Online Regression with Partial Information: Generalization and Linear Projection

Shinji Ito
NEC Corporation
s-ito@me.jp.nec.com

Daisuke Hatano
National Institute of Informatics
hatano@nii.ac.jp

Hanna Sumita
National Institute of Informatics
sumita@nii.ac.jp

Akihiro Yabe
NEC Corporation
a-yabe@cq.jp.nec.com

Takuro Fukunaga
RIKEN AIP, JST PRESTO
takuro.fukunaga@riken.jp

Naonori Kakimura
Keio University
kakimura@math.keio.ac.jp

Ken-ichi Kawarabayashi
National Institute of Informatics
k-keniti@nii.ac.jp

Abstract

We investigate an online regression problem in which the learner makes predictions sequentially while only the limited information on features is observable. In this paper, we propose a general setting for the limitation of the available information, where the observed information is determined by a function chosen from a given set of observation functions. Our problem setting is a generalization of the online sparse linear regression problem, which has been actively studied. For our general problem, we present an algorithm by combining multi-armed bandit algorithms and online learning methods. This algorithm admits a sublinear regret bound when the number of observation functions is constant. We also show that the dependency on the number of observation functions is inevitable unless additional assumptions are adopted. To mitigate this inefficiency, we focus on a special case of practical importance, in which the observed information is expressed through linear combinations of the original features. We propose efficient algorithms for this special case. Finally, we also demonstrate the efficiency of the proposed algorithms by simulation studies using both artificial and real data.

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

1 Introduction

Motivation Online regression refers to the problem of making a prediction of properties for sequentially arriving examples. The prediction should be performed using the numerical features associated with each example, but the relationship between the features and properties to be predicted is not known. Hence, in order to reach a good prediction, it is necessary to learn a "hidden" model quickly in the situations in which not all examples are available simultaneously.

Even though this problem is difficult already, there are many real-word applications that require extending the problem setting by restricting the information available from the examples. For example, in a manufacturing process, we must predict whether or not products are defect-free through diagnostic tests [22]. Because these tests are costly, they cannot all be executed. In the medical diagnosis of a disease [7], it is undesirable to subject a patient to many medical tests. In such applications, predictions should be made using a limited number of features. Motivated by these applications, Kale et al. [11] formulated the *online sparse linear regression* (OSLR) problem, in which an algorithm is only allowed to observe a limited number of features per example.

However, there still exist applications that demand more general restrictions on the available information. For example, computed tomography (CT) [19] presents the Radon transforms of the original 2D images. This means that a learner can observe the linear projections of features. Similarly, in magnetic resonance imaging (MRI) [14] one can observe transformed information. In these cases, it is suitable to consider a setting in which a learner makes predictions from a limited number of projections of features. This setting cannot be captured by the OSLR problem.

Motivated by this fact, we consider an online regression problem with more general limitations on the available information. In this problem, the available information is represented by the output of an observation function f, chosen from a given set $\mathcal{F} \subseteq \{f \colon \mathbb{R}^d \to \mathbb{R}^k\}$. For each arriving example with the feature vector $\mathbf{x} \in \mathbb{R}^d$, an algorithm chooses $f \in \mathcal{F}$ and observes $f(\mathbf{x})$. Then, the algorithm makes a prediction on the label of the example. We evaluate performance of this algorithm by its regret, which is defined by comparison with an adversary. In our problem, we assume that a hypothesis space $\mathcal{H} \subseteq \{h \colon \mathbb{R}^k \to \mathbb{R}\}$ is also given, and the prediction of the adversary is defined as $h(f(\mathbf{x}))$ from the optimal choice of an observation function $f \in \mathcal{F}$ and a hypothesis $h \in \mathcal{H}$. This setting indeed generalizes OSLR, and also includes other practical situations. A more precise definition of the problem and special cases included by the problem are given in Sec. 2.

Contributions Our contributions are twofold. We first investigate the most general setting of our problem, and present an algorithm that achieves a nontrivial regret bound under reasonable assumptions. Second, we consider important special cases obtained by restricting the observation functions and hypotheses to linear mappings, and present better regression algorithms.

Let us now explain the result for the general setting. Our problem clearly includes the online learning problem [16] defined on the hypothesis space $\mathcal{H}$. Hence, if the online learning problem with $\mathcal{H}$ admits no efficient algorithm, then no regression algorithm achieves a nontrivial regret bound for our problem. Because of this relationship, we assume that the online learning problem with $\mathcal{H}$ admits a sublinear regret algorithm. For example, if the number of arriving examples is T, then an online learning algorithm achieves an $O(T^q)$ regret for some $q \in (0, 1)$. Under this assumption, we present an online regression algorithm that achieves a regret bound of $O(|\mathcal{F}|^{\frac{1-q}{3-2q}} T^{\frac{2-q}{3-2q}})$. This bound is clearly sublinear with respect to T. Although the dependence on $|\mathcal{F}|$ is undesirable when it is large (e.g., $|\mathcal{F}| = \binom{d}{k} = O(d^k)$ in OSLR), it is indeed impossible to achieve a regret bound that is independent of $|\mathcal{F}|$. Indeed, we prove that it is difficult to improve the regret bound over $O(\sqrt{|\mathcal{F}|T})$ unless additional assumptions are given on $\mathcal{F}$ or $\mathcal{H}$.

In the second result, we focus on practically important special cases, where the observation functions $\mathcal{F}$ and hypothesis spaces $\mathcal{H}$ consist of linear mappings. For the case that $\mathcal{F}$ is the set of all linear mappings from $\mathbb{R}^d$ to $\mathbb{R}^k$, we propose an algorithm to achieve an $O(\sqrt{dT/k})$ regret. Moreover, we consider the case in which the observation is given by a combination of k out of m given linear functions, which is a generalization of OSLR. We present an algorithm for this case with the aid of techniques developed in studies on sparse linear regression. When the error made by a sparse linear regression algorithm is bounded by ϵ, the regret of our algorithm is bounded by $O(\sqrt{mT/k} + \epsilon T)$.

Related work Attribute-efficient learning [7, 10] is a batch learning problem aiming to find a linear regressor using a limited number of features per example. The goal of attribute-efficient learning is not prediction, but rather computing the optimal linear regressor, and the performance of a given algorithm is evaluated by the cumulative loss calculated with *full access* to the features. In contrast to this setting, the goal of OSLR [8, 11, 12], which is a special case of our problem, includes not only learning, but also making predictions from a limited number of features. For OSLR, in the absence of any assumptions on input data only exponential-time algorithms [8, 11] are known, and Foster et al. [8] showed that there is no polynomial-time algorithm with sublinear regret unless $\mathrm{NP} \subseteq \mathrm{BPP}$. On the other hand, Kale et al. [12] proposed computationally efficient algorithms to achieve sublinear regret under the assumption that input features satisfy the *restricted isometory property* (RIP) [6]. Zolghadr et al. investigated a similar problem, called *online probing* [22], in which a learner *purchases* a subset of features in each round, and aims to minimize the sum of the cumulative prediction error and the observation cost (which depends on the features chosen to observe). Only computationally inefficient algorithms [22] have been created for this problem.

A common difficulty of our problem, OSLR, and online probing lies in the choice of features to observe in each round. This has a similar structure to the *multi-armed bandit* [2, 5] problem, which is a sequential decision problem with an exploration-exploitation trade-off. In a sequential manner, we need to observe as much informative data as possible (exploration), while we simultaneously have to make correct decisions in as many rounds as possible to minimize the loss (exploitation). A multi-armed bandit technique is used to design an algorithm for online probing [22]. Our first result also makes use of multi-armed bandit techniques. The connection to multi-armed bandit is also used to show the difficulty of our problem.

Another technique used in this paper is the well-establish paradigm of *online learning* [16, 9], which corresponds to our problem with full access to the features. Because a learner does not have full access to the features in our setting, online learning methods do not directly apply to our problem. However, online learning methods combined with a multi-armed bandit algorithm lead to an algorithm for our problem with finite $\mathcal{F}$. For the special case of linear $\mathcal{F}$ and $\mathcal{H}$, we use the dual averaging method [18] to compute a linear regressor.

2 Problem setting

We suppose that there are T rounds, and an example arrives online in each round. Each example is represented by d features, and is associated with a label. We denote the features of the example arriving in round t by $\mathbf{x}_t = (x_{t1}, \ldots, x_{td})^\top \in \{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x}\| \leq 1\}$, where $\|\cdot\|$ denotes the ℓ_2 norm. The label of the example in round t is denoted by $y_t \in [-1, 1]$.

The purpose of our problem is to predict the label $y_t \in [-1, 1]$ from a compressed observation of $\mathbf{x}_t$ in each round $t = 1, \ldots, T$. We suppose that the compressed observation $z_t = f_t(\mathbf{x}_t) \in \mathbb{R}^k$ is the output of a function $f_t : \mathbb{R}^d \to \mathbb{R}^k$, where f_t is chosen from a family $\mathcal{F} \subseteq \{f : \mathbb{R}^d \to \mathbb{R}^k\}$ of *observation functions*. Note that k is usually much smaller than d. The prediction is performed through the following four steps: (i) We choose an observation function $f_t \in \mathcal{F}$, (ii) observe the limited information $z_t = f_t(\mathbf{x}_t)$, (iii) on the basis of observation z_t, estimate a predictor $\hat{y}_t$ of y_t, and (iv) observe the true label y_t. It should be noted that we cannot observe $\mathbf{x}_t$ in this process.

Regret. Let $\ell(\hat{y}, y)$ be the loss function for evaluating how the prediction $\hat{y}$ deviates from the true label y. A typical choice of ℓ is the quadratic loss, i.e., $\ell(\hat{y}, y) = (\hat{y} - y)^2$. We use the quadratic loss in Sec. 4 and 5, while we consider arbitrary loss functions in Sec. 3. In our problem, the performance of the prediction is evaluated by the *regret* R_T, defined as

$$R_T = \sum_{t=1}^{T} \ell(\hat{y}_t, y_t) - \min_{f \in \mathcal{F}, h \in \mathcal{H}} \sum_{t=1}^{T} \ell(h(f(\mathbf{x}_t)), y_t), \quad (1)$$

where $\mathcal{H} \subseteq \{h : \mathbb{R}^k \to \mathbb{R}\}$ is a *hypothesis space*, which is a set of functions for predicting y_t from k-dimensional feature vectors. Our goal is to minimize the regret R_T.

Below, we give examples of observation functions $\mathcal{F}$ and hypothesis spaces $\mathcal{H}$.

Example 1 (online sparse linear regression [11, 8]). Define $\mathcal{F}$ and $\mathcal{H}$ by

$$\begin{aligned}\mathcal{F} = \mathcal{F}^{(k)}_{\mathrm{sp}} &:= \{f : \mathbb{R}^d \to \mathbb{R}^k \mid \\ &\quad \exists\{i_1, \ldots, i_k\} \subseteq [d], \quad [f(\mathbf{x})]_j = x_{i_j}\} \\ \mathcal{H} = \mathcal{H}^{(k)}_{\mathrm{lin}} &:= \{h : \mathbb{R}^k \to \mathbb{R} \mid \exists \mathbf{w} \in \mathbb{R}^k, \quad h(\mathbf{z}) = \mathbf{w}^\top \mathbf{z}\}.\end{aligned}$$

This case is equivalent to the online sparse linear regression problem introduced in [11]. Foster et al. [8] proved that there is no polynomial-time algorithm to achieve $O(\mathrm{poly}(d)T^{1-\delta})$ regret for any $\delta > 0$. Recently, Kale et al. [12] proposed a polynomial-time algorithm that achieves a sublinear regret under the assumption of the RIP condition.

Example 2. Let $A = [\mathbf{a}_1 \ldots \mathbf{a}_m]$ be a real $d \times m$ matrix, where $m \geq k$. Define $\mathcal{F}$ by

$$\begin{aligned}\mathcal{F} = \mathcal{F}^{(k)}_{A} &:= \{f : \mathbb{R}^d \to \mathbb{R}^k \mid \\ &\quad \exists\{i_1, \ldots, i_k\} \subseteq [m], \quad [f(\mathbf{x})]_j = \mathbf{a}_{i_j}^\top \mathbf{x}\}.\end{aligned}$$

That is, we choose k linear functions from m given linear functions, and observe their function values. The family $\mathcal{F}^{(k)}_A$ of observation functions is a generalization of $\mathcal{F}^{(k)}_{\mathrm{sp}}$ defined in Example 1. Indeed, we have that $\mathcal{F}^{(k)}_{\mathrm{sp}} = \mathcal{F}^{(k)}_A$, for the identity matrix A of size d. Thus, for this problem it is also difficult to achieve a sublinear regret, as implied by [8]. On the other hand, under an assumption on the matrix A, we can solve the problem efficiently. We will give details of the assumptions and algorithms in Sec. 5.

Example 3. Define $\mathcal{F}$ by

$$\mathcal{F} = \mathcal{F}^{(k)}_{\mathrm{lin}} := \{f : \mathbb{R}^d \to \mathbb{R}^k \mid \exists B \in \mathbb{R}^{k \times d}, f(\mathbf{x}) = B\mathbf{x}\},$$

i.e., $\mathcal{F}^{(k)}_{\mathrm{lin}}$ is the family of all linear maps from $\mathbb{R}^d$ to $\mathbb{R}^k$. Note that we have that $\mathcal{F}^{(k)}_{\mathrm{sp}} \subseteq \mathcal{F}^{(k)}_{\mathrm{lin}}$ and $\mathcal{F}^{(k)}_A \subseteq \mathcal{F}^{(k)}_{\mathrm{lin}}$. We will show in Sec. 4 that the problem with $\mathcal{F} = \mathcal{F}^{(k)}_{\mathrm{lin}}$ and $\mathcal{H} = \mathcal{H}^{(k)}_{\mathrm{lin}}$ admits a polynomial-time algorithm to achieve a sublinear regret.

3 Algorithm for finite $\mathcal{F}$

In this section, we suppose that $\mathcal{F}$ is a finite set. We remark that this case includes Examples 1 and 2 as special cases. In this case, we construct a principled method for choosing f_t, with the aid of a multi-armed bandit (MAB) algorithm. This method requires that the hypothesis space $\mathcal{H}$ admits an online learning algorithm. Sec. 3.1 explains this assumption in further detail. Sec. 3.2 introduces an algorithm for the MAB problem, which our algorithm uses as a subroutine. Sec. 3.3 presents our algorithm, and Sec. 3.4 presents a regret bound for our algorithm. Sec. 3.5 proves that a lower bound on the regret.

3.1 Online learning algorithm

We assume that there exists an online learning algorithm $\mathcal{A}$ that achieves an $\alpha(\tau)$ regret for the hypothesis space $\mathcal{H}$. That is, we assume the following: (i) Algorithm $\mathcal{A}$ takes inputs data $\mathbf{z}_1, y_1, \ldots, \mathbf{z}_\tau, y_\tau$ sequentially, where $\mathbf{z}_t \in \mathbb{R}^k$. For each $t = 1, 2, \ldots, \tau$, the algorithm receives $\mathbf{z}_t$, gives a prediction $\hat{y}_t$ of its label, and then receives the true label y_t. (ii) For arbitrary τ and arbitrary inputs $\{(\mathbf{z}_j, y_j)\}_{j=1}^{\tau}$, the regret R^{OL}_τ defined by

$$R^{\mathrm{OL}}_\tau = \sum_{j=1}^{\tau} \ell(\hat{y}_j, y_j) - \min_{h \in \mathcal{H}} \sum_{j=1}^{\tau} \ell(h(\mathbf{z}_j), y_j),$$

is bounded by $\alpha(\tau)$ in expectation, i.e., $\mathbf{E}[R^{\mathrm{OL}}_\tau] \leq \alpha(\tau)$.

Such algorithms are developed in the context of online learning or online optimization [16, 9]. For example, if $\mathcal{H}$ is a finite set and $\ell(\hat{y}, y)$ is a convex function of $\hat{y} - y$, then there exists an algorithm that admits an $\alpha(\tau) = O(\sqrt{\tau})$ regret [16]. Another typical example is the case that $\mathcal{H} =$

$\mathcal{H}^{(k)}_{\mathrm{lin}}$ and $\ell(\hat{y}, y) = (\hat{y} - y)^2$, for which there exists an algorithm to achieve an $\alpha(\tau) = O(\sqrt{\tau})$ regret [9]. It is known that under the condition that $\mathcal{H}$ has one-to-one correspondence to a compact convex set C, there is an algorithm $\mathcal{A}$ that achieves $\alpha(\tau) = O(\sqrt{\tau})$ if $\ell(h(\mathbf{z}), y)$ is a convex function[1] and $\alpha(\tau) = O(\log \tau)$ if it is strongly convex, as summarized, e.g., in [9].

Without loss of generality, we assume that $\ell(h(f(\mathbf{x})), y) \in [0, 1]$ for $f \in \mathcal{F}$, $h \in \mathcal{H}$, $y \in [-1, 1]$ and $\mathbf{x} \in \mathbb{R}^d$ such that $\|\mathbf{x}\| \leq 1$.

3.2 Multi-armed bandit problem

The multi-armed bandit (MAB) problem is a sequential decision making problem. In this problem, an instance is specified by the number K of available actions and a sequence $\{L_{k,t}\}_{t=1,\ldots,n} \in [0, 1]$ of losses from each action $k \in [K]$. Note that the loss sequences are not presented to the algorithm. In each round $t = 1, \ldots, n$, an algorithm selects an action $k_t \in [K]$ and observes the loss $L_{k_t,t}$ of the selected action. The performance of the algorithm is evaluated by the regret R^{MAB}_n defined by

$$R^{\mathrm{MAB}}_n = \sum_{t=1}^{n} L_{k_t,t} - \min_{k \in [K]} \sum_{t=1}^{n} L_{k,t}.$$

We assume that we have an MAB algorithm $\mathcal{B}$ to achieve $\mathbf{E}[R^{\mathrm{MAB}}_n] \leq \beta(n)$. It has been shown, e.g., in [1], that there exists an MAB algorithm $\mathcal{B}$ with $\beta(n) = O(\sqrt{Kn})$ for arbitrary loss sequences. For general MAB problems, it is known that $O(\sqrt{Kn})$ regret algorithms are minimax optimal, because every MAB algorithm admits an $\Omega(\sqrt{Kn})$ regret for some input sequences [3]. Under additional assumptions, such as stochastic models on loss sequences, there are more effective MAB algorithms. For more details on MAB algorithms, see, e.g., [5].

3.3 Algorithm

Our problem uses an algorithm $\mathcal{A}$ for the online learning problem associated with the hypothesis space $\mathcal{H}$, and an MAB algorithm $\mathcal{B}$. Suppose that $|\mathcal{F}| = K$ and denote $\mathcal{F} = \{f_k\}_{k=1}^{K}$. We assume for convenience that T is a multiple of an integer τ, which is optimized later.

Our algorithm, summarized in Algorithm 1, is performed as follows. We divide all rounds into blocks such that each block consists of consecutive τ rounds. The i-th block consists of rounds $i\tau + 1, i\tau + 2, \ldots, (i + 1)\tau$.

At the beginning of the i-th block, we define an instance of the MAB problem by regarding each observation function in $\mathcal{F}$ as an action, and apply the MAB algorithm $\mathcal{B}$ to select an action $f_{k_i} \in \mathcal{F}$. When the algorithm $\mathcal{B}$ is applied, the loss of the action $f_{k_{i'}}$ taken in the previous block i' $(< i)$ is defined by

Algorithm 1 An algorithm for finite $\mathcal{F} = \{f_k\}_{k=1}^{K}$

```
Input: An algorithm $\mathcal{A}$ for the online learning problem associated with the hypothesis space $\mathcal{H}$. An MAB algorithm $\mathcal{B}$. A positive integer $\tau$.
1: for $i = 0, \ldots, T/\tau - 1$ do
2:   Choose $k_i \in [K]$ by $\mathcal{B}$.
3:   Set $L_{k_i,i} = 0$.
4:   Initialize the state of $\mathcal{A}$.
5:   for $j = 1, \ldots, \tau$ do
6:     Observe $f_{k_i}(\mathbf{x}_{i\tau+j})$ and input it to $\mathcal{A}$.
7:     Determine $\hat{y}_{i\tau+j}$ by $\mathcal{A}$ and output it.
8:     Observe $y_{i\tau+j}$ and input it to $\mathcal{A}$.
9:     Set $L_{k_i,i} = L_{k_i,i} + \ell(\hat{y}_{i\tau+j}, y_{i\tau+j})/\tau$.
10:  end for
11:  Input $L_{k_i,i}$ to $\mathcal{B}$.
12: end for
```

$$L_{k_{i'},i'} = \frac{1}{\tau} \sum_{j=1}^{\tau} \ell(\hat{y}_{i'\tau+j}, y_{i'\tau+j}).$$

The selected observation function f_{k_i} is used through all the rounds in the i-th block.

At the round $i\tau + j$ in the i-th block, the prediction $\hat{y}_{i\tau+j}$ is computed by applying the algorithm $\mathcal{A}$ to the inputs $(f_{k_i}(\mathbf{x}_{i\tau+1}), y_{i\tau}), \ldots, (f_{k_i}(\mathbf{x}_{i\tau+j-1}), y_{i\tau+j-1}), f_{k_i}(\mathbf{x}_{i\tau+j})$.

3.4 Regret bound of Algorithm 1

In this section, we give an upper bound on the regret achieved by Algorithm 1.

Theorem 1. *Assume that the algorithm $\mathcal{A}$ achieves $\mathbf{E}[R^{\mathrm{OL}}_\tau] \leq \alpha(\tau)$, and that the algorithm $\mathcal{B}$ achieves $\mathbf{E}[R^{\mathrm{MAB}}_n] \leq \beta(n)$, where $\mathbf{E}[\cdot]$ stands for the expectation with respect to the algorithm's own randomization. Then, Algorithm 1 achieves the following regret bound:*

$$\mathbf{E}[R_T] \leq \frac{T}{\tau}\alpha(\tau) + \tau\beta\left(\frac{T}{\tau}\right).$$

Corollary 2. *If $\alpha(\tau) = O(\tau^q)$ for $q \in [0, 1]$ and $\beta(n) = O(\sqrt{Kn})$, then we obtain $\mathbf{E}[R_T] = O(K^{\frac{1-q}{3-2q}} T^{\frac{2-q}{3-2q}})$ by setting $\tau = \Theta((T/K)^{\frac{1}{3-2q}})$.*

Proof of Theorem 1. For each i and $k \neq k_i$, let $\hat{y}^{(k)}_{i\tau+1}, \ldots, \hat{y}^{(k)}_{(i+1)\tau}$ be the output from $\mathcal{A}$ for the input $(f_k(\mathbf{x}_{i\tau+1}), y_{i\tau+1}), \ldots, (f_k(\mathbf{x}_{(i+1)\tau}), y_{(i+1)\tau})$, which are not computed in Algorithm 1 but are used just for the analysis. Define $L_{k,i} = \frac{1}{\tau} \sum_{j=1}^{\tau} \ell(\hat{y}^{(k)}_{i\tau+j}, y_{i\tau+j})$. Then, for

[1] Under the correspondence $\mathcal{H} \simeq C$, $\ell(h(\mathbf{z}), y)$ is regarded as a function on C, for fixed $\mathbf{z}, y$.

arbitrary $k \in [K]$ and arbitrary $h \in \mathcal{H}$, we have that

$$\begin{aligned}
\mathbf{E}\left[\sum_{t=1}^{T} \ell(\hat{y}_t, y_t)\right] &= \tau \mathbf{E}\left[\sum_{i=0}^{T/\tau-1} L_{k_i,i}\right] \\
&\leq \tau\left(\mathbf{E}\left[\sum_{i=0}^{T/\tau-1} L_{k,i}\right] + \beta\left(\frac{T}{\tau}\right)\right) \\
&= \sum_{i=0}^{T/\tau-1} \mathbf{E}\left[\sum_{j=1}^{\tau} \ell(\hat{y}_{i\tau+j}^{(k)}, y_{i\tau+j})\right] + \tau\beta\left(\frac{T}{\tau}\right) \\
&\leq \sum_{t=1}^{T} \ell(h(f_k(\mathbf{x}_t)), y_t) + \frac{T\alpha(\tau)}{\tau} + \tau\beta\left(\frac{T}{\tau}\right),
\end{aligned}$$

where the first inequality comes from $\mathbf{E}[R_n^{\mathrm{MAB}}] \leq \beta(n)$, and the second inequality comes from $\mathbf{E}[R_\tau^{\mathrm{OL}}] \leq \alpha(\tau)$. Because the above inequality holds for all $k \in [K]$ and $h \in \mathcal{H}$, we have that $\mathbf{E}[R_T] \leq T\alpha(\tau)/\tau + \tau\beta(T/\tau)$. □

Corollary 2 implies that if $\mathcal{A}$ achieves a sublinear regret, i.e., $q < 1$, then Algorithm 1 also achieves a sublinear regret.

3.5 Regret lower bound for finite $\mathcal{F}$

The upper bound on the regret achieved by Algorithm 1 is at least $O(\sqrt{KT})$ if we use an MAB algorithm achieving a $\beta(n) = O(\sqrt{Kn})$ regret (minimax optimal [3]). This dependence on K is undesirable, because $K = |\mathcal{F}|$ may be extremely large in many practically important problems. For example, in Example 1, we have that $K = |\mathcal{F}_{\mathrm{sp}}^{(k)}| = \binom{d}{k}$, which grows exponentially w.r.t. k. However, the worst-case regret unfortunately cannot be improved over $\sqrt{KT}$, as implied by the following theorem.

Theorem 3. *Consider the problem in which $\mathcal{F} = \mathcal{F}_{\mathrm{sp}}^{(k)}$, $\ell(\hat{y}, y) = |\hat{y} - y|$, and $\mathcal{H} = \{h^*\}$, which is the set consisting of exactly one element defined by $h^*([z_i]_{i=1}^k) = \prod_{i=1}^k z_i$. For all algorithms with $T > 0$, there exist input sequences $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$ such that $\mathbf{E}[R_T] = \Omega(\sqrt{KT}) = \Omega(\sqrt{\binom{d}{k}T})$.*

The proof of this theorem is summarized in Appendix A of the supplementary material.

Note that in the case that $|\mathcal{H}| = 1$, as in this theorem, we can assume that $\alpha_\tau = 0$. Indeed, for the hypothesis space $\mathcal{H} = \{h^*\}$, the trivial online learning algorithm that outputs $\hat{y}_t = h^*(\mathbf{x}_t)$ achieves $R_T^{\mathrm{OL}} = 0$. Hence, for the problem setting in Theorem 3, Algorithm 1 with $\tau = 1$ achieves $\mathbf{E}[R_T] = O(\sqrt{KT})$, which matches the lower bound up to a constant factor.

Algorithm 2 An algorithm for $\mathcal{F} = \mathcal{F}_{\mathrm{lin}}^{(k)}$, $\mathcal{H} = \mathcal{H}_{\mathrm{lin}}^{(k)}$

```
Input: A positive real $D > 0$, a positive non-decreasing real sequence $\{\lambda_t\}$.
1: Set $\hat{\mathbf{h}}_0 = 0$.
2: for $t = 1, \ldots, T$ do
3:     Define $\mathbf{w}_t$ by (4).
4:     Randomly choose a set $S_t$ of $(k-1)$ elements from $[d]$, uniformly without replacement.
5:     Define $f_t$ by (2).
6:     Observe $f_t(\mathbf{x}_t)$ and output $\hat{y}_t = [f_t(\mathbf{x}_t)]_1 = \mathbf{w}_t^\top \mathbf{x}_t$.
7:     Observe $y_t$ and define $\hat{\mathbf{g}}_t$ by (3)
8:     Set $\hat{\mathbf{h}}_t = \hat{\mathbf{h}}_{t-1} + \hat{\mathbf{g}}_t$.
9: end for
```

4 Algorithm for $\mathcal{F}_{\mathrm{lin}}^{(k)}$ and $\mathcal{H}_{\mathrm{lin}}^{(k)}$

In this section, we consider the case when $\mathcal{F} = \mathcal{F}_{\mathrm{lin}}^{(k)}$ and $\mathcal{H} = \mathcal{H}_{\mathrm{lin}}^{(k)}$ (see Example 3). That is, in each round t, we construct a linear mapping $f_t \in \mathcal{F}_{\mathrm{lin}}^{(k)}$ and a vector $\mathbf{v}_t \in \mathbb{R}^k$, and compute a predictor $\hat{y}_t$ by $\hat{y}_t = \mathbf{v}_t^\top f_t(\mathbf{x}_t)$ from $\mathbf{x}_t$. In the following, we consider the quadratic loss $\ell(\hat{y}, y) = (\hat{y} - y)^2$.

4.1 Algorithm

Our algorithm always uses $\mathbf{v}_t = (1\ 0\ \ldots 0)^\top$, i.e., $\hat{y}_t = (f_t(\mathbf{x}_t))_1$ for each round t. To construct f_t, our algorithm first computes a weight vector $\mathbf{w}_t \in \mathbb{R}^d$ where the detalis are described below. Then the algorithm randomly chooses a set S_t of $k-1$ elements from $[d]$, uniformly without replacement. On the basis of $\mathbf{w}_t$ and $S_t = \{i_1, \ldots, i_{k-1}\} \subseteq [d]$, the observation function $g_t \in \mathcal{F}_{\mathrm{lin}}^{(k)}$ is determined by

$$f_t(\mathbf{x}) = [\mathbf{w}_t^\top \mathbf{x}, x_{i_1}, \ldots, x_{i_{k-1}}]^\top. \tag{2}$$

After observing $f_t(\mathbf{x}_t)$, the algorithm outputs the prediction $\hat{y}_t = [f_t(\mathbf{x}_t)]_1 = \mathbf{w}_t^\top \mathbf{x}_t$, and after that, observes the true label y_t. For reference, we describe the algorithm as Algorithm 2.

Computing $\mathbf{w}_t$. We compute $\mathbf{w}_t$ based on the dual averaging method [18]. For this purpose, we construct an unbiased estimator $\hat{\mathbf{g}}_j$ of the gradient $\mathbf{g}_j$ of the loss function in rounds $j = 1, \ldots, t-1$. Let us explain this in more details.

For a weight vector $\mathbf{w}$, define the loss function $\ell_j(\mathbf{w}) = (\hat{y}_j - y_j)^2 = (\mathbf{w}^\top \mathbf{x}_j - y_j)^2$. We denote the gradient of $\ell_j(\mathbf{w}_j)$ by $\mathbf{g}_j = \nabla \ell_j(\mathbf{w}_j) \in \mathbb{R}^d$. Then it holds that $\mathbf{g}_j = 2(\mathbf{x}_j^\top \mathbf{w}_j - y_j)\mathbf{x}_j = 2(\hat{y}_j - y_j)\mathbf{x}_j$. We remark that $\mathbf{g}_j$ cannot be computed without knowing the full information of $\mathbf{x}_j$.

Instead of $\mathbf{g}_j$, we use an estimator $\hat{\mathbf{g}}_j$ of $\mathbf{g}_j$, computed as follows. For all $i \in [d]$, let $\chi_i \in \mathbb{R}^d$ be the indicator vector of index i. Define $\hat{\mathbf{g}}_j \in \mathbb{R}^d$ by

$$\hat{\mathbf{g}}_j = \frac{2d}{k-1}(\hat{y}_j - y_j)\sum_{i \in S_j} x_{ji}\chi_i. \tag{3}$$

The vector $\hat{\mathbf{g}}_j$ can be computed from the observation $f_j(\mathbf{x}_j)$ and the true label y_j.

The weight $\mathbf{w}_t$ is defined from the estimators $\{\hat{\mathbf{g}}_j\}_{j=1,\ldots,t-1}$, a monotonically non-decreasing sequence $(\lambda_1, \ldots, \lambda_t)$ of positive numbers, and a positive real number $D > 0$. Let $\hat{\mathbf{h}}_{t-1} = \sum_{j=1}^{t-1} \hat{\mathbf{g}}_j$. Then we define $\mathbf{w}_t$ by

$$\begin{aligned}\mathbf{w}_t &= \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^d, \|\mathbf{w}\| \le D} \left\{ \hat{\mathbf{h}}_{t-1}^\top \mathbf{w} + \frac{\lambda_t}{2}\|\mathbf{w}\|^2 \right\} \\ &= -\frac{1}{\max\{\lambda_t, \|\hat{\mathbf{h}}_{t-1}\|/D\}}\hat{\mathbf{h}}_{t-1}. \end{aligned}\tag{4}$$

From the definition, $\mathbf{w}_t$ satisfies $\|\mathbf{w}_t\| \le D$.

4.2 Regret bound of Algorithm 2

In this subsection, we show that the regret achieved by Algorithm 2 is $O(d\sqrt{T})$ in expectation, as stated in the following theorem.

Theorem 4. *Assume that* $\mathbf{w}^* := \operatorname*{arg\,min}_{\mathbf{w} \in \mathbb{R}^d} \sum_{t=1}^T \|\mathbf{w}^\top \mathbf{x}_t - y_t\|^2$ *satisfies* $\|\mathbf{w}^*\|^2 \le D$. *Algorithm 2 achieves*

$$\mathbf{E}[R_T] \le \frac{2(D+1)^2 d}{k-1}\sum_{t=1}^T \frac{1}{\lambda_t} + \frac{D^2\lambda_{T+1}}{2}. \tag{5}$$

By setting $\lambda_t = \sqrt{\frac{8dt}{k-1}}$ *for each* $t = 1, \ldots, T$, *we obtain* $\mathbf{E}[R_T] \le (D+1)^2\sqrt{\frac{8d}{k-1}} \cdot \sqrt{T+1}$.

In the rest of this section, we prove Theorem 4. We first show that $\hat{\mathbf{g}}_t$ is an unbiased estimator of $\mathbf{g}_t$ and give an upper bound on $\mathbf{E}[\|\hat{\mathbf{g}}_t\|^2]$.

Lemma 5. *Function* $\hat{\mathbf{g}}_t$ *defined in* (3) *satisfies* $\mathbf{E}[\hat{\mathbf{g}}_t] = \mathbf{g}_t$. *Moreover,* $\mathbf{E}[\|\hat{\mathbf{g}}_t\|^2] \le \frac{4(D+1)^2 d}{k-1}$ *holds.*

The proof of this lemma is provided in Appendix B of the supplementary material.

Let us discuss the regret of Algorithm 2. Recall that the loss in each round t is expressed as $\ell_t(\mathbf{w}_t) = (\mathbf{w}_t^\top \mathbf{x}_t - y_t)^2$. We hence rewrite the regret R_T in (1) as

$$R_T = \max_{\mathbf{w} \in \mathbb{R}^d} \sum_{t=1}^T (\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{w})) \tag{6}$$

because $\min_{f \in \mathcal{F}^{(k)}_{\mathrm{lin}}, h \in \mathcal{H}^{(k)}_{\mathrm{lin}}} \sum_{t=1}^T (h(f(\mathbf{x}_t)) - y_t)^2 = \min_{\mathbf{w} \in \mathbb{R}^d} \sum_{t=1}^T \ell_t(\mathbf{w})$.

From the convexity of the loss function ℓ_t and Lemma 5, we obtain the following upper bound of the expected regret:

$$\begin{aligned}\mathbf{E}[R_T] &= \mathbf{E}\left[\sum_{t=1}^T (\ell_t(\mathbf{w}_t) - \ell_t(\mathbf{w}^*))\right] \\ &\le \mathbf{E}\left[\sum_{t=1}^T \mathbf{g}_t^\top(\mathbf{w}_t - \mathbf{w}^*)\right] = \mathbf{E}\left[\sum_{t=1}^T \hat{\mathbf{g}}_t^\top(\mathbf{w}_t - \mathbf{w}^*)\right]. \end{aligned}\tag{7}$$

The value $\sum_{t=1}^T \hat{\mathbf{g}}_t^\top(\mathbf{w}_t - \mathbf{w}^*)$ can be bounded via a standard analysis of the dual averaging method, as follows.

Lemma 6. *Suppose that* $\mathbf{w}_t$ *is defined by* (4) *for each* $t = 1, \ldots, T$, *and* $\mathbf{w}^* \in \mathbb{R}^d$ *satisfies* $\|\mathbf{w}^*\| \le D$. *Then, we have*

$$\sum_{t=1}^T \hat{\mathbf{g}}_t^\top(\mathbf{w}_t - \mathbf{w}^*) \le \sum_{t=1}^T \frac{\|\hat{\mathbf{g}}_t\|^2}{2\lambda_t} + \frac{D^2\lambda_{T+1}}{2}. \tag{8}$$

Proof. See, e.g., [18]. □

We are now ready to prove Theorem 4.

Proof of Theorem 4. By taking the expectation of both sides of (8) and applying Lemma 5, we obtain

$$\begin{aligned}\mathbf{E}\left[\sum_{t=1}^T \hat{\mathbf{g}}_t^\top(\mathbf{w}_t - \mathbf{w}^*)\right] &\le \sum_{t=1}^T \frac{\mathbf{E}[\|\hat{\mathbf{g}}_t\|^2]}{2\lambda_t} + \frac{D^2\lambda_{T+1}}{2} \\ &\le \frac{2(D+1)^2 d}{k-1}\sum_{t=1}^T \frac{1}{\lambda_t} + \frac{D^2\lambda_{T+1}}{2}.\end{aligned}$$

Combining the above inequality with the inequality (7), we obtain (5). □

5 Algorithm for $\mathcal{F}^{(k)}_A$ and $\mathcal{H}^{(k)}_{\mathrm{lin}}$

In this section, we consider the case that $\mathcal{F} = \mathcal{F}^{(k)}_A$ and $\mathcal{H} = \mathcal{H}^{(k)}_{\mathrm{lin}}$ (see Example 2). That is, in each round t we construct a mapping $f_t \in \mathcal{F}^{(k)}_A$ and a linear function $h_t(\mathbf{z}) = \mathbf{w}_t^\top \mathbf{z}$, and compute a predictor $\hat{y}_t$ by $\hat{y}_t = \mathbf{w}_t^\top f_t(\mathbf{x}_t)$ from $\mathbf{x}_t$. A mapping f_t corresponds to choosing k column indices $\{i_1, \ldots, i_k\}$ from the given matrix A.

Define a vector $\mathbf{u}_t \in \mathbb{R}^m$ by $u_{ti_j} = w_{tj}$ $(j \in [k])$ and $u_{ti} = 0$ $(i \in [m] \setminus \{i_j\}_{j=1}^k)$. Then, we can rewrite $\hat{y}_t = \mathbf{x}^\top A\mathbf{u}_t$. Therefore, instead of constructing f_t and $\mathbf{w}_t$, we attempt to find $\mathbf{u}_t$ in each round t. We denote the loss when we use a vector $\mathbf{u}$ by $\zeta_t(\mathbf{u}) = \ell_t(\mathbf{x}_t^\top A\mathbf{u}) = (\mathbf{x}_t^\top A\mathbf{u} - y_t)^2$.

Without loss of generality, we assume that the ℓ_2 operator norm $\|A\|_2 := \max_{\|\mathbf{u}\| \le 1} \|A\mathbf{u}\|$ of A satisfies $\|A\|_2 \le 1$.

5.1 Algorithm

Similar to Algorithm 2, we aim to compute $\mathbf{u}_t$ in each round such that $\sum_{t=1}^{T} \zeta_t(\mathbf{u}_t)$ is as small as possible, based on the dual averaging method. Let $\hat{\gamma}_j$ be an estimator of the gradient $\gamma_j = \nabla \zeta_j(\mathbf{u}_j)$ of the loss function, which is computed in the previous round j, and define $\hat{\eta}_{t-1} = \sum_{j=1}^{t-1} \hat{\gamma}_j$. Then, letting $(\lambda_1, \ldots, \lambda_T)$ be a monotonically non-decreasing sequence of positive numbers, we define $\tilde{\mathbf{u}}_t$ by

$$\tilde{\mathbf{u}}_t = \operatorname*{arg\,min}_{\mathbf{u} \in \mathbb{R}^m, \|\mathbf{u}\| \leq D} \left\{ \hat{\eta}_{t-1}^\top \mathbf{u} + \frac{\lambda_t}{2} \|\mathbf{u}\|^2 \right\} = - \frac{1}{\max\{\lambda_t, \|\hat{\eta}_{t-1}\|/D\}} \hat{\eta}_{t-1}, \tag{9}$$

where $D > 0$ is a fixed positive real number. Then, $\mathbf{x}_t^\top A \tilde{\mathbf{u}}_t$ may be suitable for use as $\hat{y}_t$.

The main difficulty is that the value $\mathbf{x}_t^\top A \tilde{\mathbf{u}}_t$ cannot be computed from the limited observation. In fact, to compute $\mathbf{x}_t^\top A \tilde{\mathbf{u}}_t$ it is necessary to observe $\mathbf{x}_t^\top A = [\mathbf{a}_1^\top \mathbf{x}_t \ldots \mathbf{a}_m^\top \mathbf{x}_t]$, while only k values from $\{\mathbf{a}_i^\top \mathbf{x}_t\}_{i=1}^m$ are available in our setting.

In order to mitigate this issue, our algorithm computes $\mathbf{u}_t \in \mathbb{R}^m$ from $\tilde{\mathbf{u}}_t$ such that $A\mathbf{u}_t$ is close to $A\tilde{\mathbf{u}}_t$, and the number k' of non-zero entries of $\mathbf{u}_t$ satisfies $k' < k$. Further detail are given below.

On the basis of $\mathrm{supp}(\mathbf{u}_t)$, the observation function $f_t \in \mathcal{F}_A^{(k)}$ is determined by

$$[f_t(\mathbf{x})]_{1 \ldots k'} = [\mathbf{a}_i^\top \mathbf{x}]_{i \in \mathrm{supp}(\mathbf{u}_t)}, \quad [f_t(\mathbf{x})]_{k'+1 \ldots k} = [\mathbf{a}_i^\top \mathbf{x}]_{i \in S}, \tag{10}$$

where S is a set of indices chosen uniformly at random from $\binom{[m]}{k-k'}$. Then, our algorithm outputs $\hat{y} := \mathbf{x}_t^\top A \mathbf{u}_t$, which can be computed from a part of the observation $[f_t(\mathbf{x}_t)]_{1 \ldots k'} = [\mathbf{a}_i^\top \mathbf{x}_t]_{i \in \mathrm{supp}(\mathbf{u}_t)}$, and is close to $\mathbf{x}_t^\top A \tilde{\mathbf{u}}_t$. The remainder of the observation $[f_t(\mathbf{x})]_{k'+1 \ldots k}$ is used to compute an unbiased estimator $\hat{\gamma}_t$ of the gradient γ_t, as described below.

Computing $\mathbf{u}_t$ from $\tilde{\mathbf{u}}_t$. Let k' be a fixed integer parameter satisfying $1 \leq k' \leq k-1$, which corresponds to the number of non-zero entries of the vector $\mathbf{u}_t$. When searching for $\mathbf{u}_t$ such that $A\mathbf{u}_t$ is close to $A\tilde{\mathbf{u}}_t$ and $\mathbf{u}_t$ has at most k' non-zero entries, we naturally arrive at the following optimization problem:

$$\text{Minimize} \quad \|A\tilde{\mathbf{u}}_t - A\mathbf{u}\| \quad \text{subject to } \|\mathbf{u}\|_0 \leq k', \tag{11}$$

which is a sparse linear regression. This problem is in general NP-hard [15], but there are many approximation algorithms for computing it, e.g., LASSO [17], orthogonal matching pursuit (OMP) [21], iterative hard thresholding (IHT) [4], and the forward-backward greedy algorithm

Algorithm 3 An algorithm for $\mathcal{F} = \mathcal{F}_A^{(k)}, \mathcal{H} = \mathcal{H}_{\mathrm{lin}}^k$

```
Input: A positive real $D > 0$, a positive non-decreasing real sequence $\{\lambda_t\}$, an integer $k' \in [1, k-1]$. The matrix $A$ corresponding to $\mathcal{F}_A^{(k)}$.
1: Set $\hat{\eta}_0 = 0$.
2: for $t = 1, \ldots, T$ do
3:    Define $\tilde{\mathbf{u}}_t$ by (9).
4:    Solve (11) and let $\mathbf{u}_t$ be the (approximate) solution to (11).
5:    Randomly choose a set $S_t$ of $(k - k')$ elements from $[d]$ uniformly, without replacement.
6:    Output $\hat{y}_t = \mathbf{u}_t^\top A^\top \mathbf{x}_t$.
7:    Observe $y_t$ and define $\hat{\gamma}_t$ by (12).
8:    Set $\hat{\eta}_t = \hat{\eta}_{t-1} + \hat{\gamma}_t$.
9: end for
```

[20]. These algorithms are proven to achieve a small objective value, under the assumptions that A satisfies a sufficient condition for sparse recovery, such as the restricted isometry property [6], and that the true optimal value of (11) is close to zero.

Our procedure employs an arbitrary algorithm for approximately solving the problem (11) to determine $\mathbf{u}_t$. Let $\epsilon_t > 0$ denote the achieved objective value of (11), i.e., define $\epsilon_t := \|A\tilde{\mathbf{u}}_t - A\mathbf{u}_t\|$ for computed $\mathbf{u}_t$.

Computing $\hat{\gamma}_t$. Consider estimating $\gamma_t = \nabla \zeta_t(\mathbf{u}_t) = 2(\mathbf{x}_t^\top A \mathbf{u}_t - y_t) A^\top \mathbf{x}_t = 2(\hat{y}_t - y_t) A^\top \mathbf{x}_t$. Because $\hat{y}_t$ is directly computed from a part of the observation $[f_t(\mathbf{x}_t)]_{1 \ldots k'}$ and y_t is observed after returning $\hat{y}_t$, we need to estimate $A^\top \mathbf{x}_t$ alone. This $A^\top \mathbf{x}_t$ can be estimated from the remainder of the observation $[f_t(\mathbf{x}_t)]_{k'+1 \ldots k} = [\mathbf{a}_i^\top \mathbf{x}_t]_{i \in S}$. In fact, $\frac{m}{k-k'} \sum_{i \in S} \mathbf{a}_i^\top \mathbf{x}_t \chi_i$ is an unbiased estimator of $A^\top \mathbf{x}_t$, where $\chi_i \in \mathbb{R}^m$ is the indicator vector of the index i. Accordingly, $\hat{\gamma}_t$ is defined by

$$\hat{\gamma}_t = \frac{2m}{k-k'} (\hat{y}_t - y_t) \sum_{i \in S} \mathbf{a}_i^\top \mathbf{x}_t \chi_i. \tag{12}$$

Lemma 7. *$\hat{\gamma}_t$ defined in (12) is an unbiased estimator of γ_t, i.e., we have that $\mathbf{E}[\hat{\gamma}_t] = \gamma_t$. The expected value of $\|\hat{\gamma}_t\|^2$ is bounded by $\mathbf{E}[\|\hat{\gamma}_t\|^2] \leq \frac{4(D+1)^2 m}{k-k'}$.*

Proof. Because $\|A\|_2 \leq 1$ and $\|\mathbf{x}_t\| \leq 1$, we have that $\|A^\top \mathbf{x}_t\| \leq 1$. Using this fact, the lemma is proved in a similar manner to Lemma 5. □

5.2 Regret bound of Algorithm 3

In this subsection, we show that the regret achieved by Algorithm 3 is $O(d\sqrt{T})$ in expectation, as claimed in the following theorem.

Theorem 8. *Assume that* $\mathbf{u}^* := \arg\min_{u \in \mathbb{R}^m, \|\mathbf{u}\|_0 \le k} \sum_{t=1}^T \|\mathbf{u}^\top A^\top \mathbf{x}_t - y_t\|^2$ *satisfies* $\|\mathbf{u}^*\|^2 \le D$. *Then, Algorithm 3 achieves*

$$\mathbf{E}[R_T] \le \frac{2(D+1)^2 m}{k-k'} \sum_{t=1}^T \frac{1}{\lambda_t} + \frac{D^2 \lambda_{T+1}}{2} + 2(D+1)\mathbf{E}\left[\sum_{t=1}^T \epsilon_t\right]$$

where ϵ_t *is the achieved objective value of* (11). *By setting* $\lambda_t = \sqrt{\frac{8mt}{k-k'}}$ *for each* $t = 1, \ldots, T$, *we obtain* $\mathbf{E}[R_T] \le (D+1)^2 \sqrt{\frac{8m}{k-k'}} \cdot \sqrt{T+1} + 2(D+1) \sum_{t=1}^T \mathbf{E}[\epsilon_t]$.

Proof. The regret R_T can be expressed as

$$\begin{aligned} R_T &= \max_{\mathbf{u} \in \mathbb{R}^m, \|\mathbf{u}\|_0 \le k} \sum_{t=1}^T (\ell_t(A\mathbf{u}_t) - \ell_t(A\mathbf{u})) \\ &= \max_{\mathbf{u} \in \mathbb{R}^m, \|\mathbf{u}\|_0 \le k} \sum_{t=1}^T (\zeta_t(\mathbf{u}_t) - \zeta_t(\mathbf{u})). \end{aligned}$$

From the convexity of the loss function ζ_t and Lemma 5, we obtain

$$\mathbf{E}[R_T] = \mathbf{E}\left[\sum_{t=1}^T (\zeta_t(\mathbf{u}_t) - \zeta_t(\mathbf{u}^*))\right] \le \mathbf{E}\left[\sum_{t=1}^T \gamma_t^\top (\mathbf{u}_t - \mathbf{u}^*)\right].$$

Because $\|A\mathbf{u}_t - A\tilde{\mathbf{u}}_t\| = \|A(\mathbf{u}_t - \tilde{\mathbf{u}}_t)\| = \epsilon_t$, we have that

$$\begin{aligned} \gamma_t^\top (\mathbf{u}_t - \mathbf{u}^*) &= \gamma_t^\top (\tilde{\mathbf{u}}_t - \mathbf{u}^*) + \gamma_t^\top (\mathbf{u}_t - \tilde{\mathbf{u}}_t) \\ &= \gamma_t^\top (\tilde{\mathbf{u}}_t - \mathbf{u}^*) + 2(\hat{y}_t - y_t) A(\mathbf{u}_t - \tilde{\mathbf{u}}_t) \\ &\le \gamma_t^\top (\tilde{\mathbf{u}}_t - \mathbf{u}^*) + 2(D+1)\epsilon_t. \end{aligned}$$

Combining the above two inequalities, we obtain

$$\mathbf{E}[R_T] \le \mathbf{E}\left[\sum_{t=1}^T \gamma_t^\top (\tilde{\mathbf{u}}_t - \mathbf{u}^*)\right] + 2(D+1) \sum_{t=1}^T \mathbf{E}[\epsilon_t]. \tag{13}$$

The first term on the right-hand side can be bounded as

$$\begin{aligned} \mathbf{E}\left[\sum_{t=1}^T \gamma_t^\top (\tilde{\mathbf{u}}_t - \mathbf{u}^*)\right] &= \mathbf{E}\left[\sum_{t=1}^T \hat{\gamma}_t^\top (\tilde{\mathbf{u}}_t - \mathbf{u}^*)\right] \\ &\le \mathbf{E}\left[\sum_{t=1}^T \frac{\|\hat{\gamma}_t\|^2}{2\lambda_t} + \frac{D^2 \lambda_{T+1}}{2}\right] \\ &\le \frac{2(D+1)^2 m}{k-k'} \sum_{t=1}^T \frac{1}{\lambda_t} + \frac{D^2 \lambda_{T+1}}{2} \end{aligned} \tag{14}$$

where the first and the second inequalities come from Lemma 7, and the third inequality comes from Lemma 6. By combining (13) and (14), we obtain the desired result. □

6 Experiments

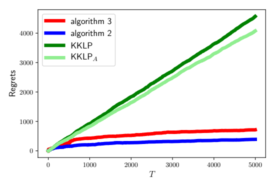
In this section, we evaluate Algorithms 2 and 3 through numerical experiments. To solve (11) in Algorithm 3, we employ the orthogonal matching pursuit method [21]. We use the algorithm proposed in [12, Algorithm2] as a baseline. Note that this baseline algorithm is designed for OSLR, and hence it cannot be applied to the more general settings considered in Sec. 4 and 5. To evaluate the performance of our algorithms in these general settings, we apply the baseline algorithm as follows.

Recall that Algorithm 2 receives a feature vector $\mathbf{x}_t \in \mathbb{R}^d$, chooses a linear mapping $f \in \mathcal{F}_{\text{lin}}^{(k)}$, and estimates the true label y_t from the observation $f(\mathbf{x}_t) \in \mathbb{R}^k$ in each round t. To evaluate Algorithm 2, we apply the baseline algorithm to the feature vector $\mathbf{x}_t$ directly. Then, the baseline algorithm makes a prediction after observing k entries in $\mathbf{x}_t$. We call this version of the baseline Kale-Karnin-Liang-Pál (KKLP).

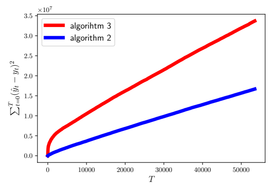
Algorithm 3 makes a prediction after observing k entries chosen from $A^\top \mathbf{x}_t$. To evaluate this algorithm, we apply the baseline algorithm to $A^\top \mathbf{x}_t$. Hence, it observes k entries of $A^\top \mathbf{x}_t$. We call this version KKLP_A.

In this section, we define the regrets of these algorithms by comparison with an adversary that observes only k entries of $\mathbf{x}_t$. This is slightly different from the definition (1). Because of space limitations, we only present typical results here. Other results and detailed descriptions of experiment settings are provided in the supplementary material.

Synthetic data. First, we present results for synthetic datasets. For each combination of parameters (d, k, k'), we executed all algorithms on five instances with $T = 5000$, and computed the averages of the final regrets and execution time. Figure 1(a) plots the regrets over the number of rounds on a typical instance with $(d, k, k') = (10, 2, 4)$. In this instance, Algorithms 2 and 3, KKLP, and KKLP_A required 0.32, 2.31, 4.58 and 46.76 seconds, respectively. Table 1 summarizes the final regrets for small instances. We observe that our algorithms achieve the smallest regrets.



(a) A typical result for synthetic datasets.



(b) The square loss for CT-slice datasets.

Figure 1: The regrets over the number of rounds.

Table 1: Average regrets.

(d, k', k)	Algorithm 2	Algorithm 3	KKLP	KKLP_A
(10, 2, 4)	427.0	2544.3	4649.4	4335.0
(10, 2, 8)	265.2	1271.7	4365.8	3952.8
(10, 5, 8)	245.8	787.7	4370.4	4087.1

Real data. Next, we describe the experiments using a CT-slice dataset, which is available online [13]. Each data item consists of 384 features retrieved from 53500 CT images associated with a label that denotes the relative position of an image on the axial axis.

Because we do not know the ground-truth regression weights, we measure the performance by the first term of (1), i.e., the square losses of predictions. Figure 1(b) plots the losses over the number of rounds, where the parameters are $k = 50$ and $k' = 40$. For this instance, the run times of Algorithms 2 and 3 were 6.6 and 11035.7 seconds, respectively. We did not plot the regrets for KKLP and KKLP_A, because they did not terminate within three hours. We observe that our algorithms can efficiently process large instances, while the baseline algorithms cannot.

Acknowledgement

This work was supported by JST ERATO Grant Number JPMJER1201, Japan. The fifth author was supported by JST, PRESTO Grant Number JPMJPR1759, Japan.

References

[1] J.-Y. Audibert and S. Bubeck. Minimax policies for adversarial and stochastic bandits. In *COLT*, pages 217–226, 2009.

[2] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2-3):235–256, 2002.

[3] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

[4] T. Blumensath and M. E. Davies. Iterative hard thresholding for compressed sensing. *Applied and computational harmonic analysis*, 27(3):265–274, 2009.

[5] S. Bubeck, N. Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 5(1):1–122, 2012.

[6] E. J. Candes and T. Tao. Decoding by linear programming. *IEEE transactions on information theory*, 51(12):4203–4215, 2005.

[7] N. Cesa-Bianchi, S. Shalev-Shwartz, and O. Shamir. Efficient learning with partially observed attributes. *Journal of Machine Learning Research*, 12(Oct):2857–2878, 2011.

[8] D. Foster, S. Kale, and H. Karloff. Online sparse linear regression. In *29th Annual Conference on Learning Theory*, pages 960–970, 2016.

[9] E. Hazan et al. Introduction to online convex optimization. *Foundations and Trends® in Optimization*, 2(3-4):157–325, 2016.

[10] E. Hazan and T. Koren. Linear regression with limited observation. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 807–814, 2012.

[11] S. Kale. Open problem: Efficient online sparse regression. In *Proceedings of The 27th Conference on Learning Theory*, pages 1299–1301, 2014.

[12] S. Kale, Z. Karnin, T. Liang, and D. Pál. Adaptive feature selection: Computationally efficient online sparse linear regression under RIP. In *Proceedings of the 34th International Conference on Machine Learning (ICML2017)*, pages –, 2017.

[13] M. Lichman. UCI machine learning repository, 2013.

[14] M. Lustig, D. L. Donoho, J. M. Santos, and J. M. Pauly. Compressed sensing mri. *IEEE signal processing magazine*, 25(2):72–82, 2008.

[15] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM journal on computing*, 24(2):227–234, 1995.

[16] S. Shalev-Shwartz et al. Online learning and online convex optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2012.

[17] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[18] L. Xiao. Dual averaging methods for regularized stochastic learning and online optimization. *Journal of Machine Learning Research*, 11(Oct):2543–2596, 2010.

[19] H. Yu and G. Wang. Compressed sensing based interior tomography. *Physics in medicine and biology*, 54(9):2791, 2009.

[20] T. Zhang. Adaptive forward-backward greedy algorithm for sparse learning with linear models. In *Advances in Neural Information Processing Systems*, pages 1921–1928, 2009.

[21] T. Zhang. Sparse recovery with orthogonal matching pursuit under rip. *IEEE Transactions on Information Theory*, 57(9):6215–6221, 2011.

[22] N. Zolghadr, G. Bartók, R. Greiner, A. György, and C. Szepesvári. Online learning with costly features and labels. In *Advances in Neural Information Processing Systems*, pages 1241–1249, 2013.

Appendix

A Proof of Theorem 3

First we provide an instance of the input sequence satisfying the condition in Theorem 3. The construction of the instance and the proof of the lower bound are given in a quite similar way to the proof of the well-known bandit lower bound shown in [3]. In our proof, each observation function in $\mathcal{F}_{\rm sp}^{(k)}$ in our problem corresponds to each arm in the MAB problem.

Let ϵ be a fixed real number in $(0, 1/2)$. For $S \in \binom{[d]}{k}$, let $\mathcal{D}_S$ be a probabilistic distribution on $\mathbb{R}^d \times \mathbb{R}$ such that every sample $(\mathbf{x}, y)$ drawn from $\mathcal{D}_S$ is generated by $\mathbf{x} \sim U(\{-1,1\}^d)$ and $y = \sigma h^*(f_S(\mathbf{x}))$, where $U(\{-1,1\}^d)$ is the uniform distribution on $\{-1,1\}^d$ and σ is a $\{-1,1\}$-random variable independent of $\mathbf{x}$, defined by $\mathrm{Prob}[\sigma = 1] = \frac{1}{2} + \epsilon$ and $\mathrm{Prob}[\sigma = -1] = \frac{1}{2} - \epsilon$. We will show that, for any algorithm, there exists $S \in \binom{[d]}{k}$ such that

$$\mathbf{E}[\sum_{t=1}^{T} \ell(\hat{y}_t, y_t) - \sum_{t=1}^{T} \ell(h^*(f_S(\mathbf{x}_t)), y_t] \geq 2T\epsilon(1 - \frac{1}{K} - \sqrt{\epsilon \ln \frac{1+\epsilon}{1-\epsilon}}\sqrt{\frac{T}{2K}}) \tag{15}$$

where the expectations are with respect to both the random generation of $\{(\mathbf{x}_t, y_t)\}_{t=1}^T$ from $\mathcal{D}_S^T$ and the internal randomization of the algorithm. The above inequality implies that, for all algorithms and $\epsilon \in (0, \frac{1}{2})$, there exists an input sequence such that $\mathbf{E}[R_T] \geq 2T\epsilon(1 - \frac{1}{K} - \sqrt{\epsilon \ln \frac{1+\epsilon}{1-\epsilon}}\sqrt{\frac{T}{2K}})$. The lower bound in Theorem 3 follows by a simple optimization over ϵ.

In the rest of this section, we provide a proof of (15). For $S = \{i_1, \ldots, i_k\} \in \binom{[d]}{k}$, define $f_S : \mathbb{R}^d \to \mathbb{R}^k$ by $[f_S(\mathbf{x})]_j = x_{i_j}$. Then we have $\mathcal{F}_{\rm sp}^{(k)} = \{f_S | S \in \binom{[d]}{k}\}$.

For an arbitrary algorithm and input sequence, let $n_S(T)$ denote the number of times that the observation function $f_S \in \mathcal{F}_{\rm sp}^{(k)}$ is selected in the first T rounds. Note that $n_S(T)$ is a random variable if the algorithm is a randomized one or the input sequence is drawn from a probabilistic distribution. Then, the accumulated loss for input sequences from $\mathcal{D}_S$ can be bounded by using $n_S(T)$, as follows:

Lemma 9. *Suppose that the input $(\mathbf{x}_t, y_t)$ follows $\mathcal{D}_S$ independently for $t = 1, 2, \ldots, T$. Then, for arbitrary algorithms, we have*

$$\mathbf{E}[\sum_{t=1}^{T} \ell(\hat{y}_t, y_t) - \sum_{t=1}^{T} \ell(h^*(f_S(\mathbf{x}_t)), y_t)] \geq 2\epsilon(T - \mathbf{E}[n_S(T)]). \tag{16}$$

Proof. First, we will show that

$$\mathbf{E}[\ell(\hat{y}_t, y_t)] \geq \begin{cases} 1 - 2\epsilon & (f_t = f_S) \\ 1 & (\text{otherwise}) \end{cases}. \tag{17}$$

Suppose that $f_t = f_S$. From the definition of $\mathcal{D}_S$, y_t is expressed as $y_t = \sigma_t h^*(f_S(\mathbf{x}_t))$ where σ_t is independent of all observation except for y_t and $\mathrm{Prob}[\sigma_t = 1] = \frac{1}{2} + \epsilon$ and $\mathrm{Prob}[\sigma_t = -1] = \frac{1}{2} - \epsilon$. Since $\hat{y}_t$ does not depend on y_t, σ_t and $\hat{y}_t$ are also independent. Hence, we have

$$\begin{aligned}
&\mathbf{E}[\ell(\hat{y}_t, y_t)] = \mathbf{E}[|\hat{y}_t - \sigma_t h^*(f_S(\mathbf{x}_t))|] \\
&= (\frac{1}{2} + \epsilon)\mathbf{E}[|\hat{y}_t - h^*(f_S(\mathbf{x}_t))|] + (\frac{1}{2} - \epsilon)\mathbf{E}[|\hat{y}_t + h^*(f_S(\mathbf{x}_t))|] \\
&= \frac{1}{2}\mathbf{E}[|\hat{y}_t - h^*(f_S(\mathbf{x}_t))| + |\hat{y}_t + h^*(f_S(\mathbf{x}_t))|] \\
&\quad + \epsilon\mathbf{E}[|\hat{y}_t - h^*(f_S(\mathbf{x}_t))| - |\hat{y}_t + h^*(f_S(\mathbf{x}_t))|] \\
&\geq \frac{1}{2}\mathbf{E}[|2h^*(f_S(\mathbf{x}_t))|] - \epsilon\mathbf{E}[|2h^*(f_S(\mathbf{x}_t))|] = 1 - 2\epsilon,
\end{aligned}$$

where the second and the last equalities come respectively from the stochastic independence of σ_t and that $h^*(f_S(\mathbf{x}_t)) \in \{-1, 1\}$, and the inequality comes from the triangle inequality. The first half of (17) follows from the above inequality. Next we show the second half of (17). In the case of $f_t = f_{S'}$ for $S' \neq S$, the posterior probability distribution of y_t given $f_t(\mathbf{x}_t)$ is the uniform distribution $U(\{-1,1\})$. Indeed, since we have $y_t = \sigma_t \prod_{i \in S \cap S'} x_{ti} \cdot \prod_{i \in S \setminus S'} x_{ti}$, $\sigma_t \prod_{i \in S \cap S'} x_{ti} \in \{-1, 1\}$ and since $\prod_{i \in S \setminus S'} x_{ti}$ is a random variable following $U(\{-1,1\})$ independently of $f_{S'}(\mathbf{x}_t)$, we have $\prod[y_t = 1] = \prod[y_t = -1] = 1/2$ given $f_{S'}(\mathbf{x}_t)$. Hence, we have

$$\begin{aligned}
&\mathbf{E}[\ell(\hat{y}_t, y_t)] = \mathbf{E}[|\hat{y}_t - \sigma_t h^*(f_{S'}(\mathbf{x}_t))|] \\
&= \frac{1}{2}\mathbf{E}[|1 - \sigma_t h^*(f_{S'}(\mathbf{x}_t))|] + \frac{1}{2}\mathbf{E}[|-1 - \sigma_t h^*(f_{S'}(\mathbf{x}_t))|] \\
&= \frac{1}{2}\mathbf{E}[|1 - \sigma_t h^*(f_{S'}(\mathbf{x}_t))| + |1 + \sigma_t h^*(f_{S'}(\mathbf{x}_t))|] = 1,
\end{aligned}$$

which implies that (17) holds. From (17) we have

$$\begin{aligned}
\sum_{t=1}^{T} \mathbf{E}[\ell(\hat{y}_t, y_t)] &\geq \mathbf{E}[(1 - 2\epsilon) \cdot n_S(T) + 1 \cdot (T - n_S(T))] \\
&= T - 2\epsilon\mathbf{E}[n_S(T)].
\end{aligned} \tag{18}$$

Moreover, we have

$$\begin{aligned}
&\mathbf{E}[\ell(h^*(f_S(\mathbf{x}_t)), y_t)] = \mathbf{E}[|(1 - \sigma_t)h^*(f_S(\mathbf{x}_t))|] \\
&= (\frac{1}{2} + \epsilon)\mathbf{E}[|0 \cdot h^*(f_S(\mathbf{x}_t))|] + (\frac{1}{2} - \epsilon)\mathbf{E}[|2 \cdot h^*(f_S(\mathbf{x}_t))|] \\
&= 1 - 2\epsilon.
\end{aligned} \tag{19}$$

Combining (18) and (19), we obtain (16). □

From the proof of [5, Lemma 3.6.], the right-hand side of (16) is bounded as

$$\epsilon(T - \mathbf{E}[n_S(T)]) \geq T\epsilon(1 - \frac{1}{K} - \sqrt{\epsilon \ln \frac{1+\epsilon}{1-\epsilon}}\sqrt{\frac{T}{2K}})$$

for some $S \in \binom{[d]}{k}$ and random inputs $(\mathbf{x}_t, y_t)$ drawn from $\mathcal{D}_S$. This inequality and (16) lead to (15).

B Proof of Lemma 5

Proof. We have $\mathbf{E}[\sum_{i \in S_t} x_{ti}\chi_i] = \sum_{j=1}^d \mathrm{Prob}[j \in S_t] x_{tj}\chi_j$. Since the definition of S_t implies that $\mathrm{Prob}[j \in S_t] = (k-1)/d$ for all j, we have $\mathbf{E}[\sum_{i \in S_t} x_{ti}\chi_i] = \frac{k-1}{d}\sum_{j=1}^d x_{tj}\chi_j = \frac{k-1}{d}\mathbf{x}_t$. Hence, $\hat{\mathbf{g}}_t$ defined by (3) satisfies $\mathbf{E}[\hat{\mathbf{g}}_t] = \frac{2d}{k-1}(\hat{y}_t - y_t)\mathbf{E}[\sum_{i \in S_t} x_{ti}\chi_i] = 2(\hat{y}_t - y_t)\mathbf{x}_t = \mathbf{g}_t$.

Next, we consider $\mathbf{E}[\|\hat{\mathbf{g}}_t\|^2]$. We have

$$\mathbf{E}[\|\hat{\mathbf{g}}_t\|^2] = \mathbf{E}[\hat{\mathbf{g}}_t^\top \hat{\mathbf{g}}_t]$$
$$= \frac{4d^2}{(k-1)^2}(\hat{y}_t - y_t)^2 \mathbf{E}\left[\left(\sum_{i \in S_t} x_{ti}\chi_i\right)^\top \left(\sum_{j \in S_t} x_{tj}\chi_j\right)\right]. \tag{20}$$

Since $\|\mathbf{x}_t\| \leq 1$, $|y_t| \leq 1$ and $\|\mathbf{w}_t\| \leq D$ from the definition (4) of $\mathbf{w}_t$, we have

$$(\hat{y}_t - y_t)^2 = (\mathbf{w}_t^\top \mathbf{x}_t - y_t)^2 \leq (D+1)^2. \tag{21}$$

Moreover, we have

$$\mathbf{E}\left[\left(\sum_{i \in S_t} x_{ti}\chi_i\right)^\top \left(\sum_{j \in S_t} x_{tj}\chi_j\right)\right] = \mathbf{E}\left[\sum_{i \in S_t} x_{ti}^2\right]$$
$$= \sum_{i=1}^d \mathrm{Prob}[i \in S_t] x_{ti}^2 = \frac{k-1}{m}\|\mathbf{x}_t\|^2 \leq \frac{k-1}{m}.$$

Combining the above inequality with (20) and (21), we obtain $\mathbf{E}[\|\mathbf{g}_t\|^2] \leq \frac{4(D+1)^2 d}{k-1}$. □

C More Experiments

In this section, we provide supplementary descriptions on our experiments.

Environment of experiments The experiments are performed on a server with Intel Xeon E5-2680 v3 CPUs. All algorithms are implemented in Python.

The generation procedure of synthetic datasets. For each combination of parameters (d, k', k), we generate five instances as follows. Through all the experiments, we set $m = 100$. We first create a matrix $A \in \mathbb{R}^{d \times m}$ by sampling each entry from $\mathcal{N}(0,1)$ and normalizing the matrix. We define a vector $\mathbf{u}$ by sampling each of the first k entries from $\mathcal{N}(0,1)$ and setting the others to be zero. The grand truth weight vector $\mathbf{w}_{\text{true}}$ is set to be Au. For each $t \in T$, we generate $\mathbf{x}_t$ by sampling $x_{t,i}$ $(i \in [d])$ from $\mathcal{N}(0,1)$ and set $y_t = \mathbf{w}_{\text{true}}^\top \mathbf{x}_t + z$, where z is sampled from $\mathcal{N}(0,1)$.

Preprocessing of CT-slice datasets. We deal with features that are outside of an image as ones having value zero. The sequence of the dataset is randomly shuffled in order to avoid the effects from biased sequences of images. The maximum number of positive features in one image is 165, the minimum is 9, and the average is 73. Thus we set $k = 50, k' = 40$ in this paper. For Algorithm 3 and KKLP_A, we create a matrix $A \in \mathbb{R}^{d \times m}$ with $d = 384$ and $m = 200$ in the same way as the synthetic datasets.

Results on synthetic datasets Figure 2 show typical results for some instance with $(d, k, k') = (10, 5, 8)$. We observe that our algorithms achieve small regrets at the end of iterations. Figure 3 indicates that the increase in the regrets of our algorithms is smaller than baseline algorithms for large T. Thus we can expect that our algorithms perform much better than all baseline algorithms.

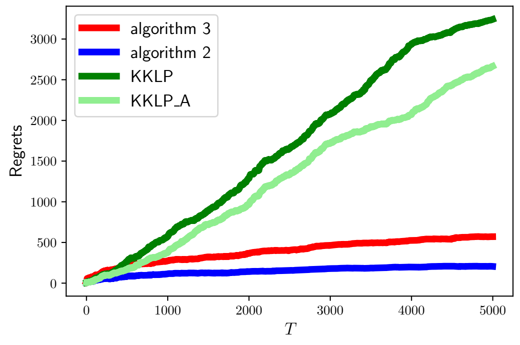


Figure 2: The regrets for a synthetic instance with $(d, k, k') = (10, 5, 8)$.

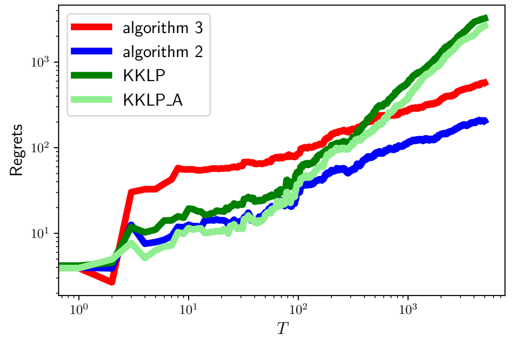


Figure 3: The log-log plots of the regrets for a synthetic instance with $(d, k, k') = (10, 5, 8)$.

We compute the averages of final regrets and execution time, respectively, for each combination of $d \in \{10, 20\}$, $k' \in \{4, 8, 12\}$ and $k' \in \{2, 5\}$. We choose parameters so that $d > k > k'$. For each combination of (d, k', k), we executed all algorithms on five instances with $T = 5000$. Tables 2 and 3 summarize the final regrets and execution time. We observe that for all cases, Algorithm 2 outperforms KKLP, and Algorithm 3 performs better than KKLP_A. The average final regret of Algorithm 2 is about ten percent of the one of KKLP. Our algorithms can process each round faster than KKLP and KKLP_A. Thus we can say that our algorithms outperform the baseline algorithms in terms of both regrets and execution time.

Table 2: Average regrets.

(d, k', k)	Algorithm 2	Algorithm 3	KKLP	KKLP_A
(10, 2, 4)	427.0	2544.3	4649.4	4335.0
(10, 2, 8)	265.2	1271.7	4365.8	3952.8
(10, 5, 8)	245.8	787.7	4370.4	4087.1
(20, 2, 4)	965.3	1553.6	5198.6	4820.4
(20, 2, 8)	527.9	1905.6	5353.1	4727.5
(20, 5, 8)	579.2	1714.6	5428.9	4929.1
(20, 2, 12)	401.2	4730.0	5304.4	4696.2
(20, 5, 12)	408.4	1047.2	5391.4	4750.1

Table 3: Average run time [s].

(d, k', k)	Algorithm 2	Algorithm 3	KKLP	KKLP_A
(10, 2, 4)	0.08	0.85	1.88	12.65
(10, 2, 8)	0.11	1.03	3.36	39.72
(10, 5, 8)	0.11	1.89	3.33	37.90
(20, 2, 4)	0.09	1.97	2.98	11.85
(20, 2, 8)	0.08	1.42	3.94	38.76
(20, 5, 8)	0.09	3.45	5.63	36.87
(20, 2, 12)	0.09	1.07	6.08	46.23
(20, 5, 12)	0.07	1.35	3.76	30.36