Improved Pure Exploration in Linear Bandits with No-Regret Learning

Mohammadi Zaki[1] , **Avinash Mohan**[2] and **Aditya Gopalan**[1]
[1]Indian Institute of Science, Bangalore
[2]Boston University, MA
mohammadi@iisc.ac.in, avimohan@bu.edu, aditya@iisc.ac.in

Abstract

We study the best arm identification problem in linear multi-armed bandits (LMAB) in the fixed confidence setting ; this is also the problem of optimizing an unknown linear function over a discrete domain with noisy, zeroth-order access. We propose an explicitly implementable and provably order-optimal sample-complexity algorithm for best arm identification. Existing approaches that achieve optimal sample complexity assume either access to a nontrivial minimax optimization oracle (e.g. RAGE and Lazy-TS) or knowledge of an upper-bound on the norm of the unknown parameter vector(e.g. LinGame). Our algorithm, which we call the Phased Elimination Linear Exploration Game (PELEG), maintains a high-probability confidence ellipsoid containing the unknown vector, and uses it to eliminate suboptimal arms in phases, where a minimax problem is essentially solved in each phase using two-player low regret learners. PELEG does not require to know a bound on norm of the unknown vector, and is asymptotically-optimal, settling an open problem. We show that the sample complexity of PELEG matches, up to order and in a non-asymptotic sense, an instance-dependent universal lower bound for linear bandits. PELEG is thus the first algorithm to achieve both order-optimal sample complexity and explicit implementability for this setting. We also provide numerical results for the proposed algorithm consistent with its theoretical guarantees.

1 Introduction

Noisy function optimization over a domain is a basic primitive in machine learning, and a well-known formalization of it is the problem of Probably Approximately Correct (PAC) best arm identification in multi-armed bandits [Even-Dar *et al.*, 2006]. In this problem, a learner is given a set of arms with unknown means. The learner must sequentially test arms and output, as soon as possible with high confidence, a near-optimal arm (where optimality is defined in terms of the largest mean).

Often, the arms (decisions) and their associated rewards are structurally related, allowing for more efficient learning of the rewards and transfer of learnt information. One of the best-known examples of structured decision spaces is the linear bandit, whose arms are vectors (points) in $\mathbb{R}^d$. The reward or function value of an arm is an unknown linear function of its vector representation, and the goal is to find an arm with maximum reward in the shortest possible time by measuring arms' rewards sequentially with noise. This framework models an array of structured online linear optimization problems including adaptive routing [Awerbuch and Kleinberg, 2008b], smooth function optimization over graphs [Valko *et al.*, 2014], subset selection [Kuroki *et al.*, 2019] and, in the nonparametric setting, black box optimization in smooth function spaces [Srinivas *et al.*, 2010], among others.

We study the problem of best arm identification (BAI) in linearly parameterized multi-armed bandits (LMAB), which was first introduced by [Soare *et al.*, 2014] and has received extensive attention ever since. Given a (finite) set of feature vectors in $\mathbb{R}^d$, the goal is to identify the vector with the largest inner product with an unknown weight vector $\theta^* \in \mathbb{R}^d$ using only noisy linear measurements of the form $x^T\theta^* + \eta$.

Our main contribution is to propose a new phased elimination algorithm for this problem. At a high level, the design of the algorithm follows the template of the phased RAGE algorithm of [Fiez *et al.*, 2019], which enjoys information-theoretically optimal (instance-dependent) PAC sample complexity. Their algorithm, however, requires repeated oracle access to a non-trivial minimax oracle in each phase. It is not clear, from a performance standpoint, in what manner (and to what accuracy) this optimization problem should be *practically* solved[1] to enjoy the claimed sample complexity. [Jedra and Proutière, 2020] propose an algorithm (Lazy Track-and-Stop) inspired by ideas from [Garivier and Kaufmann, 2016]. Lazy-TS is shown to achieve optimal expected sample complexity, again in the limit as $\delta \downarrow 0$. However, their algorithm again prescribes a sampling strategy which requires a minimax oracle, the formal implementation of which is left unclear.

[1]For its experiments, the paper implements a (approximate) minimax oracle using a naive variant of the the Frank-Wolfe algorithm (with subgradients instead of gradients) and a heuristic stopping rule, but such a scheme is not theoretically guaranteed to converge for non-smooth optimization, see Sec. 4 for a detailed discussion.

[Wang *et al.*, 2021] propose a scheme where they modify the Frank-Wolfe implementation proposed in [Jedra and Proutière, 2020] to deal with non-smooth optimization objectives and implement it for the case of unstructured and structured bandits. However, their algorithm (which we call FWS) requires *explicit forced exploration* which can prove to be costly especially for the linear bandit setting, where K could be *very* large. We show via simulations (Sec. 6) that this indeed causes FWS to waste many samples on uninformative arms causing the performance to degrade. In contrast, PELEG performs exploration *implicitly* by constructing *correct* confidence sets around the unknown parameter, thus circumventing this problem.

In summary, we give an explicit linear-bandit best-arm identification algorithm with instance-optimal PAC sample complexity and, more importantly, a clearly quantified computational effort by using new techniques. The algorithm relies upon a game-theoretic interpretation of the minimax optimization problem that is at the heart of the instance-based sample complexity lower bound. This yields a concrete adaptive sampling strategy that uses carefully constructed confidence sets for the unknown parameter θ^*. The adaptive sampling strategy is driven by the interaction of two no-regret online learning subroutines that attempt to solve the minimax problem approximately (the seeds of this game-theoretic approach were laid by [Degenne *et al.*, 2019] for the simple (i.e., unstructured) multiarmed bandit problem), obviating the worry of *i)* solving the optimal minimax allocation to a suitable precision and *ii)* making an integer sampling allocation from it by rounding, which occur in the approach of Fiez et al [Fiez *et al.*, 2019]. As will be evident in Sec. 5 this approach (and careful choices of parameters) obviate the need of knowing bounds on the norm of the parameter-space; trading it off with a computable parameter from the arm set.

2 Preliminaries

Problem statement. Given a finite set of arms $\mathcal{X} = \{x_1, \dots, x_K\} \subset \mathbb{R}^d$, the learning agent can play in each round $t = 1, 2, \dots$ an arm $x_t \in \mathcal{X}$, and receives a reward $y(x_t) = {\theta^*}^T x_t + \eta_t$, where the noise η_t is zero-mean assumed to be conditionally $1-$ sub-Gaussian. In this regard, a learning algorithm for the best arm identification problem comprises the following rules: (1) a *sampling rule*, which determines which arm to play based on past data, (2) a *stopping rule*, which controls the end of sampling phase and is a function of the past observations and reward, and (3) a *recommendation rule*, which, after the algorithm stops, outputs a guess $A \in [K]$ for the best arm's index.

An algorithm that stops and returns $A = a^* = \operatorname{argmax}_{a\in[K]} x_a^T \theta^*$ with probability at least $1-\delta$ is said to be *δ-Probably Approximately Correct (δ-PAC)*, and by its sample complexity we mean how soon the algorithm stops.

Assumptions and Notation. We assume that there is a unique optimal arm for the true parameter θ^*, i.e., $\exists a \in [K] : \forall b \in [K] \setminus \{a\} : {\theta^*}^T x_a > {\theta^*}^T x_b$. More generally, for any $\theta \in \mathbb{R}^d$ for which there is a unique optimal arm w.r.t. θ, we define $a^*(\theta) = \operatorname{argmax}_{a\in[K]} \theta^T x_a$. We denote by $\nu^k_{\theta^*}$ the distribution of the reward (observation) obtained by sampling arm $k \in [K]$, i.e., $\forall t \geqslant 1, y(x_t) \sim \nu^k_{\theta^*}$, whenever $x_t = x_k$. Given two probability distributions μ, ν over $\mathbb{R}$, $KL(\mu, \nu)$ denotes the Kullback-Leibler divergence of μ with respect to ν (assuming absolute continuity of the measures). We assume that $\|x_k\|_2 \leqslant 1, \forall x_k \in \mathcal{X}$. Given a positive definite matrix A, $\|x\|_A := \sqrt{x^T A x}$ denotes the matrix norm induced by A. When the matrix is unspecified, this defaults to the standard 2-norm. For any $i \in [K], i \neq a^*$, let $\Delta_i := {\theta^*}^T (x_{a^*} - x_i)$ be the gap between the largest expected reward and the expected reward of (suboptimal) arm x_i. Let $\Delta_{\min} := \min_{i\in[K]} \Delta_i$. We denote $B(z, r)$ as the *closed* ball with center z and radius r. We define $\mathcal{P}_K$ to be the set of all probability mass functions on an alphabet of size K. For the benefit of the reader, we provide a glossary of commonly used symbols in Sec. A.

3 A Game-theoretic Approach towards Optimal Sample Complexity

The phased elimination approach. We first note that a lower bound on the sample complexity of any δ- PAC algorithm for the canonical (i.e., unstructured) bandit setting [Garivier and Kaufmann, 2016] was generalized by [Fiez *et al.*, 2019] and [Zaki *et al.*, 2019] to the linear bandit setting. This result states that any δ-PAC algorithm in the linear setting must satisfy $\mathbb{E}_{\theta^*}[\tau] \geqslant (\log 1/2.4\delta) \frac{1}{T_{\theta^*}} = (\log 1/2.4\delta) \frac{1}{D_{\theta^*}}$, where $T_{\theta^*} := \max_{w\in\mathcal{P}(\mathcal{X})} \min_{\theta:a^*(\theta)\neq a^*(\theta^*)} \sum_{k\in[K]} w_k KL\left(\nu^k_\theta, \nu^k_{\theta^*}\right)$, $D_{\theta^*} := \max_{w\in\mathcal{P}(\mathcal{X})} \min_{x\in\mathcal{X}, x\neq x^*} \frac{\left({\theta^*}^T (x^* - x)\right)^2}{\|x^* - x\|^2_{\left(\sum_{x\in\mathcal{X}} w_x x x^T\right)^{-1}}}$, and $x^* = x_{a^*}$. The bound suggests a natural δ-PAC strategy in hindsight, namely, to sample arms according to the distribution

$$w^* = \operatorname*{argmin}_{w\in\mathcal{P}(\mathcal{X})} \max_{x\in\mathcal{X}\setminus\{x^*\}} \frac{\| x^* - x \|^2_{\left(\sum_{x\in\mathcal{X}} w_x x x^T\right)^{-1}}}{\left((x^* - x)^T \theta^*\right)^2}. \quad (1)$$

Note however that x^* is unknown a priori. [Fiez *et al.*, 2019] design a strategy (RAGE) that attempts to mimic the optimal allocation w^* in phases. In each phase m, RAGE tries to eliminate arms that are 2^{-m}-suboptimal (compared to the empirical best arm), by solving (1) with a plugin estimate of θ^*. The resulting fractional allocation, is passed through a separate discrete rounding procedure, to give an integer pull count distribution to ensure that all surviving arms' mean differences are estimated with high precision and confidence. This approach, however, is based crucially on solving minmax problems of the form (1). Though the inner (max) function is convex as a function of w on the probability simplex [Zaki *et al.*, 2019, Lemma 1], it is *non-smooth*, and it is not clear how, and to what precision, it must be solved to satisfy the δ-PAC guarantee. using ideas from games between no-regret online learners with optimism, as introduced in [Degenne *et al.*, 2019] for unstructured bandits, we are able to sidestep this obstacle *entirely.*

From Pure-exploration Games to δ-PAC Algorithms. We briefly explain some of the insights in [Degenne *et al.*, 2019] that we leverage to design an explicit linear bandit-δ-PAC algorithm with low computational complexity. We

Algorithm	Sample Complexity	Remarks
$\mathcal{XY}$-static [Soare *et al.*, 2014]	$\mathcal{O}\Big(\frac{d}{\Delta_{\min}}(\ln\frac{1}{\delta}+\ln K+\ln\frac{1}{\Delta_{min}})+d^2\Big)$	Static allocation, worst-case optimal Dependence on d cannot be removed
LinGapE [Liyuan Xu, 2017]	$\mathcal{O}\Big(dH_0\log\Big(dH_0\log\frac{1}{\delta}\Big)\Big)$	Fully adaptive, sub-optimal in general.
ALBA [Tao *et al.*, 2018]	$\mathcal{O}\Big(\sum_{i=1}^{d}\frac{1}{\Delta^2_{(i)}}\ln\Big(\frac{K}{\delta}+\ln\frac{1}{\Delta_{min}}\Big)\Big)$	Fully adaptive, sub-optimal in general (see [Fiez *et al.*, 2019])
RAGE [Fiez *et al.*, 2019]	$\mathcal{O}\Big(\frac{1}{D_{\theta^*}}\log 1/\Delta_{\min}\log\Big(\frac{K^2\log^2 1/\Delta_{min}}{\delta}\Big)\Big)$	Instance-optimal, but **Minimax oracle required**
LinGame [Degenne *et al.*, 2020]	$\mathcal{O}\left(\frac{\log 1/\delta+K^2 d\sqrt{\log 1/\delta}+S^2(\log 1/\delta)^{3/4}}{D_{\theta^*}}\right)$	**Requires knowledge of a upperbound on** $\|\theta^*\|_2$ Optimal lower bound is achieved only in the limit as $\delta\downarrow 0$
Lazy Track-and-Stop [Jedra and Proutière, 2020]	$\mathcal{O}\left(\frac{\log(1/\delta)}{D_{\theta^*}}\right)+o(\log(1/\delta))$	Asymptotically-optimal, but **Minimax oracle required**
PELEG (this paper)	$\mathcal{O}\left(\frac{\log_2(1/\Delta_{min})}{D_{\theta^*}}\log\left(\left(\log_2(1/\Delta_{min})\right)^2K^2/\delta\right)\right)$	**Instance-optimal,** **Explicitly implementable** **Does not require knowledge of ANY parameter that cannot be gleaned from the given arm set** **Access to minimax oracle not required.**

Table 1: Comparison of Sample complexities achieved by various algorithms for LMAB in the literature. Here, S denotes a bound on $|\theta^*|_2$ and H_0 is the solution to an offline optimization problem in LinGapE.

consider the related geometrical optimization problem of fitting an ellipsoid inside a polygon, both centered at the origin, namely $\max_{w\in\mathcal{P}_K}\min_{\lambda\in\cup_{x\in\mathcal{X}_m}\mathcal{C}_m(x)}\|\lambda\|^2_{\sum_{x\in\mathcal{X}}w_x xx^T}$, where $\mathcal{C}_m(x)$ is a union of halfspaces, defined in Alg. 1. By subsequently reducing the size of the polygon (i.e., value of ε_m in alg 1) we achieve a reduction in size of the confidence-ellipsoid. To this end, consider the two-player, zero-sum "pure exploration" game in which the MAX player (or column player) plays an arm $k_t\in[K]$ while the MIN (or row) player chooses an $\lambda_t\in\cup_{x\in\mathcal{X}_m}\mathcal{C}_m(x)$. MAX then receives a payoff of $\|\lambda_t\|^2_{x_k x_k^T}$ from MIN. With MAX moving first and playing a mixed strategy $w\in\mathcal{P}(\mathcal{X})$, the value of the game becomes $\max_{w\in\mathcal{P}_K}\min_{\lambda\in\cup_{x\in\mathcal{X}_m}\mathcal{C}_m(x)}\|\lambda\|^2_{\sum_{x\in\mathcal{X}}w_x xx^T}$. As shown in Appendix F, this quantity is directly related to $1/D_{\theta^*}$.

We crucially employ no-regret online learners to solve this Pure Exploration Game. More precisely, no-regret learning with the well-known Exponential Weights rule/Negative-entropy mirror descent algorithm [Shalev-Shwartz and others, 2011] on the one hand, and a best-response convex programming subroutine on the other, *directly* provides a sampling strategy that obviates the need for separate allocation optimization and rounding for sampling as in [Fiez *et al.*, 2019]. An advantage of our approach (inspired by [Degenne *et al.*, 2019]) is that we only use a best-response oracle to solve for $T_{\theta^*}(w)$, which gives us a computational edge over [Fiez *et al.*, 2019] who employ the more computationally costly max-min oracle to solve $T_{\theta^*}(w)$, or, equivalently, D_{θ^*}.

4 Proposed Algorithm and Sample Complexity Bound

Our algorithm, that we call "Phased Elimination Linear Exploration Game" (PELEG), is presented in detail in Algorithm 1. PELEG proceeds in phases with each phase comprising multiple rounds, maintaining a set of *active* arms $\mathcal{X}_m$ for testing during phase m. A least squares estimate $\widehat{\theta}_m$ of θ^* is used to estimate the mean reward of active arms and, at the end of phase m, every active arm with a plausible reward more than $\approx 2^{-m}$ below that of some arm in $\mathcal{X}_m$ is eliminated.

Suppose $\mathcal{S}_m:=\Big\{x\in\mathcal{X}\setminus\{x^*\}:{\theta^*}^\top(x^*-x)<1/2^m\Big\}$. If we can ensure that $\mathcal{X}_m\subset\mathcal{S}_m$ in every Phase $m\geqslant 1$, then PELEG will terminate within $\lceil\log_2(1/\Delta_{\min})\rceil$ phases, where $\Delta_{\min}=\min_{x\neq x^*}{\theta^*}^\top(x^*-x)$. This statement is proved in Corollary 2 in the Supplementary Material.

Note 1. *The set $\mathcal{X}_B$ in the input to Algorithm 1 is a subset of arms of cardinality d (value of ambient dimension), which is played during the burn-in period in each phase of the algorithm. We need to compute it only once offline in order to run the entire algorithm, and it is desirable to have $\lambda_{min}(\sum_{x\in\mathcal{X}_B}xx^\top)$ as large as possible. A principled way in which this subset can be chosen is, for instance, as a barycentric spanner of the arm set $\mathcal{X}$, which can be computed in time $O(d^2)$ [Awerbuch and Kleinberg, 2008a].*

Note 2. *$C=\lambda_{min}(\sum_{x\in\mathcal{X}_B}xx^\top)$ can be interpreted as a measure of how much information arms give about each other. If the arms in $\mathcal{X}_B$ are near-collinear, this eigenvalue will be very small and it is hard to transfer information across arms. Conversely, if the arms in $\mathcal{X}_B$ are highly 'spread out' this eigenvalue will be large and sampling an arm will give more information about the best arm [Zaki* et al.*, 2019].*

Approximating the minimax problem using no regret learners. We formulate the minimax problem discussed in Sec. 3 as a two player, zero-sum game. We solve the game sequentially, converging to its Nash equilibrium by invoking the use of the EXP-WTS algorithm [Cesa-Bianchi and Lugosi, 2006]. Specifically, in each round t in a phase, PELEG supplies EXP-WTS (*MAX* player) with an appropriate loss function l^{MAX}_{t-1} and receives the requisite sampling distribution w_t . This w_t is then fed to the second no-regret learner (*MIN* player) – a best response subroutine – that in turn returns a 'competing' model $\lambda\in\mathbb{R}^d$ (i.e., a model whose best arm is different but whose distribution is not too different) to focus on. This is a minimization of a quadratic function over a union of finitely many convex sets (halfspaces intersecting a ball) which can be transparently implemented in polynomial time. Once the sampling distribution is found, we use an efficient *tracking* procedure, from [Degenne *et al.*, 2020] (see Lemma 6), to ensure that $\sum_{s=1}^t w_s^k-(\log K)\leqslant n_t^k\leqslant\sum_{s=1}^t w_s^k+1$ for every $t\geqslant 1$. This procedure avoids the need for explicit rounding techniques that plagues [Fiez *et al.*, 2019].

Finally, in each phase m, we need to sample arms often enough to (i) construct confidence intervals of size at most

Algorithm 1 Phased Elimination Linear Exploration (PELEG)

Input: $\mathcal{X}$ (Arm set), $\mathcal{X}_B$ (set of d arms to be played in the burn-in period), δ (probability of error).
Init: $m \leftarrow 1, \mathcal{X}_m \leftarrow \mathcal{X}$.
Output: Guess for the best arm.

1: $C \leftarrow \lambda_{min}\left(\sum_{x\in\mathcal{X}_B} xx^{\mathsf{T}}\right)$.
2: $\tilde{S} \leftarrow \sqrt{\frac{d}{C}}$.
3: **while** $\{|\mathcal{X}_m| > 1\}$ **do**
4: $\quad\delta_m \leftarrow \frac{\delta}{m^2}$. {*Phase starts* }
5: $\quad\varepsilon_m \leftarrow \left(\frac{1}{2}\right)^{m+1}$.
6: $\quad r_m \leftarrow \sqrt{8\log\left(K^2/\delta_m\right)}$.
7: $\quad l_m \leftarrow \left\lceil \frac{r_m^2}{\tilde{S}^2 C} \right\rceil$.
8: $\quad\mathcal{C}_m(x) \leftarrow \{\lambda \in \mathbb{R}^d : \exists x' \in \mathcal{X}_m, x' \neq x|$ $\hookrightarrow\ \lambda^{\mathsf{T}}x' \geqslant \lambda^{\mathsf{T}}x + \varepsilon_m\}$, for $x \in \mathcal{X}_m$.
9: $\quad t \leftarrow 0, n_0^k \leftarrow 0, \forall k \in [K]$.
10: $\quad$Play each arm in $\mathcal{X}_B$ l_m times and collect rewards. {*Burn-in period*}
11: $\quad\forall x \in \mathcal{X}_B, n_x^d = l, V_{dl_m}^m \leftarrow l_m \sum_{x\in\mathcal{X}_B} xx^{\mathsf{T}}, t \leftarrow dl_m$.
12: $\quad$Initialize $\mathcal{A}_m^{MAX} \equiv EXP - WTS$ with expert set $\{\widehat{e}_1, \cdots, \widehat{e}_K\} \subset \mathbb{R}^K$ and loss function $l_{t-1}^{MAX}()$.{MAX *player: Exponential Weights*}
13: $\quad$**while** $\left\{ \min_{\lambda \in \bigcup_{x\in\mathcal{X}_m} \mathcal{C}_m(x)\cap B(0,\tilde{S})} \|\lambda\|_{V_t^m}^2 \leqslant r_m^2 \right\}$ **do** {*Phase Stopping Criterion*}
14: $\quad\quad t \leftarrow t + 1$
15: $\quad\quad$Get w_t from $\mathcal{A}_m^{MAX}$ and form the matrix $W_t = \sum_{k=1}^K w_t^k x_k x_k^{\mathsf{T}}$.
16: $\quad\quad\lambda_t \leftarrow \operatorname{argmin}_{\lambda\in\cup_{x\in\mathcal{X}_m}\mathcal{C}_m(x)\cap B\left(0,\tilde{S}\right)} \|\lambda\|_{W_t}^2$.{$MIN$ *player: Best response*}
17: $\quad\quad$**for** $k \in [K]$ **do**
18: $\quad\quad\quad U_t^k := \left(\lambda_t^{\mathsf{T}} x_k\right)^2$
19: $\quad\quad$**end for**
20: $\quad\quad$Construct loss function $l_t^{MAX}(w) = -w^{\mathsf{T}}U_t$
21: $\quad\quad$Play arm $k_t = \operatorname{argmax}_{k\in[K]} \sum_{s=1}^t w_s^k - n_{t-1}^k$ {*Tracking the play distribution provided by the MAX player*}
22: $\quad\quad n_t^{k_t} \leftarrow n_t^{k_t} + 1$
23: $\quad\quad$Collect sample $Y_t = \theta^{*\mathsf{T}} x_{k_t} + \eta_t$
24: $\quad\quad V_t^m = V_{t-1}^m + x_{k_t} x_{k_t}{}^{\mathsf{T}}$.
25: $\quad$**end while**
26: $\quad N_m \leftarrow t$
27: $\quad\widehat{\theta}_m \leftarrow \left(V_{N_m}^m\right)^{-1}\left(\sum_{s=1}^{N_m} Y_s x_{k_s}\right)$ {*Least-squares estimate of* θ^*}
28: $\quad\widehat{x}_{m+1} \leftarrow \operatorname{argmax}_{x\in\mathcal{X}_m} \widehat{\theta}_m^{\mathsf{T}} x$.
29: $\quad\mathcal{X}_{m+1} \leftarrow \left\{x \in \mathcal{X}_m | \widehat{\theta}_m^{\mathsf{T}}(\widehat{x}_{m+1} - x) \leqslant 2^{-(m+2)}\right\}$. { *Elimination of suboptimal arms* }
30: $\quad m \leftarrow m + 1$. {*Phase ends*}
31: **end while**
32: **return** $\mathcal{X}_m$ {*Output the sole surviving arm*}

$2^{-(m+1)}$ around $(x - x')^{\top}\theta^*$, $\forall x, x' \in \mathcal{X}_m$, (ii) ensure that $\mathcal{X}_m \subset \mathcal{S}_m$ and (iii) that $x^* \in \mathcal{X}_m$. In Sec. E, we prove a Key Lemma (whose argument is discussed in Sec. 5) to show that our novel *Phase Stopping Criterion* ensures this with high probability. It is worth remarking that the use of phased elimination template of [Fiez *et al.*, 2019] eliminates the need to use more complex, self-tuning online learners like AdaHedge [de Rooij *et al.*, 2014], as used in [Degenne *et al.*, 2019] and more recently [Degenne *et al.*, 2020], in favor of the simpler Exponential Weights (Hedge) algorithm. The main theoretical result of this paper is the following performance guarantee. A more detailed version is in Appendix.

Theorem 1 (Sample Complexity of PELEG)**.** *With probability at least* $1-\delta$*, the worst-case sample complexity of PELEG, which we denote as* τ*, is bounded as*

$$\tau \leqslant C_1 \frac{\log_2\left(1/\Delta_{min}\right)}{D_{\theta^*}} \log\left(\left(\log_2\left(1/\Delta_{min}\right)\right)^2 K^2/\delta\right) + C_2 \log_2\left(1/\Delta_{min}\right) \log\left(\frac{K^2\left(\log_2\left(1/\Delta_{min}\right)\right)^2}{\delta}\right), \tag{2}$$

where C_1 *and* C_2 *are universal constants.*

We sketch the arguments behind the result in Sec. 5; the full proof appears in Appendix F.

Note 3. *As explained in Sec. 3, the optimal (oracle) allocation requires* $\mathcal{O}\left(\frac{1}{D_{\theta^*}} \log \frac{1}{\delta}\right)$ *samples. Comparing this with* (2)*, we see that our algorithm is instance-optimal up to logarithmic factors.*

5 Sketch of Sample Complexity Analysis

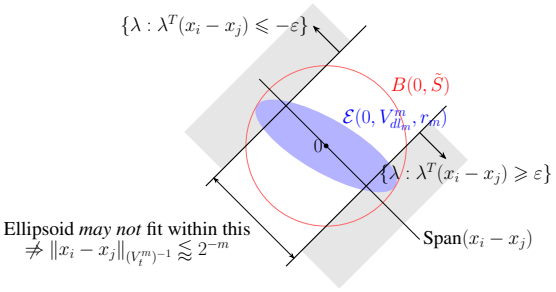
This section outlines the proof of the δ-PAC sample complexity of Algorithm 1 (Theorem 1) and describes the main ideas and challenges involved in the analysis.

As described, the algorithm operates in phases, with the goal of the m-th phase being to eliminate, with high probability, arms which are roughly 2^{-m}-suboptimal w.r.t. other surviving arms and hence whose reward gaps are at least 2^{-m}. This can be guaranteed if all available arms (i.e., directions) are sampled sufficiently enough to reduce the uncertainty in estimating mean differences by about 2^{-m}. To achieve such a sampling pattern, alternating no-regret learners are used in the central **while** loop which is terminated according to a phase stopping criterion (Step 13 of PELEG).

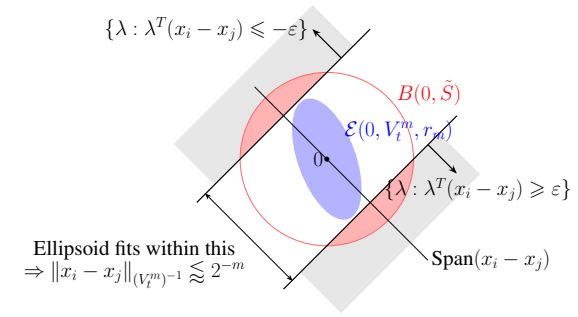
At a high level, the proof of Theorem 1 involves two main parts: (1) a correctness argument for the phase stopping criterion that eliminates suboptimal arms, and (2) a bound for the length of the central **while** loop, which, when added across all phases, gives the overall sample complexity bound.

1. Uncertainty control along relevant directions. At the heart of the analysis is the following result which guarantees that upon termination of the central while loop, the uncertainty in estimating all differences of means among the surviving (i.e., non-eliminated) arms remains bounded.

Lemma 1 (Key Lemma)**.** *After each phase* $m \geqslant 1$*,* $\max_{x,x'\in\mathcal{X}_m, x\neq x'} \|x - x'\|_{\left(V_{N_m}^m\right)^{-1}}^2 \leqslant \frac{\left(\left(\frac{1}{2}\right)^{m+1}\right)^2}{8\log K^2/\delta_m}$.



(a) After the *burn-in* period: The 2-norm of each vector in the blue ellipsoid is smaller than $\tilde{S}$. Its extent along $(x_i - x_j)$ *may exceed the gap* between the hyperplanes (parallel black lines).



(b) After end of phase m: The blue ellipsoid separates from the halfspaces intersecting the ball (red) by *staying within the ball*. In this case its extent along $(x_i - x_j)$ is within the gap between the hyperplanes (parallel black lines).

Figure 1: The phase stopping condition in Algorithm 1 ensures that $\|x_i - x_j\|_{(V_t^m)^{-1}} \lessapprox 2^{-m}$ after phase m.

Remarks. In each phase $m \geqslant 1$, we have a *burn-in* period, where each arm in the set $\mathcal{X}_B$, is played l_m number of times where l_m is chosen carefully. At a high level the goal of the burn-in period is to ensure that the ellipsoid $\mathcal{E}(0, V_t^m, r_m)$, becomes 'small' (see Figure 1a). More precisely, we ensure $\mathcal{E}(0, V_t^m, r_m) \subset B(0, \tilde{S})$. This step is crucial in the analysis, as it helps ensuring that the losses passed to the no-regret learners remain bounded.

Lemma 2. *For any phase $m \geqslant 1$, at the end of the burn-in period (line 10, in which each arm $x \in \mathcal{X}_B$ is played $l_m := \left\lceil \frac{r_m^2}{\tilde{S}^2 C} \right\rceil$ times), we have $\mathcal{E}(0, V_t^m, r_m) \subseteq B(0, \tilde{S})$.*

After this, the algorithm enters the while loop. Phase m ends at round t when the ellipsoid $\mathcal{E}(0, V_t^m, r_m)$, with center 0 and shape according to the arms played in the phase so far, becomes small enough to avoid intersecting the half spaces $\mathcal{C}_m(x)$, for all surviving arms x, within the ball $\cap B(0, \tilde{S})$ (Phase Stopping Criterion of the algorithm) which is required to keep loss functions bounded for no-regret properties.

We show a simpler situation in Figure 1b when there are only two arms x_i, x_j remaining when phase m starts. $\mathcal{C}_m(x_i) \equiv \mathcal{C}_m(x_i; \varepsilon_m)$ and $\mathcal{C}_m(x_j; \varepsilon_m)$ with $\varepsilon_m \approx 2^{-m}$ are halfspaces, denoted in gray, that intersect the ball $B(0, \tilde{S})$ in the areas colored red. In this situation, the ellipsoid V_t^m, shaded in blue, has just broken away from the red regions *in the interior of the ball*. Because its extent in the direction $x_i - x_j$ lies within the strip between the two hyperplanes bounding $\mathcal{C}_m(i), \mathcal{C}_m(j)$, it can be shown (see proof of lemma in appendix) that $\|x_i - x_j\|_{(V_t^m)^{-1}}$ is small enough to not exceed roughly 2^{-m}.

2. Bounding the number of arm pulls in a phase. The main bound on the length of the central **while** loop (l. 13) is:

Lemma 3 (Phase length bound)**.** *Let $B_m := \min_{w \in \mathcal{P}_K} \max_{x,x' \in \mathcal{X}_m, x \neq x'} \|x - x'\|^2_{W^{-1}}$, where $W = \sum_{k=1}^K w^k x_k x_k^T$. There exists δ_0 such that $\forall \delta < \delta_0$, the length N_m of any phase m is bounded as :*

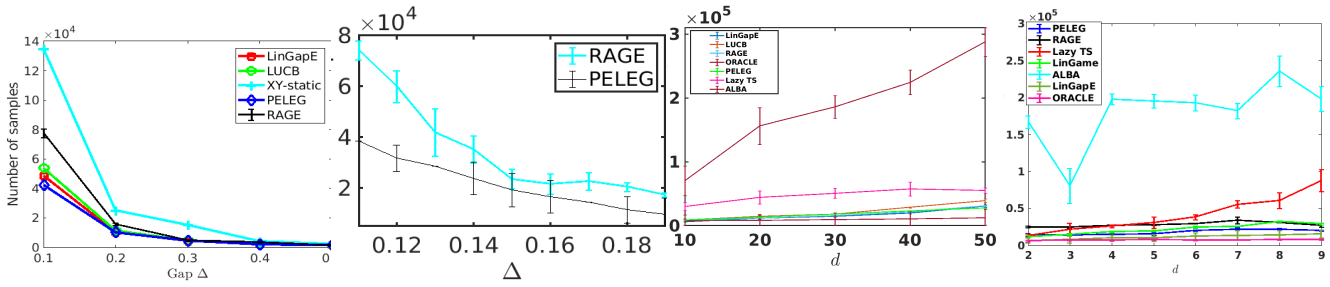
$$N_m \leqslant \max\left\{4B_m \left(2^{m+1}\right)^2 r_m^2 + 1, dl_m\right\}$$
$$\leqslant \left(4B_m \left(2^{m+1}\right)^2 + 2\right) r_m^2.$$

To prove this we use the no-regret property of both the best-response (*MIN*) and the Exponential Weights (*MAX*) learners (the full proof appears in the Appendix D). Here we essentially approximate the saddle point minimax optimization problem B_m (mentioned in the above item) to a certain level of accuracy. The details of the *MIN* and *MAX* learners are deferred to Appendix B. Finally we use the concavity of the function $t \mapsto \sqrt{t}$ to obtain the above bound on phase-length.

Note 4. *The careful choice of $\tilde{S}$ (on line 2 of the algorithm) plays a crucial rule in the phase-length bound. It provides just enough exploration at the beginning of each phase, so that the regret of the online learning algorithm (i.e., $\mathcal{A}^{MAX}$) is controllable, while not being too large to effect the sample complexity. Moreover, the minimum eigenvalue $C = \lambda_{min}\left(\sum_{x \in \mathcal{X}_B} xx^\intercal\right)$ used by the algorithm does not affect the sample complexity when δ is small. More details can be found in Appendix Sec. D.*

3. Correct elimination of sub-optimal arms at the end of each phase. Here we argue the progress made after every phase and the correctness of the algorithm. Let $\mathcal{S}_m := \left\{x \in \mathcal{X} : \theta^{*T}(x^* - x) < \frac{1}{2^m}\right\}$. Let $B_m^* := \min_{w \in \mathcal{P}_K} \max_{(x,x') \in \mathcal{S}_m^2, x \neq x'} \|x - x'\|^2_{W^{-1}}$, where $W = \sum_{k=1}^K w^k x_k x_k^\top$. We show in Appendix (Sec. E) that with high probability, after the end of any phase $m \geqslant 1$, $\{x^* \in \mathcal{X}_{m+1}\}$ and $\{\mathcal{X}_{m+1} \subseteq \mathcal{S}_{m+1}\}$. To show this, we use the sub-Gaussian property of rewards obtained in every round, along with Lemma 1 which holds for every phase $m \geqslant 1$ and the elimination criteria (Step 29). We refer the reader to Sec. E for the details.

We finally put these ingredients together, along with Prop 1 (in the appendix), to show the final sample complexity result (Appendix F).



(a) (b) (c) (d)

Figure 2: Sample complexity performance of LMAB algorithms for 3 different settings: Standard bandit (Figs. a, b), Unit Sphere (Fig. c) and Standard bandit with confounding arm (Fig. d). The means reported are computed over 30 trials with 1-standard deviation error bars.

6 Experiments

We numerically evaluate PELEG against the algorithms $\mathcal{XY}$-static [Soare *et al.*, 2014], LUCB [Kalyanakrishnan *et al.*, 2012], ALBA [Tao *et al.*, 2018], LinGapE [Liyuan Xu, 2017], RAGE [Fiez *et al.*, 2019], Lazy-TS [Jedra and Proutière, 2020] and LinGame [Degenne *et al.*, 2020], for 3 common benchmark settings. The oracle lower bound is also calculated. In our implementation, we ignore the term $B(0, \tilde{S})$ in the phase stopping criterion; this has the advantage of making the criterion verifiable in closed form. We simulate independent, $\mathcal{N}(0,1)$ observation noise in each round. All results reported are averaged over 30 trials. We also empirically observe a 100% success rate in identifying the best arm, although a confidence value of $\delta = 0.01$ is passed in all cases.

Setting 1: Standard bandit. The arm set is the standard basis $\{e_1, e_2, \ldots, e_5\}$ in 5 dimensions. The unknown parameter θ^* is set to $(\Delta, 0, \ldots, 0)$, where $\Delta > 0$, with Δ swept across $\{0.1, 0.2, 0.3, 0.4, 0.5\}$. As noted in [Liyuan Xu, 2017], for Δ close to 0, $\mathcal{XY}$-static's essentially uniform allocation is optimal, since we have to estimate all directions equally accurately. However, PELEG performs better (Fig. 2(a)) due to being able to eliminate suboptimal arms earlier instead of uniformly across all arms. Fig. 2(b) compares PELEG and RAGE in the smaller window $\Delta \in [0.11, 0.19]$, where PELEG is found to perform better than RAGE.

Setting 2: Unit sphere. The arms set comprises of 100 vectors sampled uniformly from the surface of the unit sphere $\mathbb{S}^{d-1}$. We pick the two closest arms, say u and v, and then set $\theta^* = u + \gamma(v - u)$ for $\gamma = 0.01$, making u the best arm. We simulate all algorithms over dimensions $d = 10, 20, \ldots, 50$. This setting was introduced in [Tao *et al.*, 2018], and PELEG is uniformly competitive with the other algorithms (Fig. 2(c)).

Setting 3: Standard bandit with a confounding arm . As in [Soare *et al.*, 2014], we instantiate d canonical basis arms $\{e_1, e_2, \ldots, e_d\}$ and an additional arm $x_{d+1} = (\cos(\omega), \sin(\omega), 0, \ldots, 0)$, $d = 2, \ldots, 10$, with $\theta^* = e_1$ so that the first arm is the best arm. By setting $0 < \omega \ll 1$, the $(d+1)$st arm becomes the closest competitor. Here, the performance critically depends on how much an agent focuses on comparing 1st and $(d+1)$th arm. LinGapE performs very well in this setting, and PELEG is competitive with it (Fig. 2(d)).

Note 5. *In Setting 3, we empirically observe that at the time of stopping in the case of $d = 2$, $w_t(d+1) \approx 0$. This is in accordance with the optimal allocation for the case when $0 < \omega < \pi/2$, as shown by explicit calculation in [Zaki* et al.*, 2019]. More details can be found in Appendix G.*

Setting 4: Many arms with moderate gaps. This setting is similar to the 'many arms' setting in [Fiez *et al.*, 2019]. We choose $d = 2$, with $x^* = \theta^* = (1, 0)^\top$. The second arm, which happens to be the most informative arm, is chosen as $(0, 1)^\top$. The remaining $K - 2$ arms are chosen as $(\cos(\pi/6 + \varphi(j)), \sin(\pi/6 + \varphi(j)))$ where each $\varphi(j) \sim \mathcal{N}(0, 0.1)$, $j = 1, 2, 3, \ldots, K-2$. We choose K to be large : $K \in \{600, 700\}$. We compare PELEG with FWS [Wang *et al.*, 2021] in this setting, and show the number of times each arm is pulled in both the cases. From [Zaki *et al.*, 2019], it is known that it is optimal and sufficient to pull only the first two arms. This experiment shows that FWS wastes samples by choosing uninformative arms causing the performance to suffer. We set $\delta = 0.05$, and the results are averaged over 30 trials each.

$K = 600$	PELEG	FWS
Sample Complexity	4924 ± 156	7172 ± 567
# of samples of arms $\{3, \ldots, K\}$	142 ± 49	2392 ± 74
$K = 700$	PELEG	FWS
Sample Complexity	3744 ± 125	7760 ± 739
# of samples of arms $\{3, \ldots, K\}$	656 ± 67	2792 ± 147

Table 2: Average sample complexity performance of PELEG and FWS on Setting 4 (with 1-standard deviation). The second row shows that FWS wastes a lot of sample on unimportant arms even on relatively easier problems with moderate gaps.

Acknowledgments

Mohammadi Zaki was supported in part by Microsoft Travel Grants, MS RESEARCH LAB INDIA PVT (Ref # 100316111), Aerospace Network Research Consortium (ANRC) Grant on Airplane IOT Data Management and ACM-India/IARCS Travel Grants.

References

[Awerbuch and Kleinberg, 2008a] B. Awerbuch and Robert D. Kleinberg. Online linear optimization and adaptive routing. *J. Comput. Syst. Sci.*, 74:97–114, 2008.

[Awerbuch and Kleinberg, 2008b] Baruch Awerbuch and Robert Kleinberg. Online linear optimization and adaptive routing. *Journal of Computer and System Sciences*, 74(1):97–114, 2008.

[Cesa-Bianchi and Lugosi, 2006] Nicolò Cesa-Bianchi and Gábor Lugosi. *Prediction, learning, and games.* Cambridge University Press, 2006.

[de Rooij *et al.*, 2014] Steven de Rooij, Tim van Erven, Peter Grunwald, and Wouter Koolen. Follow the leader if you can, hedge if you must. *Journal of Machine Learning Research*, 15:1281–1316, 2014.

[Degenne *et al.*, 2019] Rémy Degenne, Wouter M. Koolen, and Pierre Ménard. Non-asymptotic pure exploration by solving games. In *NeurIPS*, 2019.

[Degenne *et al.*, 2020] Rèmy Degenne, Pierre Mènard, Xuedong Shang, and Michal Valko. Gamification of pure exploration for linear bandits. Proceedings of the 37th International Conference on Machine Learning. PMLR, 2020.

[Even-Dar *et al.*, 2006] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *J. Mach. Learn. Res.*, 7:1079–1105, December 2006.

[Fiez *et al.*, 2019] Tanner Fiez, Lalit Jain, Kevin G Jamieson, and Lillian Ratliff. Sequential experimental design for transductive linear bandits. In *Advances in Neural Information Processing Systems*, pages 10666–10676, 2019.

[Garivier and Kaufmann, 2016] Aurélien Garivier and Emilie Kaufmann. Optimal best arm identification with fixed confidence. In *Conference On Learning Theory*, pages 998–1027, 2016.

[Jedra and Proutière, 2020] Yassir Jedra and A. Proutière. Optimal best-arm identification in linear bandits. *NeurIPS*, abs/2006.16073, 2020.

[Kalyanakrishnan *et al.*, 2012] Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. Pac subset selection in stochastic multi-armed bandits. In *ICML*, 2012.

[Kuroki *et al.*, 2019] Yuko Kuroki, Liyuan Xu, Atsushi Miyauchi, Junya Honda, and Masashi Sugiyama. Polynomial-time algorithms for combinatorial pure exploration with full-bandit feedback. *arXiv preprint arXiv:1902.10582*, 2019.

[Liyuan Xu, 2017] Masashi Sugiyama Liyuan Xu, Junya Honda. Fully adaptive algorithm for pure exploration in linear bandits. 2017.

[Shalev-Shwartz and others, 2011] Shai Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and trends in Machine Learning*, 4(2):107–194, 2011.

[Soare *et al.*, 2014] Marta Soare, Alessandro Lazaric, and Remi Munos. Best-arm identification in linear bandits. *Advances in Neural Information Processing Systems 27*, pages 828–836, 2014.

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

[Tao *et al.*, 2018] Chao Tao, Saúl Blanco, and Yuan Zhou. Best arm identification in linear bandits with linear dimension dependency. volume 80 of *Proceedings of Machine Learning Research*, pages 4877–4886. PMLR, 2018.

[Valko *et al.*, 2014] Michal Valko, Rémi Munos, Branislav Kveton, and Tomáš Kocák. Spectral bandits for smooth graph functions. In *International Conference on Machine Learning*, pages 46–54, 2014.

[Wang *et al.*, 2021] Po-An Wang, Ruo-Chun Tzeng, and Alexandre Proutiere. Fast pure exploration via frank-wolfe. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[Zaki *et al.*, 2019] Mohammadi Zaki, Avinash Mohan, and Aditya Gopalan. Towards optimal and efficient best arm identification in linear bandits. *arXiv preprint arXiv:1911.01695*, 2019.