a_i\}_{i \in S}$
end function
```

The *partial feedback* settings, however, present an interesting scenario where the agent can extract information from noisy feedback. For such settings, we propose modified algorithms based on racing-style procedures typically used for the standard MAB setting [Maron and Moore, 1994]. Typically, racing algorithms maintain three disjoint arm sets: accepted arms A, rejected arms R, and surviving arms S. Initially, all arms are assigned to the surviving set S. Racing procedures uniformly sample arms while removing them from the surviving set based on confidence bounds. For convenience, define the lower confidence bounds (LCB) and upper confidence bounds (UCB) for every arm i as:

$$LCB_i := \widehat{\mu}_i - C_i \tag{2}$$

$$UCB_i := \widehat{\mu}_i + C_i \tag{3}$$

where $\widehat{\mu}_i$ is the empirical mean of the feedback for arm i and the confidence bound C_i will depend on the particular racing algorithm under consideration. Let $k_t := k - |A|$ be the effective number of top arms remaining to be identified at a time step t. Each time we receive a feedback reward (full or partial), the racing procedures update these sets based on the rule that any arm in S whose LCB is greater than the UCB of $|S| - k_t$ arms is accepted. Similarly, any arm in S whose UCB is less than the LCB of k_t arms is rejected. The racing procedure is repeated until S is empty. The pseudocode for the subroutine that updates the arm sets is given in Algorithm 1.

3 SEQUENTIAL MAB

In sequential MAB, we assume that the agent can receive (partial) feedback from only a single arm pull at any given time step, *e.g.*, we can only perform one experiment at a time. We skip a separate discussion on the trivial full feedback (and the related incremental feedback) setting and discuss it only in the context of the *noisy feedback settings*. For convenience, we denote the partial feedback at the last delay as $X_{i,t_s} = Y_{i,t_s+D_s}$. Here, X_{i,t_s} is a sub-Gaussian random variable with mean μ_i and scale parameter σ_i. The proofs of all results in this section are given in the Appendix.

3.1 Unbiased noisy partial feedback

In this setting, an agent has access to unbiased partial feedback at the intermediate time steps before receiving the full delayed feedback. In the following result, we derive a variation of the finite LIL bound for the unbiased partial feedback setting.

Proposition 1. *Let* $\{Y_{i,t_1+1}, Y_{i,t_1+2}, \ldots, Y_{i,t_1+D_1}, Y_{i,t_2+1}, \ldots, Y_{i,t_2+D_2}, \ldots\}$ *denote the partial feedback sequences for the pulls of an arm i started at time steps $t_1, t_2, \ldots$ and delays $D_1, D_2, \ldots$. Then, under the distributional assumptions on the unbiased partial feedback (see Section 2) for any $F \in \mathbb{N}$, $P \in [1, D_F]$, $\delta_f > 0, \delta_p > 0$, we have with probability* $1 - \delta_f - \delta_p$*:*

$$\left| \frac{1}{F} \left[\sum_{f=1}^{F-1} X_{i,t_f} + \frac{1}{P} \sum_{l=1}^{P} Y_{i,t_F+l} \right] - \mu_i \right| \\ \leq C\left(\sigma_i, F, {}^{\delta_f}\!/\!{}_n\right) + \frac{1}{F} C\left(\sigma_i^{(p)}, P, {}^{\delta_p}\!/\!{}_n\right) \forall i \in [1, n] \quad (4)$$

where $X_{i,t_f} = Y_{i,t_f+D_f}$ by definition. At any intermediate time step between the the start and end of the F-th arm pull, Proposition 1 adaptively "splits" the confidence bounds pertaining to the full delayed feedback for F steps (first term in the RHS) and the partial delayed feedback for the F-th arm pull (second term in the RHS). Contrast this with the full delayed feedback setting where the following confidence bound holds with probability $1 - \delta$:

$$\left| \frac{1}{F-1} \sum_{f=1}^{F-1} X_{i,t_f} - \mu_i \right| \leq C\left(\sigma_i, F-1, {}^{\delta}\!/\!{}_n\right) \forall i \in [1, n] \quad (5)$$

To obtain the same target confidence in the two cases above, we constrain $\delta = \delta_f + \delta_p$. Solving for the optimal δ_f^*, δ_p^* that minimize the RHS of Eq. (4) under the constraint due to δ corresponds to a convex optimization problem that can be solved in closed form. Comparing the mean estimators in Eq. (4) and Eq. (5), we note that the agent can only use the full delayed feedback up till the $(F-1)$-th arm pull while waiting for the outcome of the F-th arm pull in the latter case while the former dynamically incorporates the partial feedback observed for the F-th arm pull.

Algorithm 2 RacingUnbiasedPF (arm parameters $\{i, \sigma_i, \sigma_i^{(p)}\}_{i=1}^n$, top k, confidence δ)

```
 1: Initialize global time step $t = 0$, surviving $S = \{i\}_{i=1}^n$, accepted $A = \{\}$, rejected $R = \{\}$.
 2: Initialize per-arm full delayed feedback counter $F_i = 0$, empirical means $\widehat{\mu}_i = 0$, confidence bounds $LCB_i = -\infty$, $UCB_i = \infty$ for all $i \in S$.
 3: while $S$ is not empty do
 4:     while True do
 5:         Increment $t \leftarrow t + 1$.
 6:         Collect partial feedback $Y_{a,t}$.
 7:         Update $\widehat{\mu}^{(p)} \leftarrow \frac{(P\widehat{\mu}^{(p)} + Y_{a,t})}{(P+1)}$.
 8:         Increment $P \leftarrow P + 1$.
 9:         Set $C^{(partial)} \leftarrow C(\sigma_a, F_a + 1, {}^{\delta_f^*}\!/\!{}_n) + \frac{C(\sigma_a^{(p)}, p, \delta_p^*/n)}{F_a+1}$.
10:         Choose FOrP $\leftarrow \arg\min\left(C(\sigma_a, F_a, {}^{\delta}\!/\!{}_n), C^{(partial)}\right)$.
11:         Update $C_a \leftarrow C(\sigma_a, F_a, {}^{\delta}\!/\!{}_n)$ if FOrP $= F$ else $C^{(partial)}$.
12:         Update $\widehat{\mu}_a \leftarrow \widehat{\mu}^{(f)}$ if FOrP $= F$ else $\frac{F_a\widehat{\mu}^{(f)} + \widehat{\mu}^{(p)}}{F_a+1}$.
13:         Update $LCB_a, UCB_a$.
14:         $A, R, S \leftarrow$ UpdateArmSets$(A, R, S, k, \{LCB_i, UCB_i)\}_{i \in S})$.
15:         if $P = D_{a,t_a}$ or $a \notin S$ then
16:             Break ▷ Pull on termination/elimination
17:         end if
18:     end while
19:     Pull arm $a$ where $a \leftarrow \arg\min_{a \in S} F_a$.
20:     Initialize start $t_a \leftarrow t$, partial feedback counter $P = 0$, partial mean $\widehat{\mu}^{(p)} = 0$, full mean $\widehat{\mu}^{(f)} \leftarrow \widehat{\mu}_i$.
21: end while
22: return $A$
```

Based on the above analysis, we propose a racing algorithm for the unbiased partial feedback setting with the psuedocode given in Algorithm 2. At any intermediate time step, the agent chooses a mean estimator and a confidence bound for the current arm (Lines 10-13). The choice corresponds to the tighter confidence bound obtained either by optimizing Eq. (4) over δ_p, δ_f or the one obtained by Eq. (5) where only the full delayed feedback are considered. Thereafter, the agent invokes the racing subroutine that checks whether a surviving arm can be rejected or accepted (Line 14). If the pull has finished running or the current arm is itself eliminated (Line 15), the agent pulls a new arm in the next time step which has the least number of full delayed feedback (Line 19).

We can make some observations about Algorithm 2. First, we see that an agent adopting the proposed algorithm can never do worse than the alternate racing strategy that considers estimates only based on the full delayed feedback. This is because even at the intermediate time steps, the agent considers the mean estimator corresponding to the smaller of the two confidence bounds, which can only reduce the delayed sample complexity of the algorithm. Whenever an arm

Algorithm 3 BatchRacingFullDF(arm parameters $\{i, \sigma_i\}_{i=1}^n$, top k, confidence δ, batch b, limit r)

```
Initialize global time step $t = 0$, pull status counts running $= 0$, surviving arms $S = \{i\}_{i=1}^n$, accepted arms $A = \{\}$, rejected arms $R = \{\}$.
Initialize per-arm global pull counts $N_i = 0$, running pull counts $a_i = 0$, full delayed feedback $F_i = 0$, empirical means $\hat{\mu}_i = 0$, confidence bounds $LCB_i = -\infty$, $UCB_i = \infty$ for all $i \in S$.
while $S$ is not empty do
    if running $> 0$ then
        Increment $t \leftarrow t + 1$.
        Collect batch full delayed feedback $Y$.
        for all $Y_{h,t} \in Y$ do
            Update $\widehat{\mu}_h \leftarrow {(F_h \widehat{\mu}^{(f)} + Y_{h,t})}/{(F_h + 1)}$.
            Increment $F_h \leftarrow F_h + 1$.
            Update $LCB_h, UCB_h$.
            Decrement $a_h \leftarrow a_h - 1$.
        end for
        if $Y$ is not empty then
            $A, R, S \leftarrow \text{UpdateArmSets}(A, R, S, k, \{LCB_i, UCB_i\}_{i \in S})$.
            Decrement running $\leftarrow$ running $- |Y|$.
        end if
    end if
    Update arms $\mathbf{m}$, counts $\{N_i, a_i\}_{i \in S} \leftarrow$ GetBatchArms$(S, \{N_i, a_i\}_{i \in S}, b -$ running$, r)$.
    Pull every arm $j \in \mathbf{m}$ $m_j$ times.
    Update running $\leftarrow$ running $+ \sum_{j \in \mathbf{m}} m_j$.
end while
return $A$
```

pull has finished, the agent also updates the mean and confidence interval by an arithmetic averaging over *only* the full delayed feedback. Using partial feedback is impractical at such time steps since the partial feedback only introduce noise and do not provide any additional information about the true mean.

If the maximum possible delay associated with any arm pull is given by $D_{\max}$, then we can trivially extend bounds for the sample complexity of racing style procedures [Jamieson and Nowak, 2014] to derive similar bounds on the *delayed sample complexity* with an extra multiplicative factor of $D_{\max}$.[1] This is similar to what one would expect from the full delayed feedback setting and is not surprising for Algorithm 2 since in the absence of any additional assumptions, the partial feedback could be completely uninformative and the algorithm will choose to ignore them. We believe domain-specific assumptions about the delay distribution and the noise associated with the partial feedback as a function of time could lead to a tighter analysis and is an interesting direction of future work. The correctness of Algorithm 2 can be summarized below.

Theorem 1. *Assuming the delay associated with any arm pull is bounded, then Algorithm 2 outputs the top-k arms with probability at least* $1 - \delta$.

To get further intuition about the working of Algorithm 2, consider the situation where all arms have been pulled once except one. When the last remaining arm is pulled for the first time, the full delayed feedback setting will necessarily have to wait for the pull to finish running before eliminating the arms whereas Algorithm 2 can potentially start eliminating arms right after the first partial delayed feedback is received.

3.2 Biased noisy partial feedback

The partial feedback at the intermediate time steps before a full delayed feedback can also correspond to biased estimates of the full delayed feedback. Although the bias for the arms is unknown, it can be estimated empirically based on differences in the full delayed feedback and the partial feedback at the corresponding intermediate time steps. Formally, we assume the bias for a particular arm is an unknown constant b_i and derive the following LIL bounds.

Proposition 2. *Let* $\{Y_{i,t_1+1}, Y_{i,t_1+2}, \ldots, Y_{i,t_1+D_1}, Y_{i,t_2+1}, \ldots, Y_{i,t_2+D_2} \ldots\}$ *denote the partial feedback sequences for the pulls of an arm* i *started at time steps* $t_1, t_2, \ldots$ *and delays* $D_1, D_2, \ldots$ *with bias* b_i. *Then, under the distributional assumptions on the partial feedback (see Section 2) for any* $F \in \mathbb{N} \backslash \{1\}$, $P \in [1, D_F]$, $\delta_f > 0, \delta_p > 0, \delta_b > 0$, *we have with probability* $1 - \delta_f - \delta_p - \delta_b$*:*

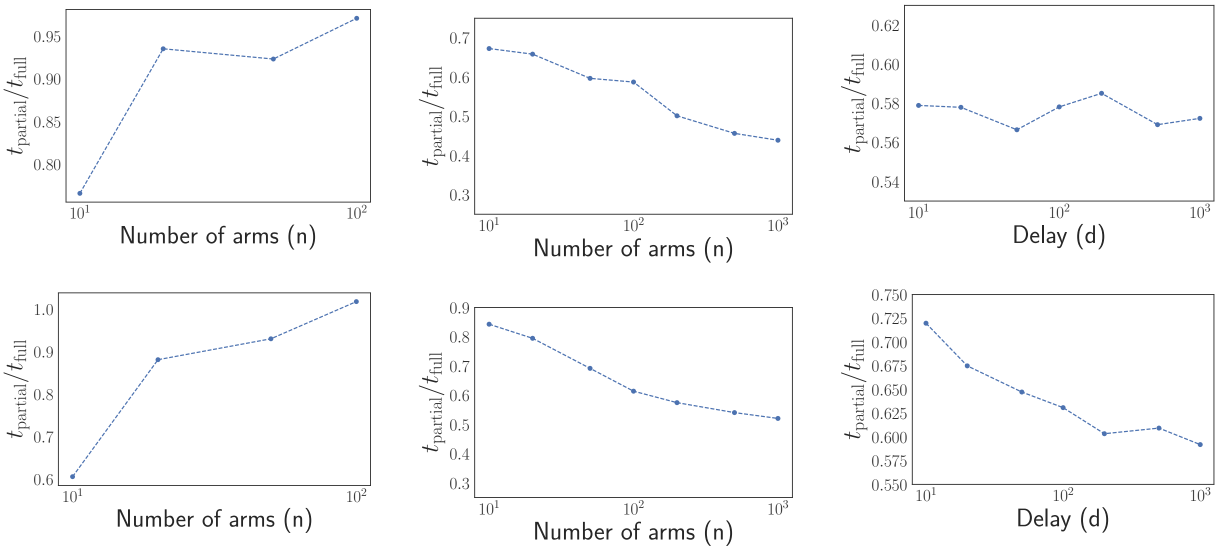
$$\left| \frac{1}{F} \left[\sum_{f=1}^{F-1} X_{i,t_f} + \frac{1}{P} \sum_{p=1}^{P} (Y_{i,t_F+p} - Z_{i,F}) \right] - \mu_i \right|$$

$$\leq C\left(\sigma_i, F, {\delta_f}/{n}\right) + \frac{1}{F}\left[C\left(\sigma_i^{(p)}, P, {\delta_p}/{n}\right) + C\left(\sigma_i^{(p)}, F-1, {\delta_b}/{n}\right)\right] \tag{6}$$

$$\forall i \in [1, n] \textit{where } Z_{i,F} = \frac{1}{F-1} \sum_{f=1}^{F-1} \left(\frac{\sum_{p=1}^{D_f - 1} Y_{i,t_f+p}}{D_f - 1} - X_{i,D_f-1} \right).$$

Comparing Eq. (6) with Eq. (5) by constraining $\delta = \delta_f + \delta_p + \delta_b$, we see that the mean estimator takes into account the partial feedback as before but also has a bias correction term. The bias correction term is an empirical average of the biases observed from the past full delayed feedback. This correction has the effect of introducing additional uncertainty (third term in the RHS) and we need at least one full feedback to estimate the bias before we can use the above bound. The corresponding racing algorithm runs similar to Algorithm 2 with the key difference being that the

[1] The delayed sample complexity for an algorithm refers to the total number of time steps (including delays) before termination.



(a) Number of arms - Bounded means (b) Number of arms - Free means (c) Delay

Figure 1: Synthetic experiments evaluating performance. **Top:** sequential. **Bottom:** parallel. Lower is better.

mean estimator corresponds to the minimum of the confidence bounds in Eq. (5) and Eq. (6), where the RHS of Eq. (6) is specified for the optimal $\delta_f^*, \delta_p^*, \delta_b^*$ minimizing the expression under the constraint due to δ. We defer the pseudocode for this setting to the Appendix (see Algorithm 4).

4 PARALLEL MAB

In parallel MAB, an agent has the additional ability to "accumulate" bulk information by controlling a batch of arm pulls. We extend the (b, r) setting proposed in Jun et al. [2016] where the agent is allowed to run at most b arm pulls in parallel at any given time step with an upper limit r on the number of pulls of each arm.

Even the full delayed feedback setting becomes interesting, as the agent can exploit information from arm pulls which have finished running in parallel to accept/reject delayed arm pulls that are still running thereby avoiding the pitfalls of long delays. The pseudocode for the proposed batch racing algorithm with full delayed feedback is given in Algorithm 3. At every time step, an agent pulls a batch of arms with the least pull count N_i that obeys the (b, r) constraints (Lines 18-19). Whenever we obtain at least one full delayed feedback, we can update our arm sets as per the racing criteria (Lines 13-15).

The algorithms for the noisy partial feedback settings discussed in Section 3 can be extended for parallel MAB in a similar manner and are skipped here to keep the presentation clean. The theoretical analysis of the batch MAB setting in Jun et al. [2016] builds on the analysis of standard MAB in ways independent of the choice of LIL bounds and hence, a merged analysis for delayed batch MAB using the LIL bounds for delayed feedback (as in Propositions 1 and 2) suggests a reduction factor of b in the corresponding upper bounds.

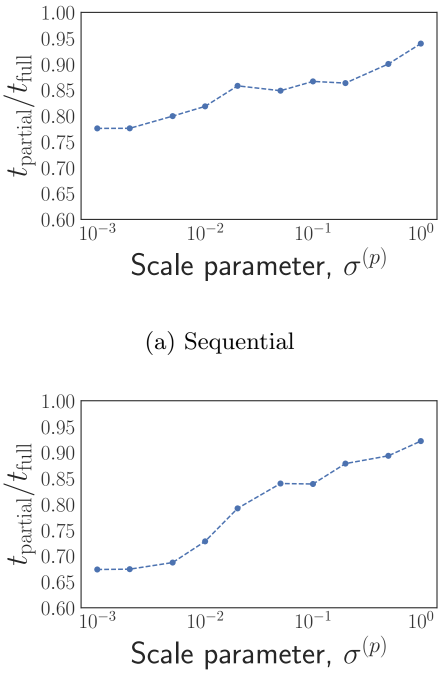
5 EXPERIMENTS

We empirically validated the proposed algorithms on a simulated setting and two real world datasets. All experiments use an error probability of $\delta = 0.05$ and we observed that in each case, the algorithm obtains the desired confidence level empirically. For the parallel MAB setting, we set $b = r = 10$.

5.1 Simulated data

We performed an ablation study of the proposed algorithms for sequential and parallel MAB under different settings of delayed feedback. All experiments were repeated for 100 random runs such that the standard errors are vanishingly small and the number of top arms to be identified, k is set to $0.2n$. We quantify improvement as the ratio $(={}^{t_{\text{partial}}}/_{t_{\text{full}}})$ of the time taken by Algorithm 2 or its parallel MAB extension (*i.e.*, t_{partial}) and the time taken by a full delayed feedback racing procedure (*i.e.*, t_{full}). We evaluate performance as a function of the following problem parameters.

Number of arms. To analyze the difference in performance as a function of the number of arms (n), we further consider two distribution of means.



(a) Sequential

(b) Parallel

Figure 2: Experiments on battery charging.

In the *bounded means* case, we set the means of the arms as $\mu_i = c - (i/n)^{\tilde{c}}$ for any choice of constants c and $\tilde{c} > 0$. Hence, the range of the means does not vary with n. In Figure 1a, we observe that accounting for unbiased partial feedback can give gains of up to 25% and 40% for the sequential and parallel MAB when the number of arms is low. The gains are reduced when the number of arms is large, which suggests that partial feedback is less advantageous in scenarios where a large number of full pulls are required for disambiguating very closely spaced means.

In the *free means* case, we set the means of the arms as $\mu_i = c - \tilde{c}i$ for any choice of constants c and $\tilde{c} > 0$. Here, the range of the means increases with n. From the results in Figure 1b, we observe that the gains due to partial feedback improve as the number of arms increases. This suggests that when the relative separation in means between the arms is fixed, Algorithm 2 and its parallel MAB extension quickly eliminate arms with extreme means (very high or very low) unlike the racing algorithms that wait for full delayed feedback.

Delay. Here, we fix $n = 100$ and vary the delay of the arms. For all settings of the delay in Figure 1c, Algorithm 2 and its parallel MAB extension require a significantly lower fraction of the time with the lowest ratios observed to be 0.59 and 0.57 for sequential and parallel MAB respectively. While we did not see much variation in improvements for sequential MAB, the improvements are better for longer delays in the case of parallel MAB.

5.2 Policy search for fast battery charging

For any given battery chemistry, the charging (and discharging) policy has a significant impact on the lifetime of the cells. However, a single run of a particular policy however takes months to complete since every cell needs to be repeatedly charged and discharged until the end of its lifetime. Hence, delayed feedback can significantly slow down the search procedure. The true, unknown reward for any arm (charging policy) is stochastic and corresponds to the lifetime of the battery [Harris et al., 2017, Baumhöfer et al., 2014, Schuster et al., 2015].[2]

We model the search for the best charging policy for the Li-ion battery chemistry as a best arm identification problem in a stochastic MAB with $n = 40$ arms, $k = 1$. The true mean cycle life, cell-to-cell variances, and delays are obtained from a battery charging simulator [Moura et al., 2017, Perez et al., 2016]. While a battery cell undergoes charging and discharging, we can additionally monitor key indicators such as voltage, temperature, and internal resistance. Predictive models of lifetime based on these factors is an active area of research, and can serve the purpose of partial feedback estimator [Burns et al., 2013, Dubarry et al., 2017]. We assume the existence of such an estimator and test the robustness of our algorithm by evaluating the relative improvements obtained from Algorithm 2 on varying the noise $\sigma_i^{(p)}$ associated with the partial feedback. The results are shown in Figure 2. When the estimator is "trustworthy" (low $\sigma_i^{(p)}$), we can achieve improvements of up to 35% in the number of experiments required. As expected, the gains diminish for poorer models of partial feedback in which case the algorithm can choose to ignore the noisy feedback.

5.3 Hyperparameter optimization for mixed integer programming

The CPLEX solver[3] for mixed integer programming has a host of hyperparameters, including options to switch on or off different *cut* strategies employed by the solver during the search process. We model the task of finding the best cut strategy as a stochastic MAB problem with $n = 32$ arms (*i.e.*, cut strategies),

[2] Formally, the lifetime of the cell is defined to be the number of cycles until a battery reaches 80% of its original capacity at which point a battery is considered dead.

[3] `https://www.ibm.com/software/commerce/optimization/cplex-optimizer/index.html`

$k = 1$. The performance is measured on CORLAT, a benchmark set of $2,000$ (maximization) mixed integer linear programming instances derived from real world data used for the construction of a wildlife corridor for grizzly bears in the Northern Rockies region [Gomes et al., 2008, Hutter et al., 2010]. The true mean for each arm is the average of lower bounds attained by the cut strategy on the feasible instances in the dataset under specified time and resource constraints per instance (10 seconds on 1 core). Every pull of an arm corresponds to running a cut strategy on a sampled problem instance.

Instead of waiting for the solver to completely solve (or time out) a sampled problem instance, we can save computation by using partial feedback about the search process. In particular, the solver outputs the best integral lower bound (LB) and real valued upper bound (UB) found after executing each cut during search. The final output of the solver is the best lower bound. To obtain an unbiased partial feedback estimator, we use a training subset of 500 instances to learn a linear model that predicts the final lower bound for a given input instance based on the intermediate lower and upper bounds. The best arm identification algorithms are tested on the remaining instances in the dataset. Conditioned on a problem instance, the uncertainty associated with the partial feedback, $\sigma_i^{(p)}$ is given by $(UB - LB)/2$ and shrinks with an increase in the time steps elapsed. Note that the delays are not fixed and depend on both the cut strategy and the problem instance under consideration. We directly report the final results: the percentage reduction in time taken by the unbiased partial feedback scenarios over full delayed feedback is 80.8% and 87.6% for sequential and parallel MAB respectively stressing the importance of partial feedback for this particular application scenario.

6 RELATED WORK

Early work in pure exploration is attributed to Bechhofer [1958] and Paulson [1964] who studied this problem in the context of optimal experimental design. Modern day literature can be categorized into either the *fixed budget* or the *fixed confidence* settings. Algorithms for the fixed budget setting strive to maximize the probability of identifying the top-k arms [Audibert and Bubeck, 2010, Bubeck et al., 2013, Kaufmann et al., 2015]. In the fixed confidence setting, which is the one we consider in this paper, the goal is to minimize the number of pulls to attain a target confidence [Maron and Moore, 1994, Bubeck et al., 2009]. See Gabillon et al. [2012] for a unified treatment of the two settings.

Algorithms for the fixed confidence setting can be broadly classified into racing style procedures which sample arms uniformly and eliminate sub-optimal arms [Maron and Moore, 1994, Even-Dar et al., 2002] and the UCB/LUCB style procedures which adaptively sample arms without explicit elimination. We direct the reader to the excellent survey by Jamieson and Nowak [2014] that summarizes the major advancements in the analysis of the sample complexity of these algorithms. Algorithmic generalizations of the best arm identification include top-k identification [Heidrich-Meisner and Igel, 2009] and the parallel MAB settings for batch arm pulls [Perchet et al., 2015, Jun et al., 2016, Wu et al., 2015] among others.

While the delayed feedback framework we propose is novel to the pure exploration problem, online learning with delays has been studied previously in the regret minimization setting [Weinberger and Ordentlich, 2002, Joulani et al., 2013, Desautels et al., 2014]. In particular, algorithms designed particularly for hyperparameter optimization have enjoyed great success. Krueger et al. [2015] proposes a modified cross-validation procedure performed on increasing subsets of data coupled with a sequential testing strategy to eliminate the poor parameter configurations early on. Jamieson and Talwalkar [2016] and Li et al. [2017] recently proposed algorithms for hyperparameter optimization based on non-stochastic MAB. Here, the arms correspond to hyperparameter configurations, and a pull is equivalent to observing a fixed sequence of losses.

For many real-world problems, we have access to a shared structure across arms that makes the overall problem amenable to Bayesian optimization techniques [Snoek et al., 2012, Eggensperger et al., 2013, Snoek et al., 2015, Feurer et al., 2015, McIntire et al., 2016b,a]. Combining the LIL bounds we proposed for noisy partial feedback with Bayesian multi-armed bandits [Srinivas et al., 2010, Krause and Ong, 2011, Hoffman et al., 2014] is a promising extension we are pursuing for our on-going real world application relating to efficient search of fast charging policies for Li-ion battery cells [Ermon et al., 2012].

7 CONCLUSIONS

We introduced a new general framework for pure exploration in stochastic multi-armed bandit problems with partial and delayed feedback. We provided efficient algorithms for solving specific instantiations of our framework that can naturally model real world scenarios, especially in the context of optimal experimental design. We leave as future work the problem of identifying information-theoretic lower bounds on the sample complexity of the new pure exploration problems we formulated. Extension of our framework to the fixed budget setting is another interesting direction for future work.

ACKNOWLEDGEMENTS

We are thankful to Neal Jean and Daniel Levy for helpful comments on early drafts. This research has been supported by a Microsoft Research PhD fellowship in machine learning for the first author, NSF grants #1651565, #1522054, #1733686, Toyota Research Institute, Future of Life Institute, Precourt Institute for Energy, and Intel.

References

Jean-Yves Audibert and Sébastien Bubeck. Best arm identification in multi-armed bandits. In *Conference on Learning Theory*, 2010.

Thorsten Baumhöfer, Manuel Brühl, Susanne Rothgang, and Dirk Uwe Sauer. Production caused variation in capacity aging trend and correlation to initial cell performance. *Journal of Power Sources*, 247: 332–338, 2014.

Robert E Bechhofer. A sequential multiple-decision procedure for selecting the best one of several normal populations with a common unknown variance, and its use with various experimental designs. *Biometrics*, 14(3):408–429, 1958.

Sébastien Bubeck, Rémi Munos, and Gilles Stoltz. Pure exploration in multi-armed bandits problems. In *International conference on Algorithmic Learning Theory*, 2009.

Séebastian Bubeck, Tengyao Wang, and Nitin Viswanathan. Multiple identifications in multi-armed bandits. In *International Conference on Machine Learning*, 2013.

JC Burns, Adil Kassam, NN Sinha, LE Downie, Lucie Solnickova, BM Way, and JR Dahn. Predicting and extending the lifetime of li-ion batteries. *Journal of The Electrochemical Society*, 160(9):A1451–A1456, 2013.

DA Darling and Herbert Robbins. Iterated logarithm inequalities. *Proceedings of the National Academy of Sciences*, 57(5):1188–1192, 1967.

Thomas Desautels, Andreas Krause, and Joel W Burdick. Parallelizing exploration-exploitation tradeoffs in gaussian process bandit optimization. *Journal of Machine Learning Research*, 15(1):3873–3923, 2014.

Matthieu Dubarry, M Berecibar, A Devie, D Anseán, N Omar, and I Villarreal. State of health battery estimator enabling degradation diagnosis: Model and algorithm description. *Journal of Power Sources*, 360: 59–69, 2017.

Katharina Eggensperger, Matthias Feurer, Frank Hutter, James Bergstra, Jasper Snoek, Holger Hoos, and Kevin Leyton-Brown. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In *Advances in Neural Information Processing Systems workshop on Bayesian Optimization in Theory and Practice*, 2013.

Stefano Ermon, Yexiang Xue, Carla Gomes, and Bart Selman. Learning policies for battery usage optimization in electric vehicles. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2012.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Pac bounds for multi-armed bandit and markov decision processes. In *Conference on Learning Theory*, 2002.

Matthias Feurer, Jost Tobias Springenberg, and Frank Hutter. Initializing bayesian hyperparameter optimization via meta-learning. In *AAAI Conference on Artificial Intelligence*, 2015.

Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems*, 2012.

Carla Gomes, Willem-Jan Van Hoeve, and Ashish Sabharwal. Connections in networks: A hybrid approach. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 303–307, 2008.

Stephen J Harris, David J Harris, and Chen Li. Failure statistics for commercial lithium ion batteries: A study of 24 pouch cells. *Journal of Power Sources*, 342:589–597, 2017.

Verena Heidrich-Meisner and Christian Igel. Hoeffding and Bernstein races for selecting policies in evolutionary direct policy search. In *International Conference on Machine Learning*, 2009.

Matthew W Hoffman, Bobak Shahriari, and Nando de Freitas. Exploiting correlation and budget constraints in bayesian multi-armed bandit optimization. In *International Conference on Artificial Intelligence and Statistics*, 2014.

Frank Hutter, Holger Hoos, and Kevin Leyton-Brown. Automated configuration of mixed integer programming solvers. *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 186–202, 2010.

Kevin Jamieson and Robert Nowak. Best-arm identification algorithms for multi-armed bandits in the fixed confidence setting. In *Conference on Information Sciences and Systems*, pages 1–6. IEEE, 2014.

Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *International Conference on Artificial Intelligence and Statistics*, 2016.

Kevin G Jamieson, Matthew Malloy, Robert D Nowak, and Sébastien Bubeck. lil'UCB: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory*, 2014.

Pooria Joulani, Andras Gyorgy, and Csaba Szepesvári. Online learning under delayed feedback. In *International Conference on Machine Learning*, 2013.

Kwang-Sung Jun, Kevin Jamieson, Robert Nowak, and Xiaojin Zhu. Top arm identification in multi-armed bandits with batch arm pulls. In *Artificial Intelligence and Statistics*, 2016.

Emilie Kaufmann, Olivier Cappé, and Aurélien Garivier. On the complexity of best arm identification in multi-armed bandit models. *Journal of Machine Learning Research*, 2015.

Andreas Krause and Cheng S Ong. Contextual gaussian process bandit optimization. In *Advances in Neural Information Processing Systems*, 2011.

Tammo Krueger, Danny Panknin, and Mikio L Braun. Fast cross-validation via sequential testing. *Journal of Machine Learning Research*, 2015.

Lisha Li, Kevin Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. In *International Conference on Learning Representations*, 2017.

Oded Maron and Andrew W Moore. Hoeffding races: Accelerating model selection search for classification and function approximation. *Advances in Neural Information Processing Systems*, 1994.

Mitchell McIntire, Tyler Cope, Daniel Ratner, and Stefano Ermon. Bayesian optimization of FEL performance at LCLS. In *International Particle Accelerator Conference*, 2016a.

Mitchell McIntire, Daniel Ratner, and Stefano Ermon. Sparse Gaussian processes for Bayesian optimization. In *Conference on Uncertainty in Artificial Intelligence*, 2016b.

Scott J Moura, Federico Bribiesca Argomedo, Reinhardt Klein, Anahita Mirtabatabaei, and Miroslav Krstic. Battery state estimation for a single particle model with electrolyte dynamics. *IEEE Transactions on Control Systems Technology*, 25(2):453–468, 2017.

Edward Paulson. A sequential procedure for selecting the population with the largest mean from k normal populations. *The Annals of Mathematical Statistics*, pages 174–180, 1964.

Vianney Perchet, Philippe Rigollet, Sylvain Chassang, and Erik Snowberg. Batched bandit problems. In *Conference on Learning Theory*, 2015.

HE Perez, X Hu, and SJ Moura. Optimal charging of batteries via a single particle model with electrolyte and thermal dynamics. In *American Control Conference*, 2016.

Simon F Schuster, Martin J Brand, Philipp Berg, Markus Gleissenberger, and Andreas Jossen. Lithium-ion cell-to-cell variation during battery electric vehicle operation. *Journal of Power Sources*, 297:242–251, 2015.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, 2012.

Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams. Scalable bayesian optimization using deep neural networks. In *International Conference on Machine Learning*, 2015.

Niranjan Srinivas, Andreas Krause, Sham M Kakade, and Matthias Seeger. Gaussian process optimization in the bandit setting: No regret and experimental design. In *International Conference on Machine Learning*, 2010.

Marcelo J Weinberger and Erik Ordentlich. On delayed prediction of individual sequences. *IEEE Transactions on Information Theory*, 48(7):1959–1976, 2002.

Yifan Wu, Andras Gyorgy, and Csaba Szepesvari. On identifying good options under combinatorially structured feedback in finite noisy environments. In *International Conference on Machine Learning*, 2015.

Shengjia Zhao, Enze Zhou, Ashish Sabharwal, and Stefano Ermon. Adaptive concentration inequalities for sequential decision problems. In *Advances in Neural Information Processing Systems*, 2016.