Fixed-Budget Best-Arm Identification in Sparse Linear Bandits

Anonymous Author(s)
Affiliation
Address
email

Abstract

We study the best-arm identification problem in sparse linear bandits under the fixed-budget setting. In sparse linear bandits, the unknown feature vector θ^* may be of large dimension d, but only a few, say $s \ll d$ of these features have non-zero values. We design a two-phase algorithm, Lasso and Optimal-Design- (Lasso-OD) based linear best-arm identification. The first phase of Lasso-OD leverages the sparsity of the feature vector by applying the thresholded Lasso introduced by Zhou (2009), which estimates the support of θ^* correctly with high probability using rewards from the selected arms and a judicious choice of the design matrix. The second phase of Lasso-OD applies the OD-LinBAI algorithm by Yang and Tan (2022) on that estimated support. We derive a non-asymptotic upper bound on the error probability of Lasso-OD by carefully choosing hyperparameters (such as Lasso's regularization parameter) and balancing the error probabilities of both phases. For fixed sparsity s and budget T, the exponent in the error probability of Lasso-OD depends on s but not on the dimension d, which yields a significant performance improvement for sparse and high-dimensional linear bandits. Furthermore, we show that Lasso-OD is almost minimax optimal in the exponent. Finally, we provide numerical examples to demonstrate the significant performance improvement over the existing algorithms for the non-sparse scenario such as OD-LinBAI, BayesGap, Peace, LinearExploration, and GSE.

1 Introduction

The stochastic multi-armed bandit (MAB) is a model that provides a mathematical formulation to study the sequential design of experiments and exploration-exploitation trade-off, where a learner pulls an arm out of a total K and receives a reward drawn from a fixed and unknown distribution according to the chosen arm. This model has several applications including online advertising, recommendation systems, and drug tests. While in the standard reward model, the arms are uncorrelated with each other, stochastic linear bandits introduced in [1] generalize the standard model by associating each arm with a d-dimensional feature vector and the reward is equal to the inner product between the feature vector and a unknown global parameter. Therefore, the arms are correlated in linear bandits, meaning that pulling an arm gives information about the rewards of some other arms.

Most prior work including [1–5] on MABs focuses on *regret minimization*, where the goal is to maximize the cumulative reward after T arm pulls by optimizing the trade-off between exploration and exploitation. Recently, the *pure exploration* setting has drawn attention from researchers. One example of pure exploration is the *best-arm identification* (BAI) problem, where the goal is to identify the arm with the largest mean reward. The BAI problem is studied in two settings: (1) the fixed-budget setting considers a budget $T \in \mathbb{N}$ and aims to minimize the probability of failing to identify the best in at most T arm pulls; (2) the fixed-confidence setting considers a confidence level $\delta \in (0, 1)$ and

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

aims to minimize the average number of arm pulls while identifying the best arm with probability at least $1-\delta$.

For the standard reward model with uncorrelated arms, the works in [6–8] and [9, 10] consider the BAI problem in the fixed-confidence and fixed-budget settings, respectively. For the linear model, the works in [11–16] develop several algorithms under the fixed-confidence setting. For the linear model under the fixed-budget setting, Hoffman *et al.* [17] develop the first algorithm, BayesGap, which is a gap-based exploration algorithm using a Bayesian approach. Katz-Samuels *et al.* [18] develop the Peace algorithm that has equally-sized rounds, where the arm-pulling strategy within each round is based on the Gaussian width of the underlying arm set. Alieva *et al.* [19] develop LinearExploration that exploits the linear structure of the model and is robust to unknown levels of observation noise and misspecification in the linear model. Yang and Tan [20] develop Optimal-Design-Based Linear Best Arm Identification (OD-LinBAI), which also employs almost equally-sized rounds, but the arm-pulling strategy within each round is based on the G-optimal design. In the first round, OD-LinBAI aggressively eliminates all empirically suboptimal arms except the top $\frac{d}{2}$ arms; in the subsequent rounds, half of the remaining arms are eliminated in each round until a single arm remains. Azizi *et al.* [21] develop the Generalized Successive Elimination (GSE) algorithm that has similar principles as OD-LinBAI with the difference that GSE eliminates the half of the remaining arms in all rounds. Among these algotithms, only OD-LinBAI is shown to be asymptotically minimax optimal.

In many practical applications of MABs, there is a large number of features available to the learner, but only a few of these features significantly affect the value of the reward of an arm. Sparse linear bandits are a mathematical abstraction of this phenomenon by considering that the d-dimensional unknown parameter θ^* in the linear model has only s nonzero values, i.e., $\|\theta^*\|_0 = s$, where s is much smaller than d. The performance in the MAB problems (e.g., cumulative regret, probability of identification error) usually deteriorates as the ambient dimension d increases. Therefore, the goal in the sparse setting is to design an algorithm whose performance is a function of s but not d. Works that study the regret minimization problem for sparse linear bandits include [22–24].

In this paper, we study the BAI problem in sparse linear bandits under the fixed-budget setting. To the best of our knowledge, this paper presents the first result on the BAI problem in linear bandits with sparse structure.

Contributions Our main contributions are summarized as follows.

(i) We design an algorithm, *Lasso and Optimal-Design- (Lasso-OD) based Linear Best Arm Identification*. This algorithm has two phases. In the first phase, we pull arms to estimate a support set $\hat{\mathcal{S}}$ that captures the support of the unknown parameter θ^* with high probability and has size as small as possible. This goal is accomplished by the thresholded Lasso (TL) introduced by Zhou [25]. TL obtains an initial estimation $\hat{\theta}_{\text{init}}$ for the parameter θ^* from the Lasso [26] and passes it through an absolute value threshold to obtain $\hat{\theta}_{\text{thres}}$. The support of $\hat{\theta}_{\text{thres}}$ is the output of the first phase. In the second phase, we apply OD-LinBAI from [20]. Lasso-OD employs 3 hyperparameters: (i) $T_1 < T$, the budget allocated for the first phase; (ii) $\lambda_{\text{init}} > 0$, the parameter in the initial Lasso problem; and (iii) $\lambda_{\text{thres}} > 0$, the threshold value in TL. The choice of the design matrix (i.e., number of times each arm is pulled) in the first phase, which determines the number of arm pulls for each arm, is crucial in attaining a good performance. Similar to [24], we choose the design matrix as the maximizer of the minimum eigenvalue of the Gram matrix associated with the design matrix. This particular choice minimizes an upper bound on a probability term related to the performance of TL.

(ii) We derive a non-asymptotic upper bound on the error probability of Lasso-OD as a function of the total budget T, the number of arms K, the ambient dimension d, the sparsity s, and the arm vectors $a(k)$, $k = 1, \ldots, K$, the first few suboptimality gaps, and the hyperparameters $T_1, \lambda_{\text{init}}$, and λ_{thres}. As a corollary to this bound, with the knowledge of s, we carefully choose the hyperparameters so that (1) with high probability, phase 1 selects all variables in θ^* and at most s^2 additional variables; (2) the probability terms due to phases 1 and 2 are balanced. This particular choice achieves the error probability $\exp\left\{-\Omega\left(\frac{T}{\log_2(s) H_{2,\text{lin}}(s+s^2)}\right)\right\}$ for fixed s, $T \to \infty$, K and d not growing exponentially with T (see Corollary 1). Here, $H_{2,\text{lin}}(s+s^2)$ is a hardness parameter that depends only on the first $s+s^2-1$ suboptimality gaps. Note that the exponent is independent of dimension d, implying that increase in d does not significantly increase the error probability. For OD-LinBAI, this exponent is given by

$\exp\{-\Omega(\frac{T}{\log_2(d)H_{2,\mathrm{lin}}(d)})\}$; therefore, Lasso-OD improves the error probability exponent by a factor of $\Omega(\frac{\log_2 d}{\log_2 s})$ for $d \geq s + s^2$.

(iii) We empirically compare the identification error of Lasso-OD with that of other existing algorithms in the literature on a synthetic dataset. The empirical results support our theoretical result that claims that the scaling of the error probability of Lasso-OD is characterized by the sparsity s while the performances of other algorithms significantly depend on d.

2 Problem Formulation

We consider a standard linear bandit with K arms with a d-dimensional unknown global parameter θ^*. Let the arm set be $[K] \triangleq \{1, \ldots, K\}$, where each arm $k \in [K]$ is associated with a known arm vector $a(k) \in \mathbb{R}^d$. A set of K arms, $\{a(1), \ldots, a(K)\}$ together with θ^* define a linear bandit instance η. At each time t, the agent chooses an arm $A_t \in [K]$ and observes a noisy reward

$$Y_t = \langle \theta^*, a(A_t) \rangle + \epsilon_t, \tag{1}$$

where $\epsilon_1, \epsilon_2, \ldots$ are independent 1-subgaussian noise variables. For the arm selection, the agent uses an online algorithm, that is, the arm pull $A_t \in [K]$ may depend only on the previous $t-1$ arm pulls $A_1, \ldots, A_{t-1}$ and their corresponding rewards $Y_1, \ldots, Y_{t-1}$. Denote the mean rewards of the arm vectors by

$$\mu_k \triangleq \langle \theta^*, a(k) \rangle, \quad \forall\, k \in [K]. \tag{2}$$

Without loss of generality, we assume that $\mu_1 > \mu_2 \geq \mu_3 \geq \cdots \geq \mu_K$, i.e., arm 1 is the unique best arm. We denote the mean gaps by $\Delta_k \triangleq \mu_1 - \mu_k$ for $2 \leq k \leq K$.

Under the fixed-budget setting of BAI, the agent is given a fixed time T, and makes an estimate $\hat{I}$ for the best arm with no more than T arm pulls. The goal is to design an online algorithm with the identification error probability, $\mathbb{P}[\hat{I} \neq 1]$, as small as possible.

Notation For any integer n, we denote $[n] \triangleq \{1, \ldots, n\}$. Let $x = (x_1, \ldots, x_d)$ be a d-dimensional vector and $\mathcal{S} \subseteq [d]$, we denote $x_S \triangleq (x_s \colon s \in \mathcal{S}) \in \mathbb{R}^{|\mathcal{S}|}$. We denote the matrices by sans-serif font, e.g., $\mathsf{A} \in \mathbb{R}^{n \times d}$. The j-th column of A is denoted by $\mathsf{A}_j \in \mathbb{R}^n$. We denote $\|x\|_{\mathsf{A}} \triangleq \sqrt{x^\top \mathsf{A} x}$. The minimum eigenvalue of a symmetric A is denoted by $\sigma_{\min}(\mathsf{A})$. We denote the set of distributions on the set $\mathcal{A}$ as $\mathcal{P}(\mathcal{A})$. Let $A_1, A_2, \ldots, A_t$ be the sequence of arm pulls. The matrix $\mathsf{X} \in \mathbb{R}^{t \times d}$ whose j-th row is $a(A_j)$ is called the *design matrix*. Let $\nu \in \mathcal{P}([K])$ be the fractions of arm pulls associated with this strategy, i.e., $\nu_k = \frac{1}{t}\sum_{j=1}^{t} 1\{A_j = k\}$ for $k \in [K]$. The *Gram matrix* associated with this strategy is denoted by $\mathsf{M}(\nu) = \frac{1}{t}\mathsf{X}^\top \mathsf{X} = \sum_{k \in [K]} \nu_k a(k) a(k)^\top \in \mathbb{R}^{d \times d}$.

Model assumptions Denote the support of θ^* by $S(\theta^*) \triangleq \{j \in [d] \colon (\theta^*)_j \neq 0\}$. We assume that the unknown parameter θ^* and the arm vectors $\{a(k)\}_{k \in [K]}$ are potentially high-dimensional, i.e., $d \gg 1$, but θ^* is sparse, i.e., the number of non-zero coefficients in θ satisfies $\|\theta^*\|_0 \triangleq |S(\theta^*)| = s \ll d$. We assume that $S(\theta^*)$ is unknown, but s and $\theta_{\min} \triangleq \min_{j \in S(\theta^*)} |(\theta^*)_j|$ are known. We further assume that $|\mu_k| \leq 1$ for all arms $k \in [K]$.

3 Our Algorithm: Lasso-OD

We now present our algorithm, *Lasso and Optimal-Design- (Lasso-OD) based linear best-arm identification*. Lasso-OD has two phases. In phase 1, we pull a judiciously chosen set of arms to learn the support of the unknown parameter θ^*. Specifically, we want phase 1 to output a subset of variables $\hat{\mathcal{S}} \subseteq [d]$ so that the estimated support $\hat{\mathcal{S}}$ captures the true variables, $S(\theta^*)$, with high probability, and its cardinality $|\hat{\mathcal{S}}|$ is small enough. To do this, we use the thresholded Lasso introduced by Zhou [25]. Once $\hat{\mathcal{S}}$ is obtained, we eliminate all variables in the arm vectors except the ones in $\hat{\mathcal{S}}$. Note that given that $\hat{\mathcal{S}} \supseteq S(\theta^*)$, this variable elimination would have no effect on the mean values $\mu_1, \ldots, \mu_K$ since by assumption, we only eliminate some variables $j \in [d]$ with $\theta^*_j = 0$. Therefore, the best arm is also preserved after the variable elimination. Building upon this principle, in phase 2, we project the arms on the estimated support $\hat{\mathcal{S}}$ and pull arms according to the OD-LinBAI algorithm by Yang and Tan [20], which is designed for linear bandits with no sparsity condition.

3.1 Motivation for Lasso-OD Algorithm

OD-LinBAI used in phase 2 is a minimax optimal algorithm up to a multiplicative factor in the exponent in the sense that it achieves an asymptotic error probability $\exp\big\{-\Omega\big(\frac{T}{\log_2(d)H_{2,\mathrm{lin}}(d)}\big)\big\}$, and for every algorithm, there exists a bandit instance η whose asymptotic error probability is lower bounded by $\exp\big\{-O\big(\frac{T}{\log_2(d)H_{2,\mathrm{lin}}(d)}\big)\big\}$. The hardness parameter

$$H_{2,\mathrm{lin}}(d) \triangleq \max_{2\le i\le d} \frac{i}{\Delta_i^2} \tag{3}$$

determines how difficult it is to identify the best arm for a given bandit instance η [20]. For sparse linear bandits, if an oracle knew the support of the unknown parameter θ^*, then the lower bound in [20, Th. 3] would be improved to $\exp\big\{-O\big(\frac{T}{\log_2(s)H_{2,\mathrm{lin}}(s)}\big)\big\}$. The purpose of TL in phase 1 is to provide an estimate for the support of θ^* with high accuracy while also pulling arms few enough that the resulting error probability is a function of s rather than d as in the oracle lower bound. Below, we provide the details on two phases of Lasso-OD.

3.2 Phase 1 (TL)

Consider a linear model $Y = \mathsf{X}\theta^* + \epsilon$, where $\mathsf{X} \in \mathbb{R}^{T_1\times d}$ is a fixed design matrix, $\theta^* \in \mathbb{R}^d$ is a fixed unknown feature vector, $Y \in \mathbb{R}^{T_1}$ is the response vector, and $\epsilon \in \mathbb{R}^{T_1}$ is a noise vector whose entries are independent and 1-subgaussian. Tibshirani [26] introduces the Lasso optimization problem to identify a sparse solution to the least squares estimation problem

$$\hat{\theta}_{\mathrm{init}} = \operatorname*{argmin}_{\theta\in\mathbb{R}^d} \frac{1}{T_1}\|Y - \mathsf{X}\theta\|_2^2 + \lambda_{\mathrm{init}}\|\theta\|_1, \tag{4}$$

where $\lambda_{\mathrm{init}} > 0$ is a suitably chosen regularization parameter. The Lasso (4) is a convex program and can be efficiently solved, e.g., using Alternating Direction Method of Multipliers (ADMM) algorithm [27].

For the task of variable selection, i.e., recovering the support of the unknown parameter θ^* without missing any of its non-zero variables, we want to get an estimate $\hat{\theta}$ that satisfies $S(\hat{\theta}) \supseteq S(\theta^*)$ while ensuring that $|S(\hat{\theta}) \setminus S(\theta^*)|$ is as small as possible. Zhou [25] introduces the following thresholding procedure that has this property

$$(\hat{\theta}_{\mathrm{thres}})_j = (\hat{\theta}_{\mathrm{init}})_j \, 1\{|(\hat{\theta}_{\mathrm{init}})_j| \ge \lambda_{\mathrm{thres}}\}, \quad \forall j \in [d], \tag{5}$$

where the initial estimate $\hat{\theta}_{\mathrm{init}}$ is given in (4), and $\lambda_{\mathrm{thres}} > 0$ is the threshold value. The set of selected variables by TL is $S(\hat{\theta}_{\mathrm{thres}})$. A variation of TL is used by Ariu *et al.* [23] to derive refined regret guarantees in sparse stochastic contextual linear bandits. Their main idea is to find the support estimate $S(\hat{\theta}^{(t)}_{\mathrm{thres}})$ at each time instance t using TL and then to compute the ordinary least squares (OLS) estimation restricted on the variables in $S(\hat{\theta}^{(t)}_{\mathrm{thres}})$. Ariu *et al.* [23] tune the free parameters $\lambda^{(t)}_{\mathrm{init}}$ and $\lambda^{(t)}_{\mathrm{thres}}$ in a way that with high probability, $S(\hat{\theta}^{(t)}_{\mathrm{thres}}) \supseteq S(\theta^*)$ and $S(\hat{\theta}^{(t)}_{\mathrm{thres}})$ is small enough, which is $s + O(\sqrt{s})$ in their case. Note that on the event $\{S(\hat{\theta}^{(t)}_{\mathrm{thres}}) \supseteq S(\theta^*)\}$, the OLS solution restricted on the subset $S(\hat{\theta}^{(t)}_{\mathrm{thres}})$ is equal to that for the unrestricted case where all d variables are used. Our approach is similar to that in [23] in using TL to reduce the effective dimension of the problem.

Let $T_1 < T$ be the budget allocated to the variable selection procedure described above.

Design matrix optimization First, we need to specify the number of pulls for each arm during phase 1, which corresponds to determining the design matrix $\mathsf{X} \in \mathbb{R}^{T_1\times d}$ in the Lasso problem (4). To do this, we solve the optimization problem given by

$$\tilde{\nu}^\star = \operatorname*{argmax}_{\nu\in\mathcal{P}([K])} \sigma_{\min}\left(\sum_{i=1}^{K} \nu_i a(i)a(i)^\top\right). \tag{6}$$

Since the function $\mathsf{A} \mapsto \sigma_{\min}(\mathsf{A})$ is concave and $\nu \mapsto \sum_{i=1}^{K} \nu_i a(i) a(i)^\top$ is linear, (6) is a convex optimization problem, and can be solved, for example, using the CVX toolbox [28].

The design matrix determined by the allocation in (6) minimizes an upper bound on a probability term related to phase 1; hence, it approximately optimizes the penalty term due to incorrectly estimating the variables of θ^*. More discussion on this choice of the design matrix appears in Appendix A. The optimization problem (6) also appears in [24] on their regret analysis in sparse linear bandits. The allocation $\tilde{\nu}^\star$ can lead to fractional number of pulls $T_1 \tilde{\nu}_i^\star$ for some arm $i \in [K]$. To guarantee integer number of pulls for all arms, we apply a rounding procedure given in [29, Ch. 12], the ROUND function in Appendix B, which is also employed in the fixed-confidence BAI algorithm in [13].

Support estimation We compute the number of pulls for each arm using (6) and ROUND, and then estimate the support from (4)–(5). Algorithm 1 below gives the pseudo-code of this procedure.

Algorithm 1 Support Estimation using Thresholded Lasso (SETLA)

Input: Time budget T_1, Lasso parameters λ_{init} and λ_{thres}, and arms $a(1), \dots, a(K)$.
Compute the arm pull fractions $\tilde{\nu}^*$ from (6).
Update $\tilde{\nu}^* \leftarrow \text{ROUND}(\tilde{\nu}^*, T_1)$ to ensure integer number of arm pulls.
Pull each arm $i \in [K]$ exactly $T_1 \tilde{\nu}_i^*$ times. Denote the vector of rewards by $Y \in \mathbb{R}^{T_1}$.
Form the design matrix $\mathsf{X} \in \mathbb{R}^{T_1 \times d}$ so that it has $T_1 \tilde{\nu}_i^*$ rows equal to $a(i)^\top$ for $i \in [K]$. Compute $\hat{\theta}_{\text{thres}}$ from (4)–(5).
Output: the support $\hat{\mathcal{S}} = S(\hat{\theta}_{\text{thres}})$.

3.3 Phase 2 (OD-LinBAI)

In this section, we review the OD-LinBAI algorithm by Yang and Tan [20]. OD-LinBAI divides the budget T into $\lceil \log_2 d \rceil$ phases, where each phase has roughly the same length.

At the beginning of round r, OD-LinBAI applies a dimensionality reduction step to maintain that the set of modified arms spans the space of its reduced dimension. The arm allocation during each round is determined by the G-optimal design [30], which takes an arm set $\{a(1), \dots, a(K)\} \subseteq \mathbb{R}^d$ and solves the optimization problem

$$\pi^* = \operatorname*{argmin}_{\pi \in \mathcal{P}([K])} \max_{i \in [K]} \|a(i)\|^2_{\mathsf{M}(\pi)^{-1}}, \tag{7}$$

where $\mathsf{M}(\pi) \triangleq \sum_{i=1}^{K} \pi_i a(i) a(i)^\top$ is the Gram matrix associated with the allocation π. At the beginning of each round, we solve (7) for the set of active arms and then apply the ROUND function in Appendix B to the resulting allocation to ensure integer number of pulls. The latter step replaces the procedure in Line 17 of Algorithm 1 in [20]. This slight modification may improve the performance of the algorithm especially if the budget T is small. At the end of round 1, we eliminate all arms except the top $\lceil \frac{d}{2} \rceil$ with respect to the OLS estimator; in the rest, we halve the remaining arms at the end each round. At the end of last round, only one arm remains and that arm is declared to be the best one. The pseudo-code of OD-LinBAI can be found in [20] and a slight modification of it which leads to the improved error probability bound in Theorem 2 can be found in Appendix B.

3.4 Lasso-OD Algorithm

The pseudo-code of Lasso-OD described above is given in Algorithm 2. Notice that since the two phases of Lasso-OD operate independently, if needed, one can replace either or both of TL and OD-LinBAI with their alternatives, e.g., the adaptive Lasso [31, Ch. 2.8] for TL and any of the algorithms in [17–19, 21] for OD-LinBAI.

4 Main Results

This section presents three non-asymptotic bounds on the performances of TL, OD-LinBAI, and Lasso-OD algorithms.

Algorithm 2 Lasso and Optimal-Design Based Linear Best Arm Identification (Lasso-OD)

Input: Time budgets T_1 and T_2 so that $T = T_1 + T_2$, Lasso parameters λ_{init} and λ_{thres}, and arm vectors $a(1), \ldots, a(K) \in \mathbb{R}^d$.

Run Algorithm 1 with T_1, λ_{init}, and λ_{thres} and get the output $\hat{\mathcal{S}} \subseteq [d]$.

Project the arm vectors on the subset $\hat{\mathcal{S}}$ by setting $a'(i) = (a(i))_{\hat{\mathcal{S}}}$ for $i \in [K]$.

Run OD-LinBAI from [20] with budget T_2 and arm vectors $\{a'(1), \ldots, a'(K)\} \subseteq \mathbb{R}^{|\hat{\mathcal{S}}|}$ with Line 17 of Algorithm 1 in [20] replaced by ROUND.

Output: the only remaining arm $\hat{I}$ as the output of OD-LinBAI.

4.1 Thresholded Lasso

Recall the linear model $Y = \mathsf{X}\theta^* + \epsilon$, where $\mathsf{X} \in \mathbb{R}^{T_1 \times d}$ is a fixed design matrix, $\theta^* \in \mathbb{R}^d$ is a fixed unknown feature vector, $Y \in \mathbb{R}^{T_1}$ is the response vector, and $\epsilon \in \mathbb{R}^{T_1}$ is a noise vector whose entries are independent and 1-subgaussian. For any set $\mathcal{S} \subseteq [d]$, define the set of vectors

$$\mathbb{C}(\mathcal{S}) \triangleq \{\theta \in \mathbb{R}^d \colon \ \|\theta_{\mathcal{S}^c}\|_1 \leq 3\,\|\theta_{\mathcal{S}}\|_1\}. \tag{8}$$

Van de Geer and Bühlmann [32] introduce the compatibility condition that allows one to control the ℓ_1-norm error for the sparse estimation of the unknown parameter θ^* where the components of the design matrix X are not highly correlated. For the rest of the section, let $\mathsf{M} = \frac{1}{T_1}\mathsf{X}^\top\mathsf{X}$ denote the Gram matrix associated with X.

Definition 1 (Compatibility condition)**.** *Given a fixed design matrix $\mathsf{X} \in \mathbb{R}^{T_1 \times d}$ (whose Gram matrix is M) and a subset $\mathcal{S} \subseteq [d]$, the compatibility constant $\phi^2(\mathsf{M}, \mathcal{S})$ is defined as*

$$\phi^2(\mathsf{M}, \mathcal{S}) \triangleq \min_{\theta \in \mathbb{R}^d \colon \|\theta_{\mathcal{S}}\|_1 \neq 0} \left\{ \frac{|\mathcal{S}|\,\|\theta\|_{\mathsf{M}}^2}{\|\theta_{\mathcal{S}}\|_1^2} : \theta \in \mathbb{C}(\mathcal{S}) \right\}. \tag{9}$$

With some abuse of notation, we also define

$$\phi^2(\mathsf{M}, s) \triangleq \min_{\mathcal{S} \subseteq [d] \colon |\mathcal{S}| = s} \phi^2(\mathsf{M}, \mathcal{S}). \tag{10}$$

The following result controls the ℓ_1-norm error of the initial Lasso estimator in (4).

Lemma 1 ([23, Lemma G.6])**.** *Assume that $\phi^2(\mathsf{M}, s) > 0$. The Lasso estimator $\hat{\theta}_{\text{init}}$ in (4) satisfies*

$$\mathbb{P}\left[\left\|\hat{\theta}_{\text{init}} - \theta^*\right\|_1 \leq \frac{4\lambda_{\text{init}} s}{\phi^2(\mathsf{M}, s)}\right] \geq 1 - 2d\exp\left\{-\frac{T_1 \lambda_{\text{init}}^2}{32\left(\frac{1}{T_1}\max_{j \in [d]} \|\mathsf{X}_j\|_2^2\right)}\right\}. \tag{11}$$

Using Lemma 1, we derive the following bound on the event that the size of the support of the TL output (5) is below a threshold and it captures the true support $S(\theta^*)$.

Theorem 1. *Fix a design matrix $\mathsf{X} \in \mathbb{R}^{T_1 \times d}$ and parameters $\lambda_{\text{init}}, \lambda_{\text{thres}} > 0$. Let $b = \frac{4}{\phi^2(\mathsf{M}, s)}$ and $c = \frac{\lambda_{\text{thres}}}{\lambda_{\text{init}}}$. Suppose that $\theta_{\min} \geq \lambda_{\text{init}}\,(c + bs)$ holds. Then,*

$$\begin{aligned}\mathbb{P}&\left[\left\{|S(\hat{\theta}_{\text{thres}})| \leq s\left(1 + \frac{b}{c}\right)\right\} \bigcap \{S(\hat{\theta}_{\text{thres}}) \supseteq S(\theta^*)\}\right] \\ &\geq 1 - 2d\exp\left\{-\frac{T_1 \lambda_{\text{init}}^2}{32\left(\frac{1}{T_1}\max_{j \in [d]} \|\mathsf{X}_j\|_2^2\right)}\right\}.\end{aligned} \tag{12}$$

Proofs of Lemma 1 and Theorem 1 are deferred to Appendix C. Theorem 1 follows steps similar to those in [23, Lemma 5.4]. The interested reader can refer to [31, Ch. 6–7] for more results and discussions on the Lasso, TL, and their variants.

4.2 An Improved Upper Bound on the Error Probability of OD-LinBAI

The theorem below gives an improved upper bound on the error probability of OD-LinBAI [20].

Theorem 2. *Let* $\tilde{T} = \left\lfloor \frac{T}{\lceil \log_2 d \rceil} \right\rfloor$. *For any linear bandit instance* η, *the output of OD-LinBAI satisfies*

$$\mathbb{P}\left[\hat{I} \neq 1\right] \leq (K + \log_2 d) \exp\left\{ -\frac{\tilde{T}}{16\left(1 + \frac{d^2}{\tilde{T}}\right) H_{2,\mathrm{lin}}(d)} \right\}. \tag{13}$$

The right-hand side of (13) is slightly different than the one presented in [20, Th. 2]. First, in [20, Th. 2], the numerator in the exponent is equal to some constant m that is approximately equal to $\frac{T}{\log_2 d}$ just like $\tilde{T}$; this is due to the modification in the distribution rounding technique. Second, the pre-factor in [20, Th. 2] is $\frac{4K}{d} + 3\log_2 d$ instead of $K + \log_2 d$. More importantly, in (13), the coefficient 32 in the denominator of the exponent in [20, Th. 2] is improved to 16. The last two differences are due to a refinement in the proof technique. Lastly, our result includes a rounding error factor $1 + \frac{d^2}{\tilde{T}}$, which becomes negligible for a large enough T. This factor appears due to the fact that the G-optimal design may yield fractional number of pulls for some arms, which is obviously not allowed. The proof of Theorem 2 is deferred to Appendix D.

4.3 Upper Bound on the Error Probability of Lasso-OD

The theorem below bounds the probability of incorrectly identifying the best arm using Lasso-OD.

Theorem 3. *Let* $T_1 < T$ *be the length of phase 1, and let* $T_2 = T - T_1$ *be the length of phase 2. Let* λ_{init} *and* λ_{thres} *be some positive scalars. Let* $c = \frac{\lambda_{\mathrm{thres}}}{\lambda_{\mathrm{init}}}$. *Let* $\tilde{\nu}^*$ *be the solution to* (6), *and let* $\tilde{\nu} = \mathrm{ROUND}(\tilde{\nu}^*, T_1)$ *be its rounded version for length* T_1. *Suppose that* $b = \phi^2(\tilde{\nu}, s) > 0$ *and* $\theta_{\min} \geq \lambda_{\mathrm{init}}(c + bs)$. *For any linear bandit instance* η, *the output of Algorithm 2 satisfies*

$$\mathbb{P}\left[\hat{I} \neq 1\right] \leq (K + \log_2 d) \exp\left\{ -\frac{\left\lfloor \frac{T_2}{\log_2(s_1)} \right\rfloor}{16\,(1+\epsilon)\, H_{2,\mathrm{lin}}(s_1)} \right\} + 2d \exp\left\{ -\frac{T_1 \lambda_{\mathrm{init}}^2}{32 x_{\max}^2} \right\}, \tag{14}$$

where

$$s_1 = \left\lfloor s\left(1 + \frac{b}{c}\right) \right\rfloor, \quad x_{\max}^2 = \max_{j \in [d]} \sum_{k=1}^{K} \tilde{\nu}_k (a(k)_j)^2, \quad \epsilon = \frac{s_1^2}{T_2}. \tag{15}$$

Proof. The proof uses Theorems 1 and 2 for the probability terms due TL and OD-LinBAI, respectively. Let $\hat{\mathcal{S}} \subseteq [d]$ denote the output of phase 1. Define the events $\mathcal{E} \triangleq \{|\hat{\mathcal{S}}| \leq s_1\}$ and $\mathcal{F} \triangleq \{\hat{\mathcal{S}} \supseteq S(\theta^*)\}$. By the law of total probability, we have

$$\mathbb{P}\left[\hat{I} \neq 1\right] \leq \mathbb{P}\left[\hat{I} \neq 1 \middle| \mathcal{E} \cap \mathcal{F}\right] + \mathbb{P}\left[\mathcal{E}^{\mathrm{c}} \cup \mathcal{F}^{\mathrm{c}}\right]. \tag{16}$$

Given $\mathcal{E} \cap \mathcal{F}$, the error probability is bounded by the right-hand side of (13) with the budget T replaced by the length of phase 2, T_2, and with the dimension d replaced by s_1. This follows since on the event $\mathcal{F}$, the mean rewards are preserved after the arm vectors and θ^* are projected on $\hat{\mathcal{S}}$ and since the right-hand side of (13) is non-decreasing in d. From Theorem 1 and the arm-pulling strategy described in Line 2 of Algorithm 2, we have

$$\mathbb{P}\left[\mathcal{E}^{\mathrm{c}} \cup \mathcal{F}^{\mathrm{c}}\right] \leq 2d \exp\left\{ -\frac{T_1 \lambda_{\mathrm{init}}^2}{32 x_{\max}^2} \right\}. \tag{17}$$

Combining (16) with (13) and (17), we complete the proof. □

The following corollary is obtained by choosing the free parameters T_1, λ_{init}, and λ_{thres} suitably to meet the conditions of Theorem 3. These nontrivial choices use the knowledge of $\theta_{\min}$ and s but not the hardness parameter and achieve an exponent of the error probability that depends only on s, T, and the hardness parameter.

Corollary 1. *For any linear bandit instance η, it holds that*

$$\mathbb{P}\left[\hat{I} \neq 1\right] \leq (K + \log_2 d + 2d) \exp\left\{-\frac{T}{16\lfloor\log_2(s+s^2)\rfloor(1+\epsilon)H_{2,\mathrm{lin}}(s+s^2)(1+c_0)}\right\}, \quad (18)$$

where

$$c_0 = \frac{25b^2 x_{\mathrm{max}}^2}{3\theta_{\mathrm{min}}^2 \log_2(s+s^2)} \quad \textit{and} \quad \epsilon = \frac{(1+c_0)(s+s^2)^2}{T}. \quad (19)$$

Here, $c_0 = \frac{T_1}{T_2}$ is the fraction of lengths of two phases of Lasso-OD, and $(1+\epsilon)$ is the penalty due to rounding. Since c_0 in (19) is lower bounded by a positive constant for all $s \in \mathbb{N}$, Corollary 1 implies that the error probability of Lasso-OD is upper bounded by

$$\exp\left\{-\Omega\left(\frac{T}{\log_2(s)H_{2,\mathrm{lin}}(s+s^2)}\right)\right\} \quad (20)$$

for fixed s, $T \to \infty$, and K and d not growing exponentially with T. Therefore, unlike the non-sparse case in [20], the error probability exponent is *independent of the dimension* d, but instead, depends on the sparsity s, which yields much smaller error probabilities for high dimensional sparse linear bandits. The parameter choices that achieve the exponent in (20) is nontrivial; we carefully choose λ_{init} and λ_{thres} so that c_0 is decreasing in s and choose T_1 so that two exponents in (14) emanating from phases 1 and 2 are "balanced". The proof of Corollary 1 is presented in Appendix E.

Assume that the agent knows the support of θ^*. Then, following the construction in the proof of [20, Th. 3], for any algorithm, there exists a bandit instance whose error probability is lower bounded by $\exp\big\{-O\big(\frac{T}{\log_2(s)H_{2,\mathrm{lin}}(s)}\big)\big\}$. This implies that the upper bound (20) is indeed almost minimax optimal in the exponent.

5 Experiments

In this section, we numerically evaluate the performance of Lasso-OD on several synthetic sparse linear bandit instances and compare it with those of OD-LinBAI [20], BayesGap [17], GSE [21], Peace [18], and LinearExploration [19]. In each setting, we report the empirical error probabilities for Lasso-OD, BayesGap, and GSE over 4000 independent trials and for Peace and LinearExploration over 100 independent trials.

5.1 Syntethic Sparse Dataset

In the first example, we draw K arms independently from the uniform distribution on the d-dimensional sphere of radius $\sqrt{d/s}$, i.e., $\{x \in \mathbb{R}^d \colon \|x\|_2^2 = \frac{d}{s}\}$, and the sparse unknown parameter is taken as $\theta^* = (1, 1, 0, \ldots, 0)$, i.e., $s = 2$. Fig. 1 demonstrates the empirical error probabilities for $d \in \{10, 20\}$, $K \in \{50, 100\}$ and $T \in [200, 10000]$, except Peace [18] and LinearExploration [19]. Since the computational complexity of Peace and LinearExploration is much higher than the rest, we compare Lasso-OD with Peace and LinearExploration only for $T = 800$ in Table 1. Among these algorithms, Lasso-OD has the best performance for all sparse instances shown in Fig. 1 and Table 1.

Table 1: Performance comparison of several algorithms for $T = 800$, $d = 10$, $K = 50$, and $s = 2$.

	Lasso-OD-CV	Lasso-OD-An.	Peace	LinearExploration
Error probability	0.0275	0.045	0.40	0.39
Std. deviation	0.0026	0.0033	0.049	0.0153

Lasso-OD-CV sets the budgets for phase 1 and phase 2 as $T_1 = \frac{T}{5}$ and $T_2 = \frac{4T}{5}$ and tunes the Lasso parameters λ_{init} and λ_{thres} using a K-fold cross-validation procedure that uses the value of s in its loss function. See Appendix F for the details of the cross-validation procedure. As an alternative to cross-validation, Lasso-OD-Analytical uses the knowledge of s, θ_{min}, and the hardness parameter $H_{2,\mathrm{lin}}(s_1)$ in (14), and sets λ_{init}, λ_{thres}, and T_1 so that s_1 in (15) equals $s + s^2$, $\theta_{\mathrm{min}} = \lambda_{\mathrm{init}}(c + bs)$, and two exponents in (14) are equal. Note that $H_{2,\mathrm{lin}}(s_1)$ is usually not available to the agent.

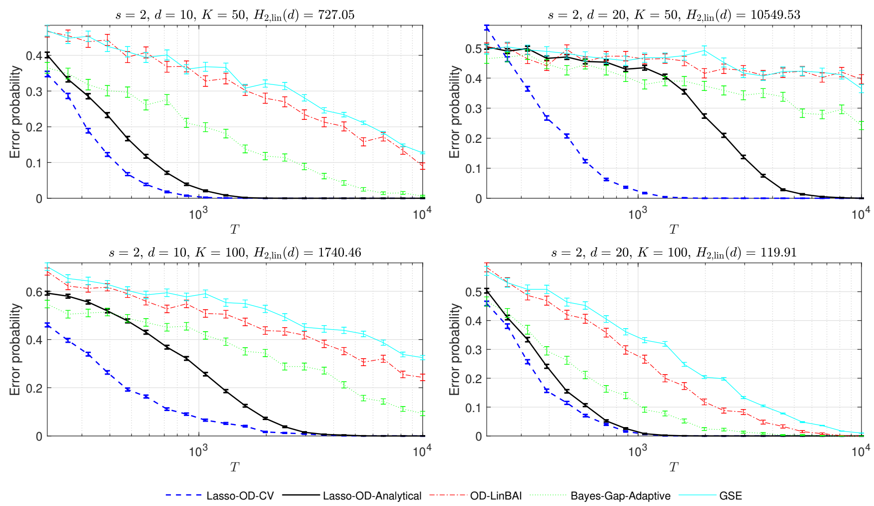

Figure 1: Comparison of several algorithms with $T \in [200, 10000]$ and $s = 2$.

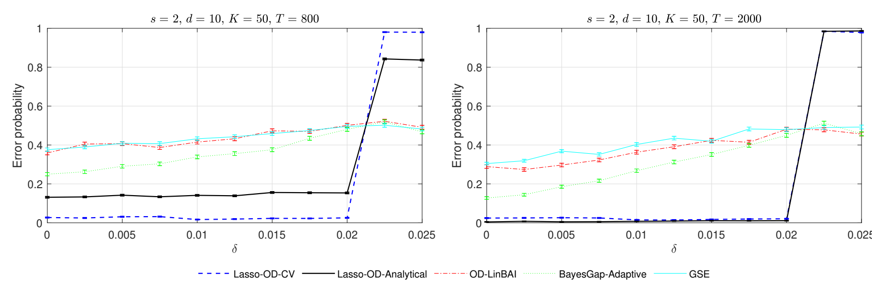

Figure 2: Comparison of several algorithms with $T \in \{800, 2000\}$, $s = 2$, and $\delta \in [0, 0.025]$.

In the second example, we test the robustness of our algorithm with respect to the variables in θ^* that are assumed to be zero by keeping the same arms as in the previous example and setting θ^* as $\theta^*_j = 1$ for $j \in [2]$, and $\theta^*_j = \delta R_j$ for $j \in \{3, \ldots, d\}$, where R_j, $j = 3, \ldots, d$, are independent Rademacher (i.e., $\{\pm 1\}$-valued) random variables, and $\delta > 0$ is a constant. Fig. 2 demonstrates the empirical error probabilities for $s = 2$, $d = 10$, $K = 50$, $T \in \{800, 2000\}$, and $\delta \in [0, 0.025]$. The phase transition for Lasso-OD in Fig. 2 suggests that Lasso-OD achieves a smaller error probability as long as δ is small enough that the approximately sparse instance (i.e., $\delta > 0$) and the sparse instance (i.e., $\delta = 0$) have the same best arm. Some examples including an instance where the hyperparameters are set as in Corollary 1 without cross-validation or knowing the hardness parameter are shown in Appendix G.

6 Conclusion

In this work, we study the BAI problem in linear bandits with sparse structure under fixed-budget setting and develop the first BAI algorithm, Lasso-OD, that exploits the sparsity of the unknown parameter θ^*. Lasso-OD combines TL for support estimation with the minimax optimal BAI algorithm, OD-LinBAI. We analyze the error probability of Lasso-OD and show that the error exponent depends on the sparsity s rather than the dimension d. Unlike other algorithms in the literature, the empirical performance of Lasso-OD does not deteriorate at large dimensions.

One future direction is to derive an instance-dependent asymptotic or non-asymptotic lower bound for the BAI problem in sparse linear bandits; however, such a bound remains open even in the non-sparse scenario. Another possible direction is to extend the TL technique used in Lasso-OD to the fixed-confidence setting. Although such an extension is relatively easy to analyze, the empirical performances of most fixed-confidence BAI algorithms in linear bandits are not heavily dependent on the dimension unlike the fixed-budget setting (see, for example, [13, 14, 16]). Therefore, the benefit of adding a TL phase in the fixed-confidence setting could be limited.

References

[1] P. Auer, “Using confidence bounds for exploitation-exploration trade-offs,” *Journal of Machine Learning Research*, vol. 3, pp. 397–422, Nov. 2002.

[2] W. R. Thompson, “On the likelihood that one unknown probability exceeds another in view of the evidence of two samples,” *Biometrika*, vol. 25, no. 3-4, pp. 285–294, 1933.

[3] H. Robbins, “Some aspects of the sequential design of experiments,” *Bull. Amer. Math. Soc.*, vol. 58, pp. 527–535, 1952.

[4] S. Bubeck and N. Cesa-Bianchi, “Regret analysis of stochastic and nonstochastic multi-armed bandit problems,” *Foundations and Trends® in Machine Learning*, vol. 5, pp. 1–122, Dec. 2012.

[5] V. Dani, T. P. Hayes, and S. M. Kakade, “Stochastic linear optimization under bandit feedback,” *21st Annual Conference on Learning Theory*, pp. 355–366, 2008.

[6] E. Even-Dar, S. Mannor, and Y. Mansour, “Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems,” *Journal of Machine Learning Research*, vol. 7, no. 39, pp. 1079–1105, 2006.

[7] Z. Karnin, T. Koren, and O. Somekh, “Almost optimal exploration in multi-armed bandits,” in *Proceedings of the 30th International Conference on Machine Learning* (S. Dasgupta and D. McAllester, eds.), vol. 28 of *Proceedings of Machine Learning Research*, (Atlanta, Georgia, USA), pp. 1238–1246, 17–19 June 2013.

[8] E. Kaufmann, O. Cappé, and A. Garivier, “On the complexity of best arm identification in multi-armed bandit models,” *Journal of Machine Learning Research*, vol. 17, pp. 1–42, 2016.

[9] A. Carpentier and A. Locatelli, “Tight (lower) bounds for the fixed budget best arm identification bandit problem,” in *29th Annual Conference on Learning Theory* (V. Feldman, A. Rakhlin, and O. Shamir, eds.), vol. 49 of *Proceedings of Machine Learning Research*, (Columbia University, New York, New York, USA), pp. 590–604, 23–26 Jun 2016.

[10] J.-Y. Audibert and S. Bubeck, “Best arm identification in multi-armed bandits,” in *COLT - 23th Conference on Learning Theory - 2010*, (Haifa, Israel), p. 13 p., June 2010.

[11] M. Soare, A. Lazaric, and R. Munos, “Best-arm identification in linear bandits,” in *Advances in Neural Information Processing Systems* (Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, eds.), vol. 27, 8–11 Dec. 2014.

[12] L. Xu, J. Honda, and M. Sugiyama, “A fully adaptive algorithm for pure exploration in linear bandits,” in *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics* (A. Storkey and F. Perez-Cruz, eds.), vol. 84 of *Proceedings of Machine Learning Research*, pp. 843–851, 09–11 Apr. 2018.

[13] T. Fiez, L. Jain, K. G. Jamieson, and L. Ratliff, “Sequential experimental design for transductive linear bandits,” in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, 10–12 Dec. 2019.

[14] C. Tao, S. Blanco, and Y. Zhou, “Best arm identification in linear bandits with linear dimension dependency,” in *Proceedings of the 35th International Conference on Machine Learning* (J. Dy and A. Krause, eds.), vol. 80 of *Proceedings of Machine Learning Research*, pp. 4877–4886, 10–15 July 2018.

[15] Y. Jedra and A. Proutiere, “Optimal best-arm identification in linear bandits,” in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 10007–10017, 7–12 Dec. 2020.

[16] M. Zaki, A. Mohan, and A. Gopalan, “Improved pure exploration in linear bandits with no-regret learning,” in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22* (L. D. Raedt, ed.), pp. 3709–3715, International Joint Conferences on Artificial Intelligence Organization, July 2022.

[17] M. Hoffman, B. Shahriari, and N. Freitas, “On correlation and budget constraints in model-based bandit optimization with application to automatic machine learning,” in *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics* (S. Kaski and J. Corander, eds.), vol. 33 of *Proceedings of Machine Learning Research*, (Reykjavik, Iceland), pp. 365–374, 22–25 Apr. 2014.

[18] J. Katz-Samuels, L. Jain, Z. Karnin, and K. G. Jamieson, "An empirical process approach to the union bound: Practical algorithms for combinatorial and linear bandits," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 10371–10382, 7–12 Dec. 2020.

[19] A. Alieva, A. Cutkosky, and A. Das, "Robust pure exploration in linear bandits with limited budget," in *Proceedings of the 38th International Conference on Machine Learning* (M. Meila and T. Zhang, eds.), vol. 139 of *Proceedings of Machine Learning Research*, pp. 187–195, 18–24 July 2021.

[20] J. Yang and V. Y. F. Tan, "Minimax optimal fixed-budget best arm identification in linear bandits," in *Advances in Neural Information Processing Systems* (A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, eds.), Nov. 2022.

[21] M. Azizi, B. Kveton, and M. Ghavamzadeh, "Fixed-budget best-arm identification in structured bandits," in *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22* (L. D. Raedt, ed.), pp. 2798–2804, International Joint Conferences on Artificial Intelligence Organization, July 2022.

[22] Y. Abbasi-Yadkori, D. Pal, and C. Szepesvari, "Online-to-confidence-set conversions and application to sparse stochastic bandits," in *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics* (N. D. Lawrence and M. Girolami, eds.), vol. 22 of *Proceedings of Machine Learning Research*, (La Palma, Canary Islands), pp. 1–9, 21–23 Apr. 2012.

[23] K. Ariu, K. Abe, and A. Proutiere, "Thresholded Lasso bandit," in *Proceedings of the 39th International Conference on Machine Learning* (K. Chaudhuri, S. Jegelka, L. Song, C. Szepesvari, G. Niu, and S. Sabato, eds.), vol. 162 of *Proceedings of Machine Learning Research*, pp. 878–928, 17–23 July 2022.

[24] B. Hao, T. Lattimore, and M. Wang, "High-dimensional sparse linear bandits," in *Advances in Neural Information Processing Systems* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), vol. 33, pp. 10753–10763, 7–12 Dec. 2020.

[25] S. Zhou, "Thresholding procedures for high dimensional variable selection and statistical estimation," in *Advances in Neural Information Processing Systems* (Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta, eds.), vol. 22, 7–10 Dec. 2009.

[26] R. Tibshirani, "Regression shrinkage and selection via the Lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[27] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends® in Machine learning*, vol. 3, no. 1, pp. 1–122, 2011.

[28] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, version 2.0." http://cvxr.com/cvx, Aug. 2012.

[29] F. Pukelsheim, *Optimal Design of Experiments*. Society for Industrial and Applied Mathematics, 2006.

[30] J. Kiefer and J. Wolfowitz, "The equivalence of two extremum problems," *Canadian Journal of Mathematics*, vol. 12, p. 363–366, 1960.

[31] P. Bühlmann and S. A. van de Geer, *Statistics for high-dimensional data*. Springer Series in Statistics, Springer, Heidelberg, 2011.

[32] S. A. van de Geer and P. Bühlmann, "On the conditions used to prove oracle results for the Lasso," *Electronic Journal of Statistics*, vol. 3, no. none, pp. 1360 – 1392, 2009.

[33] J. Löfberg, "Yalmip: A toolbox for modeling and optimization in MATLAB," in *In Proceedings of the CACSD Conference*, (Taipei, Taiwan), 2004.

A Design Matrix Optimization in Thresholded Lasso

The performance of TL in Lasso-OD is characterized by the probability term in (12). If we relax the quantity $\frac{1}{T_1}\max_{j\in[d]}\|\mathsf{X}_j\|_2^2$ in (12) by its upper bound $\max_{k\in[K]}\|a(k)\|_\infty^2$, Theorem 1 implies that the performance of TL depends on the design matrix X through b, and the best choice of X maximizes the compatibility constant $\phi^2(\mathsf{M}, s)$.

Computation of the compatibility constant Let $\nu \in \mathcal{P}_{T_1}([K])$ be the T_1-type distribution describing the fractions of the number of pulls for each arm. Then, $\frac{1}{T_1}\mathsf{X}^\top\mathsf{X} = \sum_{i\in[K]}\nu_i a(i)a(i)^\top$. Rewriting the compatibility constant $\phi^2(\mathsf{M}, \mathcal{S})$ from Definition 1, with some overload of notation, we get

$$\phi^2(\nu, \mathcal{S}) \triangleq \phi^2(\mathsf{M}, \mathcal{S}) = \min_{\theta\in\mathbb{R}^d}\left\{|\mathcal{S}|\,\|\theta\|^2_{\sum_{i\in[K]}\nu_i a(i)a(i)^\top} : \|\theta_{\mathcal{S}}\|_1 = 1, \|\theta_{\mathcal{S}^c}\|_1 \le 3\right\} \tag{21}$$

$$\phi^2(\nu, s) \triangleq \min_{\mathcal{S}\subseteq[d]:\,|\mathcal{S}|=s}\phi^2(\nu, \mathcal{S}). \tag{22}$$

Given a fixed ν, the program in (21) is non-convex due to the ℓ_1-norm equality constraint; however, by introducing binary variables, it can be turned into a mixed-integer discipled convex program (MIDCP) and be efficiently solved using CVX toolbox [28]. If we relaxed the equality constraint to $\|\theta_{\mathcal{S}}\|_1 \le 1$, then (21) would be a quadratic program (QP).

Relaxing the optimization problem According to the arguments above, the optimization problem that we originally need to solve is

$$\nu^* = \operatorname*{argmax}_{\nu\in\mathcal{P}_{T_1}([K])}\ \phi^2(\nu, s), \tag{23}$$

which is computationally intractable since the maximization constraint makes it an integer program; and even if we relaxed it to allow fractional number of pulls, the program would involve $\binom{d}{s} \approx d^s$ MIDCPs in its constraints.

Lemma 2. *For any $\nu \in \mathcal{P}([K])$ and any $\mathcal{S} \subseteq d$, it holds that $\phi^2(\nu, \mathcal{S}) \ge \sigma_{\min}(\sum_{i=1}^K \nu_i a(i)a(i)^\top)$.*

Lemma 2 follows from $|\mathcal{S}|\,\|\theta_{\mathcal{S}}\|_1^2 \le \|\theta_{\mathcal{S}}\|_2^2 \le \|\theta\|_2^2$ and relaxing the inequality constraint in (21).

Replacing $\phi^2(\nu, \mathcal{S})$ by its lower bound and allowing fractional number of pulls, we get the relaxed optimization problem in (6), which can be solved efficiently.

B Pseudo-codes of ROUND and OD-LinBAI

The pseudo-codes of the rounding procedure from [29, Ch. 12] that is used in Algorithm 1 and OD-Lasso from [20] are given below.

Algorithm 3 ROUND(π, T)

1: **Input:** a distribution π on a set with cardinality d and a positive integer T.
2: Initialize $T_i = \lceil (T - \frac{d}{2})\pi_i \rceil$ for $i = 1, \ldots, d$.
3: **while** $\sum_{i=1}^d T_i \ne T$ **do**
4:   **if** $\sum_{i=1}^d T_i < T$ **then**
5:     Set $j \leftarrow \arg\min_{i\in[d]} \frac{T_i}{\pi_i}$. Update $T_j \leftarrow T_j + 1$.
6:   **else if** $\sum_{i=1}^d T_i > T$ **then**
7:     Set $j \leftarrow \arg\max_{i\in[d]} \frac{T_i - 1}{\pi_i}$. Update $T_j \leftarrow T_j - 1$.
8:   **end if**
9: **end while**
10: **Output:** Distribution $\tilde{\pi} = \left(\frac{T_1}{T}, \ldots, \frac{T_d}{T}\right)$.

Algorithm 4 Optimal Design-Based Linear Best Arm Identification (OD-LinBAI)

Input: time budget T, arm set $\mathcal{A} = [K]$, and arm vectors $\{a(1), \ldots, a(K)\} \in \mathbb{R}^d$.
Initialize $t_0 \leftarrow 0$, $\mathcal{A}_0 \leftarrow \mathcal{A}$, $d_0 \leftarrow d$. For each $i \in \mathcal{A}_0$, set $a_0(i) = a(i)$. Set $R = \lceil \log_2 d \rceil$, $T_r = \lfloor \frac{T}{R} \rfloor$ for $r = 1, \ldots, R-1$, and $T_R = T - \sum_{i=1}^{R-1} T_i$.
for $r = 1$ **to** R **do**
  \\ Dimensionality reduction:
  Set X so that its columns are $\{a_{r-1}(i) : i \in \mathcal{A}_{r-1}\}$. Set $d_r \leftarrow \operatorname{rank}(\mathsf{X})$. Set $a_r(i) \leftarrow a_{r-1}(i)$ for $i \in \mathcal{A}_{r-1}$.
  if $d_r < d_{r-1}$ **then**
    Find the SVD of $\mathsf{X} = \mathsf{U}\Sigma\mathsf{V}^\top$, where $\mathsf{U} \in \mathbb{R}^{d_{r-1} \times d_r}$.
    Update $\mathsf{X} \leftarrow \mathsf{U}^\top \mathsf{X}$ and $a_r(i) \leftarrow \mathsf{X}_i$ for $i \in \mathcal{A}_{r-1}$.
  end if
  \\ G-optimal design:
  Input the set $\{a_r(i) : i \in \mathcal{A}_{r-1}\}$ to the G-optimal design, and set $\pi^{(r)}$ as the output of eq. (7).
  Set $\tilde{\pi}^{(r)} = \text{ROUND}(\pi^{(r)}, T_r)$ from Algorithm 2.
  \\ Arm pulling:
  Pull each arm $i \in \mathcal{A}_{r-1}$ $T_r(i) = \tilde{\pi}_i^{(r)} T_r$ times, which determines $A_{t_{r-1}+1}, \ldots, A_{t_{r-1}+T_r}$. Observe the corresponding rewards $Y_{t_{r-1}+1}, \ldots, Y_{t_{r-1}+T_r}$.
  Compute the OLS estimator:

$$\mathsf{V}^{(r)} = \sum_{i \in \mathcal{A}_{r-1}} T_r(i) a_r(i) a_r(i)^\top \tag{24}$$

$$\hat{\theta}^{(r)} = {\mathsf{V}^{(r)}}^{-1} \sum_{t=t_{r-1}+1}^{t_{r-1}+T_r} a_r(A_t) Y_t. \tag{25}$$

  \\ Arm elimination:
  Estimate the mean rewards for each $i \in \mathcal{A}_{r-1}$ as

$$\hat{\mu}_r(i) = \langle \hat{\theta}^{(r)}, a_r(i) \rangle. \tag{26}$$

  Set $\mathcal{A}_r \leftarrow$ the set of $\lceil \frac{d}{2^r} \rceil$ arms in $\mathcal{A}_{r-1}$ with the largest estimated mean rewards. Set $t_r \leftarrow t_{r-1} + T_r$.
end for
Output: $\hat{I}$ = the only remaining arm in $\mathcal{A}_R$.

C Proofs Related to Lasso

In the following, let n be the number of samples. The linear model is given by $Y = \mathsf{X}\theta^* + \epsilon$, where $Y \in \mathbb{R}^n$, $\mathsf{X} \in \mathbb{R}^{n \times d}$, and $\epsilon \in \mathbb{R}^n$, i.i.d. 1-subgaussian random variables. Recall the initial Lasso estimator

$$\hat{\theta} = \operatorname*{argmin}_{\theta \in \mathbb{R}^d} \frac{1}{n} \|Y - \mathsf{X}\theta\|_2^2 + \lambda \|\theta\|_1 . \tag{27}$$

Define the event

$$\mathcal{T} = \left\{ \max_{j \in [d]} \frac{1}{n} |\mathsf{X}_j^\top \epsilon| \le \frac{\lambda}{4} \right\}. \tag{28}$$

The following result, known as the oracle inequality, is the main tool to control the performance of the initial lasso estimator.

Lemma 3 (Oracle Inequality: Theorem 6.1 from [31])**.** *On the event $\mathcal{T}$, the initial Lasso estimator $\hat{\theta}$* (27) *satisfies*

$$\left\| \mathsf{X}(\hat{\theta} - \theta^*) \right\|_2^2 + \lambda \left\| \hat{\theta} - \theta^* \right\|_1 \le \frac{4\lambda^2 s}{\phi^2(\mathsf{M}, S(\theta^*))}. \tag{29}$$

Furthermore, it holds that

$$\mathbb{P}\left[\mathcal{T}\right] \geq 1 - 2d\exp\left\{-\frac{n\lambda^2}{32\left(\frac{1}{n}\max_{j\in[d]}\left\|\mathsf{X}_j\right\|_2^2\right)}\right\}. \tag{30}$$

Proof of Lemma 3. Since $\hat{\theta}$ minimizes (27), we have

$$\frac{1}{n}\left\|Y - \mathsf{X}\hat{\theta}\right\|_2^2 + \lambda\left\|\hat{\theta}\right\|_1 \leq \frac{1}{n}\left\|Y - \mathsf{X}\theta^*\right\|_2^2 + \lambda\left\|\theta^*\right\|_1. \tag{31}$$

Plugging $Y = \mathsf{X}\theta^* + \epsilon$ into (31), after some algebra, we get the basic inequality

$$\frac{1}{n}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2^2 + \lambda\left\|\hat{\theta}\right\|_1 \leq \frac{2}{n}\epsilon^\top\mathsf{X}(\hat{\theta} - \theta^*) + \lambda\left\|\theta^*\right\|_1. \tag{32}$$

Let $\tilde{\mathcal{T}}$ be the event

$$\tilde{\mathcal{T}} = \left\{\max_{j\in[d]}\frac{2}{n}|\epsilon^\top\mathsf{X}_j| \leq \lambda_0\right\}. \tag{33}$$

Then, on $\tilde{\mathcal{T}}$, we have using the Hölder inequality that

$$\frac{1}{n}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2^2 \leq \lambda_0\left\|\hat{\theta} - \theta^*\right\|_1 + \lambda\|\theta^*\|_1 - \lambda\left\|\hat{\theta}\right\|_1. \tag{34}$$

Let $\mathcal{S} = S(\theta^*)$. By the triangle inequality, we have

$$\left\|\hat{\theta}\right\|_1 = \left\|\hat{\theta}_{\mathcal{S}}\right\|_1 + \left\|\hat{\theta}_{\mathcal{S}^c}\right\|_1 \geq \|\theta^*_{\mathcal{S}}\|_1 - \left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 + \left\|\hat{\theta}_{\mathcal{S}^c}\right\|_1. \tag{35}$$

Applying (35) to (34), we get

$$\begin{aligned}
\frac{1}{n}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2^2 &\leq \lambda_0\left(\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 + \left\|\hat{\theta}_{\mathcal{S}^c} - \theta^*_{\mathcal{S}^c}\right\|_1\right) \\
&\quad + \lambda\left(\|\theta^*\|_1 - \|\theta^*_{\mathcal{S}}\|_1 + \left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 - \left\|\hat{\theta}_{\mathcal{S}^c}\right\|_1\right) && (36) \\
&= (\lambda_0 + \lambda)\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 + (\lambda_0 - \lambda)\left\|\hat{\theta}_{\mathcal{S}^c} - \theta^*_{\mathcal{S}^c}\right\|_1, && (37)
\end{aligned}$$

where the last step uses the fact that $\theta^*_{\mathcal{S}^c} = 0$. We set $\lambda_0 = \frac{\lambda}{2}$. Then, (37) implies that on the event $\mathcal{T}$,

$$\left\|\hat{\theta}_{\mathcal{S}^c} - \theta^*_{\mathcal{S}^c}\right\|_1 \leq 3\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1. \tag{38}$$

Therefore, $\hat{\theta} - \theta^* \in \mathbb{C}(\mathcal{S})$, and from the definition of compatibility constant in Definition 1, we have

$$\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 \leq \frac{\sqrt{s(\hat{\theta} - \theta^*)^\top\mathsf{X}^\top\mathsf{X}(\hat{\theta} - \theta^*)}}{\sqrt{n}\phi(\mathsf{M}, \mathcal{S})}. \tag{39}$$

We now continue with (37) with $\lambda_0 = \frac{\lambda}{2}$. We have

$$\begin{aligned}
\frac{2}{n}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2^2 + \lambda\left\|\hat{\theta} - \theta^*\right\|_1 &= \frac{2}{n}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2^2 + \lambda\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 + \lambda\left\|\hat{\theta}_{\mathcal{S}^c} - \theta^*_{\mathcal{S}^c}\right\|_1 && (40) \\
&\leq (3\lambda + \lambda)\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 - \lambda\left\|\hat{\theta}_{\mathcal{S}^c} - \theta^*_{\mathcal{S}^c}\right\|_1 + \lambda\left\|\hat{\theta}_{\mathcal{S}^c} - \theta^*_{\mathcal{S}^c}\right\|_1 && (41) \\
&= 4\lambda\left\|\hat{\theta}_{\mathcal{S}} - \theta^*_{\mathcal{S}}\right\|_1 && (42) \\
&\leq \frac{4\lambda\sqrt{s}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2}{\sqrt{n}\phi(\mathsf{M}, \mathcal{S})} && (43) \\
&\leq \frac{1}{n}\left\|\mathsf{X}(\hat{\theta} - \theta^*)\right\|_2^2 + \frac{4\lambda^2 s}{\phi^2(\mathsf{M}, \mathcal{S})}, && (44)
\end{aligned}$$

where (43) applies (39), and (44) applies the inequality $4uv \le u^2 + 4v^2$ to (43). Inequality (44) completes the proof of (29).

Next, we upper bound the probability $\mathbb{P}\left[\tilde{\mathcal{T}}^{\mathrm{c}}\right]$. We have

$$\mathbb{P}\left[\tilde{\mathcal{T}}^{\mathrm{c}}\right] = \mathbb{P}\left[\bigcup_{j\in[d]} \frac{1}{n}|\mathsf{X}_j^\top \epsilon| > \frac{\lambda}{4}\right] \tag{45}$$

$$\le \sum_{j=1}^{d}\left(\mathbb{P}\left[\frac{1}{n}\mathsf{X}_j^\top \epsilon > \frac{\lambda}{4}\right] + \mathbb{P}\left[-\frac{1}{n}\mathsf{X}_j^\top \epsilon > \frac{\lambda}{4}\right]\right) \tag{46}$$

$$\le 2\sum_{j=1}^{d} \exp\left\{-\frac{\lambda^2}{2\cdot 4^2 \cdot \frac{1}{n^2}\left\|\mathsf{X}_j\right\|_2^2}\right\}, \tag{47}$$

where the last inequality follows since $\frac{1}{n}\mathsf{X}_j^\top\epsilon$ $-\frac{1}{n}\mathsf{X}_j^\top\epsilon$ are subgaussian with variance proxy $\frac{1}{n^2}\left\|\mathsf{X}_j\right\|_2^2$ as $\epsilon_1, \ldots, \epsilon_n$ are independent 1-subgaussian random variables. Bounding each summand in (47) by the maximum of summands completes the proof of (30). □

Proof of Lemma 1. The right-hand side of (29) depends on the unknown set $S(\theta^*)$; however, we can further upper bound the right-hand side of (29) by replacing $\phi^2(\mathsf{M}, S(\theta^*))$ by its lower bound $\phi^2(\mathsf{M}, s)$, which is computable using only X and s. Therefore, Lemma 1 is a corollary to Lemma 3. □

Proof of Theorem 1. Define the event $\mathcal{G} \triangleq \left\{\left\|\hat{\theta}_{\mathrm{init}} - \theta^*\right\|_1 \le \frac{4\lambda_{\mathrm{init}} s}{\phi^2(\mathsf{M},s)}\right\}$. We have

$$\left\|\hat{\theta}_{\mathrm{init}} - \theta^*\right\|_1 \ge \left\|(\hat{\theta}_{\mathrm{init}} - \theta^*)_{S(\theta^*)^{\mathrm{c}}}\right\|_1 \tag{48}$$

$$= \sum_{j\in S(\theta^*)^{\mathrm{c}}} |(\hat{\theta}_{\mathrm{init}})_j| \tag{49}$$

$$\ge \sum_{j\in S(\hat{\theta}_{\mathrm{thres}})\setminus S(\theta^*)} |(\hat{\theta}_{\mathrm{init}})_j| \tag{50}$$

$$\ge |S(\hat{\theta}_{\mathrm{thres}}) \setminus S(\theta^*)|\lambda_{\mathrm{thres}}, \tag{51}$$

where (49) follows since $\theta^*_{S(\theta^*)^{\mathrm{c}}} = 0$ by assumption, and (51) follows from the thresholding step in (5). Therefore, on the event $\mathcal{G}$, it holds that

$$|S(\hat{\theta}_{\mathrm{thres}}) \setminus S(\theta^*)| \le \frac{\left\|\hat{\theta}_{\mathrm{init}} - \theta^*\right\|_1}{\lambda_{\mathrm{thres}}} \le \frac{4\lambda_{\mathrm{init}} s}{\lambda_{\mathrm{thres}}\,\phi^2(\mathsf{M}, s)}. \tag{52}$$

□

For all $j \in S(\theta^*)$, on the event $\mathcal{G}$, we have

$$|(\hat{\theta}_{\mathrm{init}})_j| \ge \theta_{\min} - \left\|(\theta^* - \hat{\theta}_{\mathrm{init}})_{S(\theta^*)}\right\|_\infty \tag{53}$$

$$\ge \theta_{\min} - \left\|(\theta^* - \hat{\theta}_{\mathrm{init}})_{S(\theta^*)}\right\|_1 \tag{54}$$

$$\ge \theta_{\min} - \left\|(\theta^* - \hat{\theta}_{\mathrm{init}})\right\|_1 \tag{55}$$

$$\ge \theta_{\min} - \frac{4\lambda_{\mathrm{init}} s}{\phi^2(\mathsf{M}, s)}. \tag{56}$$

Therefore, if

$$\lambda_{\mathrm{thres}} \ge \theta_{\min} - \frac{4\lambda_{\mathrm{init}} s}{\phi^2(\mathsf{M}, s)}, \tag{57}$$

$S(\hat{\theta}_{\mathrm{thres}}) \supseteq S(\theta^*)$ is satisfied on $\mathcal{G}$. Combining Lemma 1, (52), and (57) completes the proof of Theorem 1.

D Proof of Theorem 2

The proof of Theorem 2 closely follows the proof of [20, Th. 2]. Therefore, we only explain the differences, which are as follows.

(i) Due to our construction, m in [20] is replaced by $\tilde{T} = \left\lfloor \frac{T}{\lceil \log_2 d \rceil} \right\rfloor$.

(ii) Let $\mathcal{A}_r$ be the active arms in round r and let $\{a_r(i)\colon i \in \mathcal{A}_r\} \subset \mathbb{R}^{d_r}$ be the dimensionality-reduced arm vectors. From [13, Appendix B], it holds that

$$\max_{i \in \mathcal{A}_r} \|a_r(i)\|^2_{\mathsf{M}(\tilde{\pi}^{(r)})^{-1}} \leq d_r \left(1 + \frac{d_r^2}{\tilde{T}}\right), \tag{58}$$

where $\tilde{\pi}^{(r)}$ is the rounded version of the G-optimal design output $\pi^{(r)}$.

(iii) In the proof of [20, Lemma 3], the set $\mathcal{B}_r$ is the set of arms in $\mathcal{A}_{r-1}$ excluding the best arm and $\lceil \frac{d}{2^{r+1}} \rceil - 1$ suboptimal arms with the largest mean rewards. We re-define $\mathcal{B}_r$ as the set of arms in $\mathcal{A}_{r-1}$ excluding the best arm and $\lceil \frac{d}{2^r} \rceil - 1$ suboptimal arms with the largest mean rewards.

With the modifications in items (i) and (ii) and following the steps in the proof of [20, Lemma 2], we get for any arm $i \in \mathcal{A}_{r-1}$

$$\mathbb{P}\left[\hat{\mu}_r(1) < \hat{\mu}_r(i) | 1 \in \mathcal{A}_{r-1}\right] \leq \exp\left\{-\frac{\tilde{T}\Delta_i^2}{8\lceil \frac{d}{2^{r-1}} \rceil (1 + \frac{d^2}{\tilde{T}})}\right\}, \tag{59}$$

where $\hat{\mu}_r(i)$ denotes the estimated mean of arm i in round r.

Using item (iii) and (59), we go through the proof of [20, Lemma 3] and get

$$\mathbb{P}\left[1 \notin \mathcal{A}_r | 1 \in \mathcal{A}_{r-1}\right] \leq \begin{cases} (K - \frac{d}{2}) \exp\left\{-\frac{\tilde{T}\Delta_{\lceil \frac{d}{2^r} \rceil + 1}}{16(\lceil \frac{d}{2^r} \rceil + 1)(1 + \frac{d^2}{\tilde{T}})}\right\}, & \text{if } r = 1 \\ (\frac{d}{2^r} + 1) \exp\left\{-\frac{\tilde{T}\Delta_{\lceil \frac{d}{2^r} \rceil + 1}}{16(\lceil \frac{d}{2^r} \rceil + 1)(1 + \frac{d^2}{\tilde{T}})}\right\}, & \text{if } r > 1. \end{cases} \tag{60}$$

Finally, following the steps in the proof of [20, Th. 2] with (60), we get

$$\mathbb{P}\left[\hat{I} \neq 1\right] \leq \left(K - \frac{d}{2} + \sum_{r=2}^{\lceil \log_2 d \rceil} \frac{d}{2^r} + \lceil \log_2 d \rceil - 1\right) \exp\left\{-\frac{\tilde{T}}{16\left(1 + \frac{d^2}{\tilde{T}}\right) H_{2,\mathrm{lin}}(d)}\right\} \tag{61}$$

$$\leq (K + \log_2 d) \exp\left\{-\frac{\tilde{T}}{16\left(1 + \frac{d^2}{\tilde{T}}\right) H_{2,\mathrm{lin}}(d)}\right\}, \tag{62}$$

which completes the proof.

E Proof of Corollary 1

We set λ_{init} and λ_{thres} as

$$\kappa = \frac{b^2}{\theta_{\min}^2} \frac{25}{24} \tag{63}$$

$$\lambda_{\mathrm{init}} = \frac{1}{\sqrt{\kappa(s + s^2)}} \tag{64}$$

$$\lambda_{\mathrm{thres}} = \frac{b}{s} \lambda_{\mathrm{init}}. \tag{65}$$

Note that (65) sets $s_1 = s + s^2$ in (15), and for any $s \in \mathbb{N}$, we check that the condition in Theorem 3 holds:

$$\theta_{\min} = \frac{b}{\sqrt{\kappa}} \sqrt{\frac{25}{24}} \geq \frac{b}{\sqrt{\kappa}} \frac{s + \frac{1}{s}}{\sqrt{s + s^2}} = \lambda_{\mathrm{init}}(c + bs). \tag{66}$$

Next, we would like to set $c_0 = \frac{T_1}{T_2}$ so that the two exponents in (14) are equal. However, since $H_{2,\text{lin}}(s+s^2)$ is not available to us, we use the lower bound

$$H_{2,\text{lin}}(s+s^2) = \max_{i\in\{2,\dots,s+s^2\}} \frac{i}{\Delta_i^2} \geq \frac{s+s^2}{\Delta_{s+s^2}^2} \geq \frac{s+s^2}{4}, \tag{67}$$

where the last inequality follows from the assumption $|\mu_k| \leq 1$ for all $k \in [K]$.[1] One can further upper bound Δ_{s+s^2} by using the values of K arm vectors and searching for θ^* that gives the largest Δ_{s+s^2}.

We set the ratio $c_0 = \frac{T_1}{T_2}$ as

$$c_0 = \frac{25 b^2 x_{\max}^2}{3\theta_{\min}^2 \log_2(s+s^2)}, \tag{68}$$

which together with (63)–(67) ensures that

$$\exp\left\{-\frac{\left\lfloor \frac{T_2}{\log_2(s_1)} \right\rfloor}{16\,(1+\epsilon)\, H_{2,\text{lin}}(s_1)}\right\} \geq \exp\left\{-\frac{T_1 \lambda_{\text{init}}^2}{32 x_{\max}^2}\right\}. \tag{69}$$

Combining (69) with $s_1 = s + s^2$ completes the proof.

Remark 1. *The ratio c_0 decreasing with s as in (68) is consistent since as s approaches d, the sparse linear bandit approaches the standard linear bandit, and we would expect to spend more budget on phase 2 than phase 1 for large s. We deliberately choose the parameters in (63)–(65) to maintain this property.*

F Implementation Details

In all computations of the Lasso problem (4), we use the ADMM algorithm [27].

F.1 Lasso-OD with K-fold Cross-Validation

In the implementation of Lasso-OD-CV, the ratio of the budgets, $\frac{T_1}{T_2}$, is set to the default value $\frac{1}{4}$, i.e., naturally, the algorithm spends more budget for the BAI algorithm than for the support estimation.

For tuning the hyperparameters λ_{init} and λ_{thres}, we pull T_1 arms according to the allocation given in (6). We use the following cross-validation steps to tune the parameters.

(i) Fix two sets of hyperparameter candidates $\{\lambda_{\text{init},1}, \lambda_{\text{init},2}, \dots, \lambda_{\text{init},m}\}$ and $\{\lambda_{\text{thres},1}, \lambda_{\text{thres},2}, \dots, \lambda_{\text{thres},m}\}$.

(ii) Iteratively tune the parameters by fixing one of them and searching for the best parameter for the other one.

(iii) In each cross-validation round, the objective is to minimize the loss function

$$L = \frac{1}{T_1}\mathbb{E}\left[\left\|Y - \mathsf{X}\hat{\theta}_{\text{thres}}\right\|_2^2\right] + c_1 \mathbb{P}\left[\left\|\hat{\theta}_{\text{thres}}\right\|_0 < s\right] + c_2 \mathbb{E}\left[1\left\{\left\|\hat{\theta}_{\text{thres}}\right\|_0 > s\right\}\left\|\hat{\theta}_{\text{thres}}\right\|_0\right], \tag{70}$$

where c_1 and c_2 are ℓ_0-norm regularization parameters. Here, the first term in (70) is the mean-squared error; the second and the third terms penalize the ℓ_0-norm error and force the hyperparameters to output an estimate with s variables. In application, we set $c_1 = 200$ and $c_2 = 5$, giving more importance to detecting at least s variables. If the value of s is not available, we can still use this technique by setting $c_1 = c_2 = 0$. As standard, we approximate (70) by training the parameters in $K-1$ blocks and testing in the remaining block.

(iv) To fasten the convergence, in each round of cross-validation, we exponentially narrow down the candidate sets.

(v) To reduce the variance in cross-validation, we employ Monte-Carlo simulations, i.e., we independently divide the data into K blocks for multiple times and then take the average loss.

[1] This step is the only place where the assumption on the mean rewards is used.

F.2 Lasso-OD-Analytical

In the implementation of Lasso-OD-Analytical, we set the hyperparameters λ_{init} and λ_{thres} as in (63)–(65), and c_0 is set so that (68) holds with equality. This setup effectively uses the hardness parameter $H_{2,\text{lin}}(s+s^2)$. To compute these hyperparameters, we first need to compute $\phi^2(\mathsf{M}, s)$. As discussed in Appendix A, this requires solving a MIDCP. We do this by using the YALMIP toolbox [33] because it does not ask to convert the problem into another one. Alternatively, one can use the CVX toolbox [28] with a little bit more effort, or use Lemma 2 and solve an easier problem at the expense of some performance loss.

F.3 OD-LinBAI and Other BAI Algorithms

We implement OD-LinBAI and other BAI algorithms shown in Section 5 using the methods described in [20, Appendix E].

BayesGap-Adaptive: In general, BayesGap algorithm [17] requires the knowledge of the hardness parameter. As in [17, 20], at the beginning of each time instant, we input the estimated hardness parameter according to the three-sigma rule. In the experiments, we omit the oracle version of BayesGap that directly uses the knowledge of the hardness parameter.

G Additional Experiments

G.1 Variants of Our Algorithm

We present two variants of Lasso-OD that modify the operations in phase 2.

Lasso-$\mathcal{XY}$-Allocation: This algorithm is identical to Lasso-OD except that the G-optimal design used to determine the allocations within each round is replaced by the $\mathcal{XY}$-allocation from [11]. Let $\mathcal{X} = \{a(i) \colon i \in \mathcal{A}\}$ be the set of arms in an active set $\mathcal{A}$. Let $\mathcal{Y} = \{x - x' \colon x, x' \in \mathcal{X}, x \neq x'\}$ be the set of arm differences. The $\mathcal{XY}$-allocation solves the problem

$$\pi^* = \operatorname*{argmin}_{\pi \in \mathcal{P}(\mathcal{A})} \max_{y \in \mathcal{Y}} \|y\|_{\mathsf{M}(\pi)^{-1}} . \tag{71}$$

Lasso-$\mathcal{XY}$-allocation replaces Line 11 of Algorithm 4 with (71). In the experiments, we compute (71) using the Frank–Wolfe algorithm as in [13].

From [20, Proof of Lemma 2], the probability that a sub-optimal arm i has a smaller estimated mean than the optimal arm 1 is bounded as

$$\begin{aligned}
\mathbb{P}\left[\hat{\mu}(1) < \hat{\mu}(i)\right] &\le \exp\left\{ -\frac{\Delta_i^2}{2\,\|a(1) - a(i)\|_{\mathsf{M}(\pi)^{-1}}} \right\} &(72)\\
&\le \exp\left\{ -\frac{\Delta_i^2}{2\max_{i \neq j} \|a(j) - a(i)\|_{\mathsf{M}(\pi)^{-1}}} \right\} &(73)\\
&\le \exp\left\{ -\frac{\Delta_i^2}{8\max_{i \in \mathcal{A}} \|a(i)\|_{\mathsf{M}(\pi)^{-1}}} \right\}, &(74)
\end{aligned}$$

where π is the allocation within the round. Here, (74) follows from the triangle inequality. The G-optimal design optimizes the allocation π in (74), and $\mathcal{XY}$-allocation optimizes (73). Since $\mathcal{XY}$-allocation optimizes a tighter bound, Lasso-$\mathcal{XY}$-allocation is expected to perform better than Lasso-OD.

Lasso-BayesGap: Since BayesGap performs better than OD-LinBAI in the examples in Section 5, we propose the variant Lasso-BayesGap where in phase 2, OD-LinBAI is replaced by BayesGap-Adaptive from [17].

Both of these variants tune the Lasso parameters using cross-validation.

G.2 Experiments

In the experiments below, we include Lasso-$\mathcal{XY}$-allocation and Lasso-BayesGap to the list of algorithms in Section 5.

G.2.1 First Example

In the first example, we test the performance of the various BAI algorithms for sparsities of at least 2. We generate K d-dimensional arm vectors $a(k) = (a(k)_i : i \in [d])$, $k \in [K]$, where $a(k)_i$'s are distributed $\mathcal{N}(0, \frac{1}{s})$ independent across arms $k \in [K]$ and coordinates $i \in [d]$. The s-sparse unknown vector θ^* is set as $(\theta^*)_i = \frac{1}{\sqrt{s}}$ for $i \in [s]$ and $(\theta^*)_i = 0$ for $i = s+1, \ldots, d$. Fig. 3 compares the performances of several algorithms in the literature and variants of our algorithm for $s \in \{2, 3, 4\}$, $K = 50$, $d \in \{10, 20\}$, and $T \in [200, 10^4]$. For all bandit instances under consideration, the

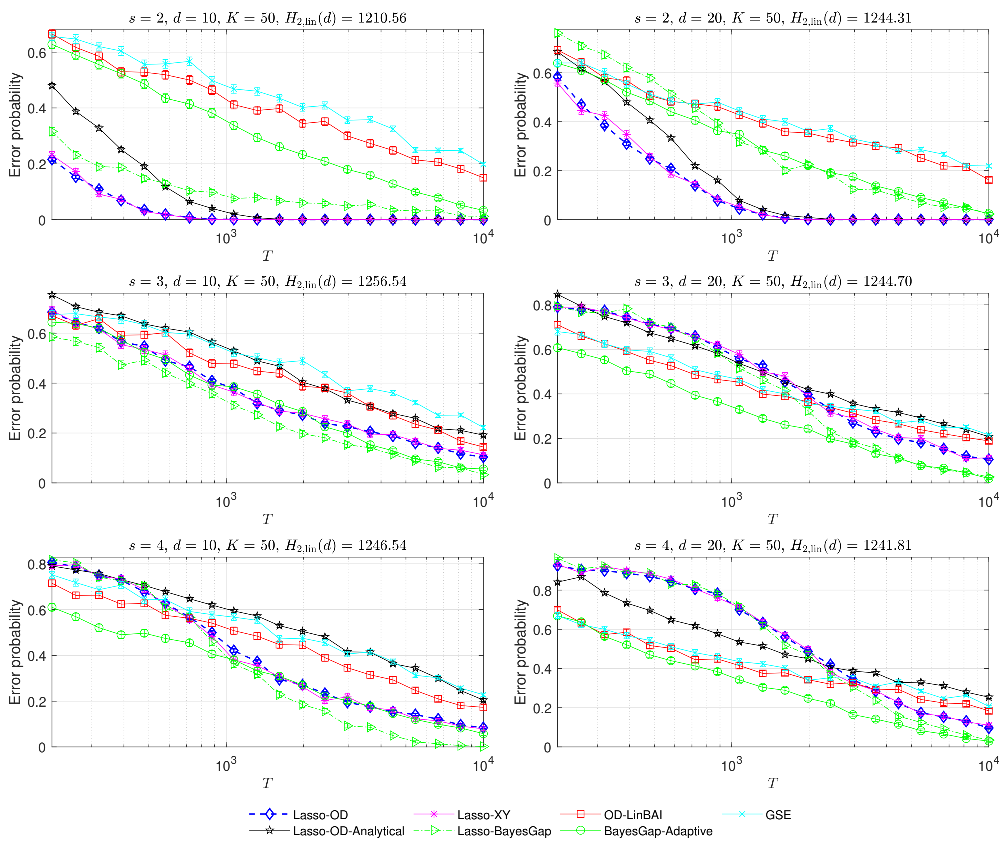

Figure 3: Comparison of several algorithms with $s \in \{2, 3, 4\}$.

performances of Lasso-OD and Lasso-$\mathcal{XY}$-allocation are almost identical. However, due to its low computational complexity (see tables below for the CPU runtimes), Lasso-OD is preferred over Lasso-$\mathcal{XY}$-allocation. Among different variants of Lasso-OD, Lasso-OD and Lasso-$\mathcal{XY}$-allocation have the best performance for the instances with $s = 2$. For $s = 3$ and $s = 4$, among the algorithms shown, Lasso-BayesGap performs the best for a large enough budget T. The poor performance of Lasso-based algorithms for small budgets is because at larger s, the minimum budget that should be allocated to phase 1 to reliably estimate the support of θ^* increases with s. For the instances with $s \in \{3, 4\}$, BayesGap-Adaptive performs remarkably well, but it is outperformed by Lasso-based algorithms for $s = 2$.

In Tables 2–3,[2] we report the average CPU runtimes for the instances in the first example with $s = 2$ and $s = 3$. Lasso-OD is superior to all other algorithms in terms of the computational complexity.[3]

G.2.2 Second Example

In the second example, we assume that θ^* belongs to a finite set $\mathcal{H} \triangleq \{\theta \in \mathbb{R}^d \colon \|\theta\|_0 = s, \theta_i \in \{-\frac{1}{\sqrt{s}}, 0, \frac{1}{\sqrt{s}}$ for $i \in [d]\}\}$. In other words, the non-zero coordinates of θ^* are assumed to have magnitude $\frac{1}{\sqrt{s}}$. We generate K d-dimensional arm vectors in the vicinity of $\mathcal{H}$ as follows: $a(k)_i = R_{k,i} \cos(\pi/4 + Z_{k,i})$, where $R_{k,i}$'s are independently and identically distributed (i.i.d.) generated with distribution $\mathrm{Unif}(\{-1, 1\})$, $Z_{k,i}$'s are i.i.d. generated with distribution $\mathcal{N}(0, 0.01)$, and $R_{k,i}$'s and $Z_{k,i}$'s are independent. For this bandit instance, given the arm vectors and using the assumption that $\theta^* \in \mathcal{H}$, we can lower bound the hardness parameter $H_{2,\mathrm{lin}}(s + s^2)$ by computing the minimum hardness parameter for the vectors $\theta \in \mathcal{H}$. In Fig. 4, Lasso-OD-An.-LB computes the hyperparameters analytically and obtains T_1 from the lower bound on $H_{2,\mathrm{lin}}(s + s^2)$ above instead of the true value of $H_{2,\mathrm{lin}}(s + s^2)$. Fig. 4 shows that Lasso-OD-An.-LB outperforms all other algorithms in the literature and achieves similar performance as Lasso-OD and Lasso-$\mathcal{XY}$-allocation for a large enough budget.

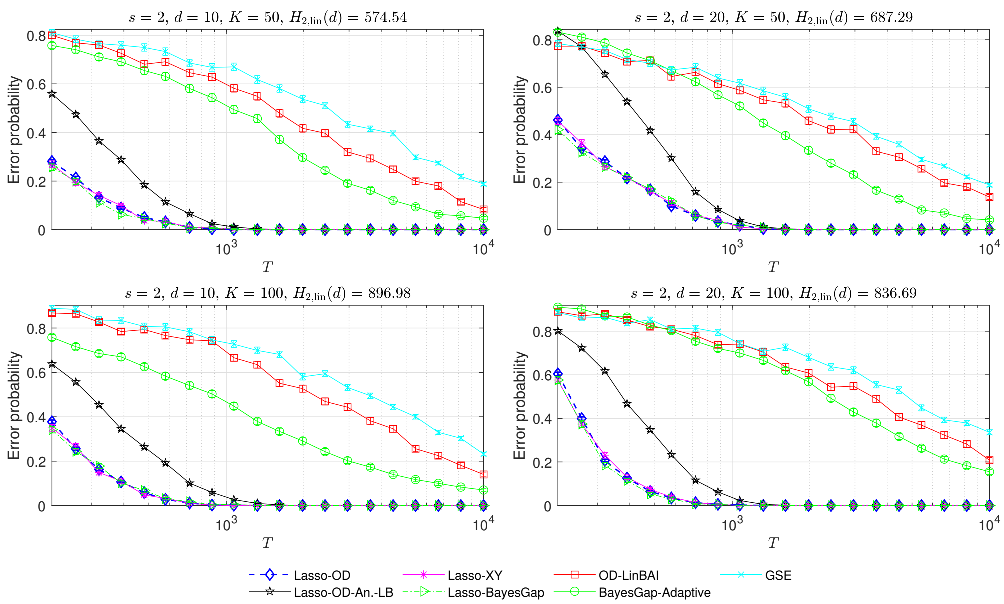

Figure 4: Comparison of several algorithms with θ^* belonging to a finite set.

G.2.3 Third Example

In the third example, we extend the example in [13, 15, 20] to sparse linear bandits. We set $\theta^* = (\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}, 0, \ldots, 0)$, i.e., $s = 2$, and $\mathcal{S} = S(\theta^*) = \{1, 2\}$. For the coordinates in $\mathcal{S}$, we pull arms as in [20]; we set $a(1)_{\mathcal{S}} = (\cos(\pi/4), \sin(\pi/4))$, $a(K)_{\mathcal{S}} = (\cos(5\pi/4), \sin(5\pi/4))$, and $a(i)_{\mathcal{S}} = (\cos(\pi/2 + \phi_i), \sin(\pi/2 + \phi_i))$ for $i = 2, \ldots, K-1$, where ϕ_i are independently drawn from $\mathcal{N}(0, 0.09)$. For any $i \in [K]$, we draw $a(i)_{\mathcal{S}^c}$ independently from the uniform distribution on the $(d - s)$-dimensional centered sphere of radius $\sqrt{\frac{d-s}{s}}$. Recall that since $\theta^*_{\mathcal{S}^c} = 0$, the values of arms on the coordinates $\mathcal{S}^c$ have no effect on the best arm or the value of the hardness parameter. The problem would be identical to that in [20] if the agent knew the support $\mathcal{S}$. In this bandit instance, arm 1 is the best arm and there are $K - 2$ arms whose mean values are close to that of the second best arm. In the non-sparse case, i.e., $d = s = 2$, Yang and Tan [20] demonstrate that OD-LinBAI outperforms

[2] Pre-calculation in Tables 2–3 refers to the calculation of $\phi^2(\mathsf{M}, s)$ that is used to determine the hyperparameters for Lasso-OD-Analytical.

[3] All experiments are implemented on MATLAB 2023a on an Intel(R) Core(TM) i9-12900H processor.

Table 2: The empirical means of the CPU runtimes for $s = 2$, $d = 10$, $K = 50$.

T	CPU runtimes (milliseconds)								
	Pre-calc.	Lasso Tuning	Lasso-OD	Lasso-$\mathcal{XY}$	Lasso-BayesG.	Lasso-OD-An.	BayesGap-Ad.	OD-LinBAI	GSE
100	9680	3300	0.98	9.0	1.6	1.7	3.2	2.2	4.0
200	9680	3500	0.61	5.2	2.9	1.2	5.1	2.2	4.0
400	9680	3800	0.55	3.6	5.5	0.83	9	2.1	4.0
800	9680	4380	0.48	3.4	11	0.76	17	2.1	4.0
1600	9680	6570	0.66	4.1	23	0.68	34	2.1	4.0
3200	9680	7520	0.66	4.0	46	0.70	69	2.1	4.0
6400	9680	7710	0.70	4.1	89	1.1	139	2.1	4.0

Table 3: The empirical means of the CPU runtimes for $s = 3$, $d = 10$, $K = 50$.

T	CPU runtimes (milliseconds)								
	Pre-calc.	Lasso Tuning	Lasso-OD	Lasso-$\mathcal{XY}$	Lasso-BayesG.	Lasso-OD-An.	BayesGap-Ad.	OD-LinBAI	GSE
100	27100	4200	1.1	12	1.6	2.2	2.5	2.4	4.2
200	27100	4500	0.96	8.9	2.9	2.1	5.2	2.4	4.1
400	27100	4900	0.94	8.1	5.7	2.3	10	2.4	4.1
800	27100	5000	0.98	7.6	12	2.3	17	2.4	4.2
1600	27100	5100	1.5	9.2	24	2.1	34	2.4	4.2
3200	27100	7000	1.5	9.2	47	2	68	2.4	4.2
6400	27100	9800	1.6	9.4	93	2	137	2.4	4.1

the other algorithms. Fig. 5 compares the performance of variants of our algorithm with the other algorithms in the literature. We report the empirical performances for $d \in \{10, 20\}$, $K \in \{50, 100\}$, and $T \in [200, 10^4]$. Among the algorithms shown, Lasso-OD and its variant Lasso-$\mathcal{XY}$-allocation significantly outperform the other algorithms. Unlike the previous two examples, for this example, Lasso-BayesGap is not the best performing algorithm.

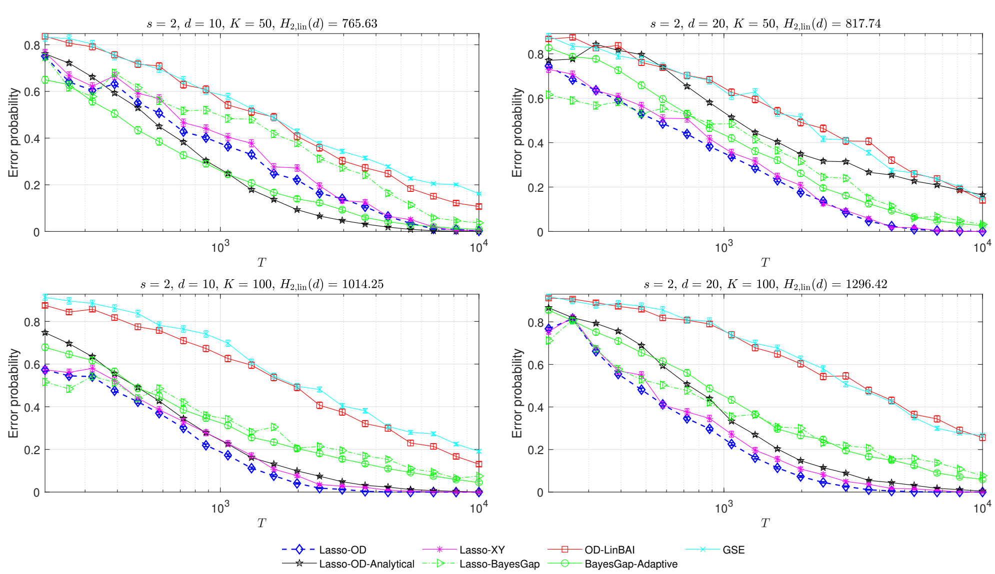

Figure 5: Comparison of several algorithms for the example bandit instance in [20].

G.2.4 Fourth Example

In the final example, we test the performance of thresholded Lasso in which the whole horizon of length T is used for learning the support of θ^*. We draw each entry of the design matrix $\mathsf{X} \in \mathbb{R}^{T\times d}$ i.i.d. from $\mathcal{N}(0, \frac{1}{s})$ and set $\theta^* = (\frac{1}{\sqrt{s}}, \ldots, \frac{1}{\sqrt{s}}, 0, \ldots, 0)$ where θ^* has s non-zero entries. In Fig. 6, we report the empirical probability of detection error $\mathbb{P}\left[S(\hat{\theta}_{\text{thres}}) \not\supseteq S(\theta^*)\right]$ and the empirical mean $\mathbb{E}[|S(\hat{\theta}_{\text{thres}})|]$ over 10,000 independent trials. For $s = 2$, the empirical error probability is 0 for $T \geq 400$; for $s = 4$, the empirical error probability is 0 for $T \geq 800$. Fig. 6 shows that for $T \geq 100$ and $s \in \{2, 4\}$, thresholded Lasso is capable of correctly detecting the active variables in θ^* with high probability while also keeping the average number of false positives close to zero. As expected, the average number of false positives is larger for larger s.

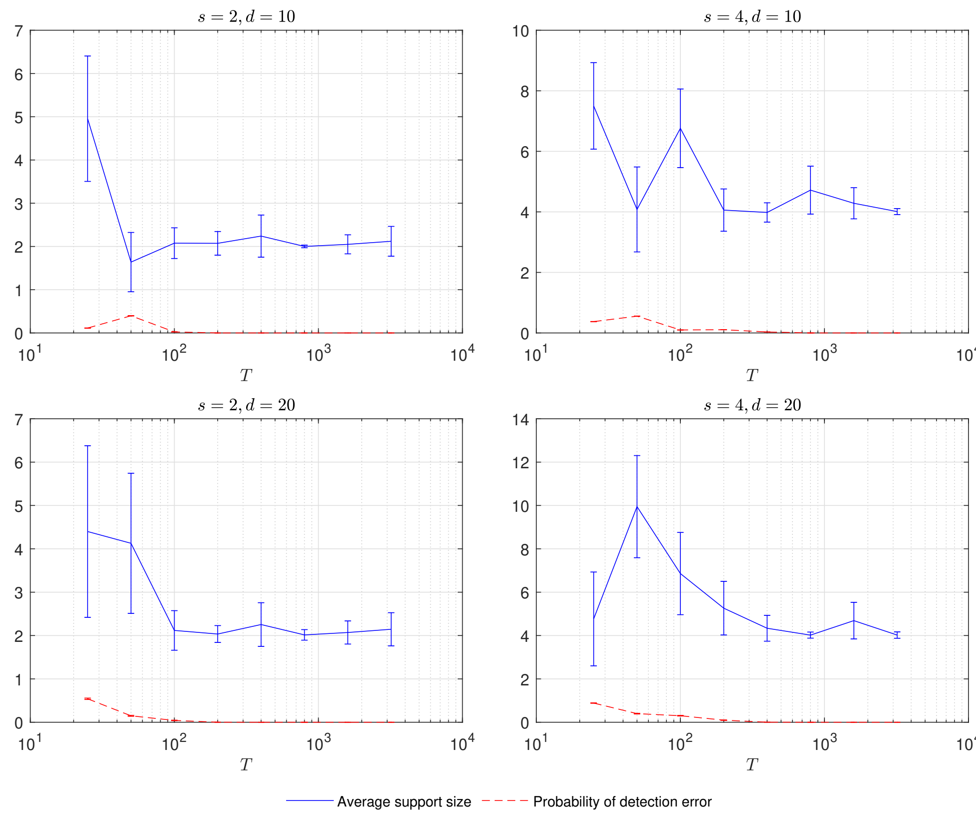

Figure 6: The empirical detection error probability and the empirical size of the thresholded Lasso output.