Refined Reduction Techniques to Online Linear Optimization

劉, 亜雄

Refined Reduction Techniques to Online Linear Optimization

Yaxiong Liu

February, 2022

Abstract

Online decision making problem plays an important role in machine learning research. Online decision making problem is represented as a repeated game between an algorithm (decision maker, player) and an environment (adversary, nature). On each single round, the algorithm is required to give a prediction firstly, and then receives the feedback given by the environment. Next, the loss is incurred with respect to the feedback and the prediction. Unlike the extensively explored statistical learning models, the feedback is not i.i.d but adversarial in online decision making framework. The exploration to the online decision making problem has been processed for years, especially to the online convex optimization framework (OCO) [26]. Online convex optimization framework is a special type of online decision making problems, where the decision set for the algorithm is convex and the feedback of the environment is the convex function. Moreover, in online convex optimization, our goal is to minimise the cumulative loss. To measure the performance of the algorithm, we involve a new concept regret, the difference between the cumulative loss and the optimal global loss in hindsight.

In this thesis, we consider some other classes of online decision making problems, which are not well-studied as the OCO framework. The basic method is that we reduce the objective problems to the online linear optimization (OLO) framework, which is a special form of online convex optimization that the feedback is a linear function.

Firstly, we consider the online load balancing problem, where the loss is not the cumulative loss but the global loss. We reduce this problem to a corresponding OLO model, where we transform the global loss to the cumulative loss and with respect to the appropriate decision set and loss space. Our proposed algorithm is efficient and achieves the best known regret bound.

Second, we explore the case of easy data for the expert advice problem, an extreme case of online linear optimization that the prediction of the algorithm is a distribution, and the online binary matrix completion (OBMC). In the former case, we concentrate on the loss sequence restricted on the noisy low rank structure. We reduce this problem to an OLO model, where the

loss sequence is arbitrary with respect to a matrix norm. Our reduction is robust to the noise compared with the previous work [29], and requires less prior information [8]. Moreover, our algorithm obtains a satisfying theoretical result and performs even better in the experiment. In the latter case, we assume that the algorithm can receive the side information before predictions. This side information can improve the prediction accuracy as shown in the work of Herbster et al. [31]. We reduce this problem to an OLO model in matrix form, called online semi-definite programming (OSDP), with mistake driven technique. We represent the side information in the form of the decision set. Combining our result to the OSDP problem, we obtain an optimal mistake bound for this OBMC problem with side information. Furthermore, we extend the existing result to the reduced OSDP framework with the variant decision set according to the side information. Actually, our algorithm is not special for the reduced OSDP problem but for the more wide situations that the algorithm is designed with respect to a general form of OSDP problems.

Conclusively, we attempt to explore some not familiar topics in online decision making problems beyond the OCO framework in this thesis. In these topics, the models are not the same as the standard OCO framework, from the cumulative loss to the global loss, from the arbitrary environment to the cooperative environment. With the reductions from these models to the OLO framework, we can analyse these problems with the existing instrument effectively and understand the problem settings more deeply. This is not only an extension of the OLO framework border or the further exploration of the online decision making framework, but also offers reliable algorithms for the practice field.

Acknowledgements

Firstly, I would like to show my gratitude to my supervisor professor Eiji Takimoto. He gave me an opportunity at the beginning of 2018 to pursue Ph.D. I learned a lot from the discussion with him not only about mathematics, and online learning but also about the correct attitude towards the academy. For instance, how to read the papers and extract the insights of the papers.

Next, I am extremely grateful for professor Kohei Hatano. He offered me a research assistant position in RIKEN AIP computational learning theory team for financial support of my work and life. Simultaneously, I am benefited from discussions with him in both academic issues and my career.

I also would like to thank Professor Jun'ichi Takeuchi, the chair of the committee, and Associate Professor Shuji Kijima, a member of the committee, gave me many valuable comments.

I am also grateful to the technical staffs of our laboratory, Ms. Sanae Wakita, Ms. Chiaki Ikechi, Ms. Akiko Ikeuchi, and a technical staff of ADS Training Unit, Ms. Yuko Hasegawa. They always supported me in many situations not only paperwork. I also express thanks to all of the staff in the Department of Informatics, Kyushu University.

Last, but not least, I really thank my family and friends for their support. I appreciate the discussion with my colleague Guangsheng Ma about mathematical proofs, and the friendship with my mate Yuchi He.

The results in this thesis were published in proceedings of ACML 2021 [39, 40], proceedings of SOFSEM 2021 [38]. I appreciate all editors, committees, anonymous referees, and publishers.

Contents

Chapter 1

Introduction

Online decision making problem is one of the most popular learning frameworks in machine learning community. Online decision making is a repeated game between two players, who select their decisions from the option sets respectively on each round. There are many sub-models from the online decision making framework, and one of the most well-known models is online convex optimization (OCO). In OCO model, on each round t, an algorithm (player one) needs to give a prediction $\boldsymbol{w}_t$ from a convex decision set W, before receiving a convex loss function $f_t \in F$ from the environment (player two) repeatedly. To measure the performance of the proposed algorithm, we involve regret, which is usually defined as the difference between the cumulative loss and the global optimal loss in hindsight. Especially, if the loss function is linear, we name the model as online linear optimization (OLO), furthermore, if the decision set is a probability simplex, we call the model expert advice.

Online linear optimization model is an important topic in the OCO framework. One of the reasons is that each OCO model can be reduced to a corresponding OLO model with the property of the convex function [48]. Thus, the research to OLO is extensive for decades(see Cesa-Bianchi and Lugosi [13], Mohri et al. [42], Hazan [26]). Same as in the OCO model, the environment in OLO model is adversarial. It implies that loss vectors are arbitrary on each round. Therefore, the proposed algorithm is supposed to be designed for the worst case guarantee. Meanwhile, the goal of the algorithm is respect to the cumulative loss.

Beyond the well-studied OLO framework with cumulative loss and the adversarial environment, the non-OCO (non-OLO) model is a quite new field in online decision making research. As we will show in this thesis, we consider firstly the different loss forms, like a global loss. Global loss is a quite different loss form from the cumulative loss. An obvious difference is that the global loss is not additive on each round t. However, the global loss is a usual mea-

surement in job scheduling and load balancing settings [3]. Naturally, we consider the online load balancing problem in our thesis. The definition of the online load balancing problem is similar to the expert advice, but the goal of the algorithm is to minimise global loss. Actually, Even-Dar et al. [19] propose an algorithm by reducing the online load balancing problem to the OLO model, and Rakhlin et al. [46] give the same regret bound as the expert advice problem but the algorithm is not efficient. Therefore, Online load balancing problem is located in the frontier of the online decision making problem research recently.

Next, we consider the online decision making models with "easy data". It implies that the environment is not adversarial but somehow "cooperative". Unlike the adversarial environment, the "easy data" setting will involve some extra parameters which can represent the "easiness" of the data. In this thesis, we introduce the environment cooperating with the algorithm in two points. In the low rank setting, we will involve the rank of the loss sequence and in the online binary matrix completion problem, we introduce the quasi-dimension with respect to the side information (See details on the next page and corresponding chapters).

In the former one, Hazan et al. [29] consider the expert advice problem receiving the structural loss sequence from the environment. Briefly, for the structural loss sequence, the dimension of the loss vector is large while the number of the factors for producing the loss vector is small. It means that the high dimensional loss vectors are from a low dimensional sub-space. Compared with the arbitrary loss sequence, the structural loss sequence is easier since all the loss vectors are impacted by few factors. Hence, with the easier loss sequence, Hazan et al. [29] offer a better regret bound than the adversarial setting. In our thesis, we assume that the structural loss sequence is noisy. It leads that the "easy data" now is not so easy since the noise. This is a natural setting in both theory and practice. There is always some noise in transmission even for the easy data. Thus, we need a robust algorithm to the noise for the structural loss sequence, further with as little as possible prior information.

In the latter one, we assume that the algorithm can receive some assistant information before prediction. This is a popular setting to OLO models recently. For instance, Dekel et al. [17] give some assistant information to improve the prediction accuracy and obtain a tighter regret bound. In this work, the algorithm can receive a hint vector before it predicts on each round t. Moreover, Bhaskara et al. [10] consider the more sophisticated case that the hint vectors might be misleading, and in the work of Bhaskara et al. [11], the regret can be tightened even if the hint is not given on each round t. We consider in this thesis the online binary matrix completion problem with side information to the algorithm. The algorithm can receive the side information

at the beginning of the learning process. In principle, the more information the algorithm can obtain, the more accurate decision the algorithm can make. Herbster et al. [31] give a regret bound for OBMC with the assistance of the side information. But the bound is not optimal.

The basic idea of our solutions to the aforementioned problems is to reduce these problems to the OLO models. This is also a common method of the work from [29, 19]. Compared with previous reductions, in this thesis, we give more refined reductions. On the one hand, our reductions are more precise, which contains more information from the original problems and are robust to the noise with less prior information. On the other hand, the reduced OLO game is not always the standard OLO model. It implies that we can not utilise the existing algorithm to our reduced OLO game directly. Like in the OBMC problem, the reduced OLO game is an extension to the standard OLO game in the decision set. Hence, it requires to design a more refined or generalised algorithm for the reduced OLO game so that the reduction can lead to a better bound.

Then, the basic definitions of the target problems are given as follows:

In the first part, we consider the online load balancing problem. In online load balancing problem, the algorithm gives a prediction $\boldsymbol{w}_t$ from an N-dimensional simplex $\Delta(N)$ and the loss vector (load vector) $\boldsymbol{l}_t$ is in an N-dimensional cube. Instead of the cumulative loss, the global loss is involved to measure the loss of each round t [19]. For makespan, a special case of online load balancing problem, the global loss in respect to the L_∞-norm and is defined as follows:

$$\max_{i\in[N]}\left\{\sum_{s=1}^{t}\boldsymbol{w}_s(1)\boldsymbol{l}_s(1),\cdots,\sum_{s=1}^{t}\boldsymbol{w}_s(j)\boldsymbol{l}_s(j),\cdots,\sum_{s=1}^{t}\boldsymbol{w}_s(N)\boldsymbol{l}_s(N)\right\}.$$

Since the difference of the global loss and the cumulative loss, we can not solve online load balancing problem directly utilising the algorithms for the OCO model (e.g., expert advice).

For makespan, Even-Dar et al. [19] achieve $O(\ln N\sqrt{T})$ bound, and Rakhlin et al. [46] gives an $O(\sqrt{T\ln N})$ bound but the algorithm is not efficient. In our thesis, we reduce the online load balancing with a general form of norms to two parallel OLO games. Furthermore, for makespan, we give an efficient algorithm that obtains the best known regret bound $O(\sqrt{T\ln N})$, by an efficient reduction from the makespan problem to two OLO games.

In the second part, we explore the expert advice framework. Expert advice is a decision making problem, where the decision set is constraint by L_1-norm (more precisely the distribution simplex) and the loss space is bounded by L_∞-norm [48]. We assume that the loss sequence

is low ranked and corrupted by a noise vector on each round. The structural loss sequence has been researched by Hazan et al. [29], Koren and Livni [35]. In their work, the loss vector is from a d-dimensional sub-space of the N-dimensional space. If $d \ll N$ we call the structure as low rank structure. For the standard expert advice problem, the loss is arbitrary when $d = N$. In the cooperative environment, $d \leq N$, especially in low rank setting. Then, Barman et al. [8] consider the noisy low rank loss, while the kernel of the loss sequence is known to the algorithm as the prior information. In all previous works, the kernel of the low rank loss plays an important role in the algorithm and the regret analysis. In [29, 35], the kernel is directly given to the algorithm or it is not difficult to be re-constructed while the loss sequence is pure (without noise). With the help of the kernel, we can reduce the decision set and the loss space of expert advice problem from L_∞-norm and its dual norm to an OLO game where the decision set and the loss space are restricted by the matrix norm with respect to the kernel. Compared with the tradition algorithm for expert advice, the Hedge algorithm achieves the regret bound $O(\sqrt{T \ln N})$, Hazan et al. [29] achieve the regret bound $\Theta(\sqrt{dT})$ if the kernel is known and $O(d\sqrt{T})$, otherwise.

In our thesis, we release the assumption for the noisy low rank loss, that the kernel is unknown to the algorithm. This is a natural setting in practice like recommendation systems, where the algorithm needs to process the data which is not perfectly transmitted but with some noise. It requires that we need to recover or approximate the low rank kernel during the process with limited steps. On the one hand, if we recover the kernel after receiving all the losses, naturally, we get the most precise kernel but make no prediction. On the other hand, if we do not recover the kernel, the reduced OLO game with matrix norm in fact reflects no advantage of the low rank property. Thus, we design an algorithm for the trade-off the exploration and exploitation of the low rank structure and robust to the noise vectors. This algorithm obtains a satisfying regret bound in theory and performs even better than Hedge algorithm and algorithm of [29] in the experiment.

In the third part, we firstly extend the setting to the decision set of the online semi-definite programming (OSDP) problem, which can be seen as an extension of the OLO model from the N-dimensional vectors to the $N \times N$ dimensional matrices [28]. In OSDP, the loss matrix is "sparse", which is equivalent to the L_1-norm bounded vector. The decision set is a set of positive definite matrices which are constraint by two norms simultaneously, the L_∞-norm to the diagonal entries and the trace norm. In our thesis, we consider a generalisation of the trace norm, the $\mathbf{\Gamma}$-trace norm. When $\mathbf{\Gamma}$ is the identity matrix, our generalisation recovers the orig-

inal setting. Due to this generalization, we utilise the follow-the-regularized-leader algorithm (FTRL) with a trace norm depending regularizer. Compared with the previous algorithm [44], our new algorithm can improve the factor about the size of the matrix N in the regret bound (see the example in the chapter 5).

Secondly, we apply our OSDP model to online binary matrix completion (OBMC)with side information [31]. OBMC is a widely applied learning framework in recommendation system and online shopping [36, 47, 18, 51, 23]. Herbster et al. [30] consider the OBMC problem and gives a mistake bound as $O\left(\frac{(m+n)\ln(m+n)}{\gamma^2}\right)$, where m, n represent the numbers of users and items respectively, and the γ is the margin complexity. Further, Herbster et al. [31] involve the side information and a new parameter, quasi-dimension $\mathcal{D}$, which can quantify the quality of the side information with respect to the users and items respectively. In adversarial settings, the side information is vacuous that the quasi-dimension is $m + n$. However, in the cooperative environment, this quasi-dimension can be set much smaller when the underlying matrix obtains some latent structure [31]. Thus, the mistake bound is improved as $O\left(\frac{\mathcal{D}\ln(m+n)}{\gamma^2}\right)$. If we set the side information as the PD-Laplacian according to the graph of users and items, then $\mathcal{D} \leq O(k+l)$, when the underlying binary matrix obtains the (k, l)-biclustered structure. However, this mistake bound is not optimal. In this thesis, we give an optimal mistake bound to OBMC with side information by reducing OBMC to the OSDP problem and transform the side information into $\mathbf{\Gamma}$. If the underlying matrix obtains (k, l)-biclustered structure, our mistake bound recovers the lower bound up to a constant factor.

1.1 Our main Contributions

In this section, we briefly describe our contribution of each problem.

1.1.1 Online load balancing problem

As we mentioned in the previous section, the global loss is firstly involved by [19]. In [19] authors propose two algorithms, the first one achieves the regret bound as $O(\ln N\sqrt{T})$, and the second one obtains $O(\sqrt{TN})$ bound by involving Blackwell approaching game [12]. But in there reduction, the reduced Blackwell game is based on the metric L_2-norm, regardless of the norm in the global cost function. [46] gives an algorithm achieving $O(\sqrt{T\ln N})$ but not efficient.

In our thesis, we further explore the relationship between the online load balancing and the Blackwell approaching game by extending the L_2-norm in [19] to the combined norm. This combined norm is in accordance with the norm in online load balancing game. It implies that even if the global cost function in online load balancing problem is with respect to L_p-norm, where $p \geq 1$, we can reduce the online load balancing problem to a more appropriate Blackwell approaching game. Next, due to the work of [50], we reduce our new Blackwell approaching game to a special OCO game, further to two parallel online linear optimization (OLO) games. Specially, if the global cost function is with respect to the L_∞-norm, by the reduction from the Blackwell game to the OLO game, we give two efficient transformations (second order cone programming and linear programming) in the reductions. Therefore, we have an efficient algorithm for the online load balancing game with L_∞-norm which obtains the best known regret bound $O(\sqrt{T \ln N})$.

1.1.2 Expert advice with noisy low rank loss

Expert advice with d rank loss has been researched by [29, 35]. If the kernel is known to the algorithm, there is a tight bound as $\Theta(\sqrt{dT})$ and $O(d\sqrt{T})$ otherwise. For the L_2-norm noisy d rank loss, where ϵ is the bound for noise vectors, [8] gives a bound $O(\sqrt{(d+\epsilon)T})$, when the kernel is known to the algorithm initially. In our thesis, we consider the case that the d rank loss is corrupted by the noise vector and the kernel is unknown to the algorithm.

The basic idea is to approximate the kernel from the received loss sequence. However, if we directly apply the algorithm in [29, 35], it leads to an $O(\sqrt{NT})$ bound, since the current loss sequence is N ranked due to the noise. The algorithm [35] requires that the loss sequence must be d ranked. The algorithm in [29] selects all the loss vectors when it belongs not to the current spanned space, and it leads the size of kernel matrix expanded to N. The size of the kernel represents the low rank structure and makes the reduction to the corresponding OLO model meaningful, otherwise, the Hedge algorithm works better than their reduction.

To ensure that the recovered or the approximated kernel performs well in the algorithm, in principle, our algorithm only selects the loss vectors satisfying the following two conditions simultaneously. The former one is that the loss vector is supposed to be long enough and the latter one is that the loss vector must be far enough to the current spanned space by the selected loss vectors. Under our selection criteria, we can select at most d loss vectors during the whole learning process and compose a pseudo-kernel on the one hand. It means that the matrix norm

of the decision vector is bounded by d. On the other hand, the distance of all the losses to this pseudo-kernel is bounded in $o(\sqrt{N})$. It implies that the dual norm of the matrix norm from the pseudo-kernel of the loss vectors is bounded by $O((N\epsilon)^{1/3})$. Therefore, we refine the reduction from the expert advice to an OLO game, where the norms to the decision set and loss space are robust to the noise, respectively. Conclusively, our algorithm can achieve a regret bound $O((d + d^{4/3}(N\epsilon)^{1/3})\sqrt{T})$ for noisy low rank loss. Although our regret bound seems to be not a large improvement, this algorithm performs well in experiment. In practice, we suggest to run our algorithm and others parallel to guarantee the best result.

1.1.3 Online semi-definite problem

Firstly, for OSDP with $\mathbf{\Gamma}$-trace norm, we propose an algorithm FTRL with generalised log-determinant regularizer. This regularizer is given according to the $\mathbf{\Gamma}$ matrix from the decision set. It implies that we need to choose different regularizer depending on the concrete problem setting. For OSDP with $\mathbf{\Gamma}$-trace norm, our proposed algorithm achieves an $O(g\sqrt{\rho\beta\tau T})$ bound, where ρ is only related to the matrix $\mathbf{\Gamma}$, and others are related to the norms for the decision set and the loss space. Compared with the previous work [44], generally, our algorithm is more appropriate for the sophisticate decision set, while in the previous algorithm, the log-determinant regularizer reflects no information about the $\mathbf{\Gamma}$-trace norm. From our example in the Chapter 5, the previous algorithm gives an $O(N\sqrt{\tau T})$ bound and our algorithm can tighten the bound to $O(\sqrt{\tau T})$, since our algorithm can process the constraint of the decision set more precisely.

Next, we show that OBMC problem with side information [31] can be reduced to our OSPD problem with $\mathbf{\Gamma}$-trace norm by mistake driven technique. Unlike the reduction of [31], our reduction explicitly utilise the hinge loss function and reduce OBMC to an OSDP problem. Thus, our reduction is more straightforward and in the reduced OSDP problem, we represent the side information with the matrix $\mathbf{\Gamma}$. Note that if we utilise the algorithm from [44], the mistake bound is $O\left(\frac{m+n}{\gamma^2}\right)$, where the side information is actually vacuous. Utilising our new achieved result for the OSDP problem with $\mathbf{\Gamma}$-trace norm, we obtain an optimal mistake bound, especially in realizable case, as $O\left(\frac{\mathcal{D}}{\gamma^2}\right)$ by improving the logarithmic factor in previous bound of [31]. Meanwhile, if the underlying matrix obtains (k, l)-biclustered, our mistake bound is $O(kl)$, where the lower bound is $\Omega(kl)$ [30]. Furthermore, in online similarity prediction, a special case of OBMC with side information [22], giving the side information as the PD-

Laplacian of the graph, our mistake bound recovers the previous mistake bound up to a constant factor and tighten a logarithmic factor as well.

1.2 Organization

The rest of this thesis is organized as follows: In Chapter 2, we give some basic learning models of online learning, like online convex optimization and expert advice. In Chapter 3, we consider the online load balancing problem, and we give an efficient algorithm achieving the best known regret bound. Next, we propose an algorithm for expert advice problem with noisy low rank loss, where the prior information about the low rank is unknown to the algorithm in Chapter 4. As last, in Chapter 5, we generalise the OSDP problem with $\mathbf{\Gamma}$-trace norm and apply our new achievement to OMBC problem with side information. For OMBC problem and further its extreme case online similarity prediction, we obtain an optimal mistake bound by reducing them to the corresponding OSDP problem.

Chapter 2

Preliminaries

In this section, we are going to introduce some basic concepts about online learning.

2.1 Online learning models

Online learning is a learning process to answer a sequence of questions. In this learning process, there are two roles: the algorithm (decision maker, player, learner) and the environment (adversary, nature). On each round t, the algorithm needs to answer the question according to previous information selected by itself before the true answer arrives. It can be described in the following chart: on each round $t \in [T] = \{1, \cdots, T\}$:

- The algorithm receives the question $x_t \in \mathcal{X}$ from the environment.
- The algorithm gives its answer $p_t \in \mathcal{P}$.
- The environment gives the true answer $y_t \in \mathcal{Y}$.
- The algorithm incurs the loss as $l(p_t, y_t)$.

We denote question domain, answering domain, and true answer domain as $\mathcal{X}, \mathcal{P}, \mathcal{Y}$ respectively. Note that $\mathcal{P}$ is not necessary same as $\mathcal{Y}$. The ultimate goal of the learner is to minimise the cumulative loss in the whole learning process. We assume that there is a target mapping $h^* : \mathcal{X} \to \mathcal{Y}$. We define the regret with respect to this target mapping h^* as follows:

$$\text{Regret}_T(h^*) = \sum_{t=1}^{T} l(p_t, y_t) - \sum_{t=1}^{T} l(h^*(x_t), y_t). \tag{2.1}$$

If we assume that $h^* \in \mathcal{H}$ for some fixed hypothesis class $\mathcal{H}$ then we can further define the regret with respect to the hypothesis class $\mathcal{H}$ as follows:

$$\text{Regret}_T(\mathcal{H}) = \max_{h^* \in \mathcal{H}} \text{Regret}_T(h^*). \tag{2.2}$$

The goal of the learner is to propose an algorithm such that the regret bound is sub-linear, i.e., $\text{Regret}_T(\mathcal{H}) \leq O(T^\alpha)$, where $\alpha < 1$.

2.1.1 Online convex optimization

In online convex optimization (OCO), we restrict the above model in a prediction process as follows: Given $\mathcal{W}, \mathcal{F}$ as the prediction domain of the algorithm and the set of convex functions respectively. On each round $t \in [T]$:

- The algorithm gives a prediction $\boldsymbol{w}_t$ from a convex set $\mathcal{W}$.
- The environment returns the a convex loss function $f_t \in \mathcal{F}$.
- The algorithm incurs the loss as $f_t(\boldsymbol{w}_t)$

Thus, the regret is now defined as

$$\text{Regret}(T) = \sum_{t=1}^{T} f_t(\boldsymbol{w}_t) - \min_{w \in \mathcal{W}} \sum_{t=1}^{T} f_t(\boldsymbol{w}). \tag{2.3}$$

Moreover, if the loss function is linear function, we call the model as online linear optimization (OLO). A typical algorithm for OCO is gradient descent [54]. We give this algorithm as follows:

Algorithm 1 Gradient Descent

Require: learning rate η, $\forall \boldsymbol{w}_1 \in \mathcal{W}$

1: For $t = 1, \cdots, T$
2: Predict $\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \nabla f_t(\boldsymbol{w}_t)$, where $\nabla f_t(\boldsymbol{w}_t)$ is the sub-gradient of f_t of point $\boldsymbol{w}_t$.
3: Receive $f_t(\cdot)$ from the environment.

Theorem 1 ([48]). *Running the gradient descent to T times, if the decision set W is bounded by $\max_{\boldsymbol{w} \in W} \|\boldsymbol{w}\|_2 \leq D$ and $\max_{\boldsymbol{w} \in W} \|\nabla f_t(\boldsymbol{w})\|_2 \leq G$ for any $t \in [T]$. Then we have the regret bound as*

$$\text{Regret}(T) \leq O(GD\sqrt{T}), \tag{2.4}$$

by setting $\eta = \frac{D}{G\sqrt{2T}}$.

2.1.2 Expert advice

Prediction with expert advice or expert advice problem is a special case of OLO. The background of this model is as follows. Given N different experts initially, on each round t, the learner is requested to choose one of the experts i_t before receiving the lost vector $\boldsymbol{l}_t \in [0,1]^N$. So the loss of each round t is $\boldsymbol{l}_t(i_t)$. To ensure that the loss is convex, we involve the expectation. The learner chooses the expert according to a distribution $\boldsymbol{w}_t \in \Delta(N)$, where

$$\Delta(N) = \left\{ \boldsymbol{w} \mid \boldsymbol{w}(i) \geq 0 \wedge \sum_{i=1}^{N} \boldsymbol{w}(i) = 1 \quad \forall i \in [N] \right\}.$$

After involving the expectations we have that

$$\mathbb{E}\left[\boldsymbol{l}_t(i_t)\right] = \sum_{i=1}^{N} \boldsymbol{w}_t(i) \cdot \boldsymbol{l}_t(i) = \boldsymbol{w}_t \cdot \boldsymbol{l}_t. \tag{2.5}$$

Thus, we can define the regret as

$$\mathrm{Regret}(T) = \sum_{t=1}^{T} \boldsymbol{l}_t \cdot \boldsymbol{w}_t - \min_{i \in [N]} \sum_{t=1}^{T} \boldsymbol{l}_t(i). \tag{2.6}$$

Note that expert advice is an OLO model with a probability simplex decision set. There is a famous algorithm for expert advice given as follows [13, 21]:

Algorithm 2 Hedge

Require: learning rate η, $\forall \boldsymbol{w}_1 \in (\frac{1}{N}, \cdots, \frac{1}{N})$

1: For $t = 1, \cdots, T$

2: Predict $\boldsymbol{w}_{t+1}(i) = \frac{\boldsymbol{w}_t(i) e^{-\eta \boldsymbol{l}_t(i)}}{\sum_{j=1}^{N} \boldsymbol{w}_t(j) e^{-\eta \boldsymbol{l}_t(j)}}$

3: Receive $\boldsymbol{l}_t$ from the environment.

Theorem 2 ([48]). *Running Hedge for T times, we obtains the regret bound for expert advice as*

$$\mathrm{Regret}(T) \leq O(\sqrt{T \ln N}), \tag{2.7}$$

by choosing particular learning rate $\eta = \sqrt{\frac{\ln N}{T}}$.

Chapter 3

Improved algorithms for online load balancing

3.1 Introduction

Online load balancing problem is an active research topic since last century. Instead of the traditional measurement of algorithm performance, competitive ratio(e.g.,[4] [5] [16] [43]), we utilize another well-known measurement as "Regret", involved by [19].

In this paper we define online load balancing problem as follows. There are K parallel servers and the protocol is defined as a game between the player and the environment. On each round $t = 1, \ldots, T$, (i) the player selects a distribution $\boldsymbol{\alpha}_t$ over K servers, which can be viewed as an allocation of data, (ii) then the environment assigns a loaded condition $l_{t,i}$ for each server i and the loss of server i is given as $\alpha_{t,i} l_{t,i}$. The goal of the player is to minimize the makespan of the cumulative loss vector of all servers after T rounds, i.e., $\max_{i=1,\ldots,K} \sum_{t=1}^{T} \alpha_{t,i} l_{t,i}$, compared relatively to the makespan obtained by the optimal static allocation $\boldsymbol{\alpha}^*$ in hindsight. More precisely, the goal is to minimize the regret, the difference between the player's makespan and the static optimal makespan. The makespan cost can be viewed as L_∞-norm of the vector of cumulative loss of each server (we will give a formal definition of the problem in the next section).

Even-Dar et al.[19] gave an algorithm based on the regret minimum framework by involving an extra concept, the Blackwell approachability [12] with respect to L_2-norm, to a target set, which is defined in the following section. This algorithm achieves the regret bound as $O(\sqrt{KT})$. Simultaneously another algorithm, DIFF, achieves the regret upper bound as $O((\ln K)\sqrt{T})$. Rahklin et al. [46] gave a theoretical result for the online load balancing prob-

lem, that the upper bound to regret can achieve $O(\sqrt{(\ln K)T})$, rather than $O((\ln K)\sqrt{T})$. However there is no efficient algorithm given in this paper to obtain this regret.

Then, there were some explorations about the equivalence between the Blackwell approachability and online linear optimization(OLO) [1], in addition and online convex optimization(OCO) by involving a support function [50].

These work [1] [50] implied that the Blackwell approachability with respect to general norm can be guaranteed by sub-linearity of regret from OCO problem reduced by Blackwell approaching game. Moreover due to this result we give an efficient algorithm to online load balancing problem, achieving the best known regret.

More specifically speaking, we propose algorithms for online load balancing for arbitrary norms under a natural assumption. This algorithm is composed by three reductions. (i) First reduction is from load balancing problem to Blackwell approaching game in a general metric. In this reduction we extend the L_2-norm of load balancing problem in [19] to any general norm with a reasonable assumption. In this reduced Blackwell approaching game the metric is induced by the norm of load balancing problem. This reduction implies that the regret of load balancing problem can be bounded by the convergence rate of a corresponding Blackwell approaching game. (ii) Second reduction directly follows the existing work. Due to [50], we give a reduction from Blackwell approaching game to an OCO problem, by showing the existence of such reduction. Thus we can bound the regret of online load balancing with the regret of corresponding OCO problem. (iii) The last reduction is from OCO problem to two OLO problems, so that we can predict with FTRL.

Conclusively, we can predict the allocation of serves on each round in online load balancing according to the prediction of corresponding two OLO problems. Simultaneously we give the regret bound of online load balancing problem with this OLO regret.

Thus our technical contributions are the following:

- We propose a new reduction technique from online load balancing to a Blackwell approaching game. This reduction enables us to use more general norms, induced by online load balancing, in Blackwell approaching game rather than L_2-norm used in the previous work [19]. Based on this reduction we can reduce online load balancing with general norm to OLO problem, by using the reduction technique of Shimkin [50] from Blackwell games to OCO problem, further to OLO problems. In conclusion, online load balancing problem can be reduced to two OLO problems according to our reduction route.

Therefore the regret bound to online load balancing problem can be optimized by the corresponding OLO regret bound.

- Especially, according to above reduction route, we give an efficient algorithm for online load balancing w.r.t. L_∞-norm, achieving the best known $O(\sqrt{T \ln K})$ regret. The algorithm involves linear programming and the second order cone programming and runs in polynomial time per trial. This is the first polynomial time algorithm achieving $O(\sqrt{T \ln K})$ regret.

3.2 Preliminaries

First we give some notations. We use $\|\cdot\|$ to denote a norm of a vector. Moreover, for a norm $\|\cdot\|$, $\|\boldsymbol{x}\|_*$ denotes the dual norm of $\|\boldsymbol{x}\|$. A norm $\|\cdot\|$ over $\mathbb{R}^d$ is monotone if $\|\boldsymbol{x}\| \leq \|\boldsymbol{y}\|$ whenever $|x_i| \leq |y_i|$ for every $1 \leq i \leq d$.

3.2.1 Online load balancing

Firstly we begin with a standard (offline) load balancing problem. Suppose that we have K servers to do a simple task with a large amount of data. The task can be easily parallelized in such a way that we can break down the data into K pieces and assign them to the servers, and then each server processes the subtask in time proportional to the size of data assigned. An example is to find blacklisted IP addresses in an access log data. Each server is associated with loaded condition, expressed in terms of "the computation time per unit data". The goal is to find a data assignment to the servers so as to equalize the computation time for all servers. In other words, we want to minimize the *makespan*, defined as the maximum of the computation time over all servers.

Formally, the problem is described as follows: The input is a K-dimensional vector $\boldsymbol{l} = (l_1, l_2, \ldots, l_K) \in \mathbb{R}_+^K$, where each l_i represents the loaded condition of the i-th server. The output is a K-dimensional probability vector $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_K) \in \Delta(K) = \{\boldsymbol{\alpha} \in [0,1]^K \mid \sum_{i=1}^K \alpha_i = 1\}$, where each α_i represents the fraction of data assigned to the i-th server. The goal is to minimize the makespan $\|\boldsymbol{\alpha} \odot \boldsymbol{l}\|_\infty$, where $\boldsymbol{\alpha} \odot \boldsymbol{l} = (\alpha_1 l_1, \alpha_2 l_2, \ldots, \alpha_K l_K)$. Note that it is clear that the optimal solution is given by $\alpha_i = l_i^{-1} / \sum_{j=1}^K l_j^{-1}$, which equalizes the computation time of every server as $C_\infty^*(\boldsymbol{l}) \stackrel{\text{def}}{=} \min_{\boldsymbol{\alpha} \in \Delta(K)} \|\boldsymbol{\alpha} \odot \boldsymbol{l}\|_\infty = \frac{1}{\sum_{j=1}^K 1/l_j}$.

Note also that the objective is generalized to the L_p-norm for any p in the literature.

In this paper, we consider a more general objective $\|\boldsymbol{\alpha} \odot \boldsymbol{l}\|$ for an arbitrary norm that satisfies certain assumptions stated below. In the general case, the optimal value is denoted by $C^*(\boldsymbol{l}) \stackrel{\text{def}}{=} \min_{\boldsymbol{\alpha} \in \Delta(K)} \|\boldsymbol{\alpha} \odot \boldsymbol{l}\|$.

Assumption 1. *Throughout the paper, we put the following assumptions on the norm. (i) The norm is monotone, and (ii) the function C^* is concave.*

Note that the first assumption is natural for load balancing and the both assumptions are satisfied by L_p-norm for $p > 1$.

Now we proceed to the online load balancing problem with respect to a norm $\|\cdot\|$ that satisfies Assumption 1. The problem is described as a repeated game between the learner and the environment who may behave adversarially. In each round $t = 1, 2, \ldots, T$, the learner chooses an assignment vector $\boldsymbol{\alpha}_t \in \Delta(K)$, and then receives from the environment a loaded condition vector $\boldsymbol{l}_t \in [0, 1]^K$, which may vary from round to round. After the final round is over, the performance of the learner is naturally measured by $\left\|\sum_{t=1}^T \boldsymbol{\alpha}_t \odot \boldsymbol{l}_t\right\|$. We want to make the learner perform nearly as well as the performance of the best fixed assignment in hindsight (offline optimal solution), which is given by $C^*(\sum_{t=1}^T \boldsymbol{l}_t)$. To be more specific, the goal is to minimize the following *regret*:

$$\text{Regret}(T) = \left\|\sum_{t=1}^T \boldsymbol{\alpha}_t \odot \boldsymbol{l}_t\right\| - C^*\left(\sum_{t=1}^T \boldsymbol{l}_t\right).$$

3.2.2 Repeated game with vector payoffs and approachability

We briefly review the notion of Blackwell's approachability, which is defined for a repeated game with vector payoffs. The game is specified by a tuple $(A, B, r, S, \text{dist})$, where A and B are convex and compact sets, $r : A \times B \to \mathbb{R}^d$ is a vector-valued payoff function, $S \subseteq \mathbb{R}^d$ is a convex and closed set called the *target set*, and $\text{dist} : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}_+$ is a metric. The protocol proceeds in trials: In each round $t = 1, 2, \ldots, T$, the learner chooses a vector $\boldsymbol{a}_t \in A$, the environment chooses a vector $\boldsymbol{b}_t \in B$, and then the learner obtains a vector payoff $\boldsymbol{r}_t \in \mathbb{R}^d$, given by $\boldsymbol{r}_t = r(\boldsymbol{a}_t, \boldsymbol{b}_t)$. The goal of the learner is to make the average payoff vector arbitrarily close to the target set S.

Definition 1 (Approachability)**.** *For a game $(A, B, r, S, \text{dist})$, the target set S is approachable with convergence rate $\gamma(T)$ if there exists an algorithm for the learner such that the average*

payoff $\bar{\boldsymbol{r}}_T = (1/T)\sum_{t=1}^{T} \boldsymbol{r}_t$ *satisfies*

$$\mathrm{dist}(\bar{\boldsymbol{r}}_T, S) \stackrel{\mathrm{def}}{=} \min_{\boldsymbol{s}\in S} \mathrm{dist}(\bar{\boldsymbol{r}}_T, \boldsymbol{s}) \le \gamma(T)$$

against any environment. In particular, we simply say that S *is approachable if it is approachable with convergence rate* $o(T)$.

Blackwell characterizes the approachability in terms of the support function as stated in the proposition below.

Definition 2. *For a set* $S \subseteq \mathbb{R}^d$*, the support function* $h_S : \mathbb{R}^d \to \mathbb{R} \cup \{\infty\}$ *is defined as*

$$h_S(\boldsymbol{w}) = \sup_{\boldsymbol{s}\in S} \langle \boldsymbol{s}, \boldsymbol{w}\rangle.$$

It is clear from definition that h_S is convex whenever S is convex.

Definition 3 (Blackwell [12])**.** *A game* $(A, B, r, S, \mathrm{dist})$ *satisfies Blackwell Condition, if and only if*

$$\forall \boldsymbol{w} \in \mathbb{R}^d \ \left(\min_{\boldsymbol{a}\in A} \max_{\boldsymbol{b}\in B} \langle \boldsymbol{w}, r(\boldsymbol{a}, \boldsymbol{b})\rangle \le h_S(\boldsymbol{w}) \right). \tag{3.1}$$

Remark 1. *In [12], Blackwell characterized the approachability of a target set for* L_2*-norm metric in terms of the Blackwell condition.*

In what follows, we only consider a norm metric, i.e, $\mathrm{dist}(\boldsymbol{r}, \boldsymbol{s}) = \|\boldsymbol{r} - \boldsymbol{s}\|$ for some norm $\|\cdot\|$ over $\mathbb{R}^d$. The following proposition is useful.

Proposition 1. *For any* $\boldsymbol{w} \in \mathbb{R}^d$*,* $\boldsymbol{s}^* = \arg\max_{\boldsymbol{s}\in S}\langle \boldsymbol{s}, \boldsymbol{w}\rangle$ *is a sub-gradient of* $h_S(\boldsymbol{w})$ *at* $\boldsymbol{w}$*.*

Proof. For any $\boldsymbol{w}, \boldsymbol{u} \in \mathbb{R}^d$, let $\boldsymbol{s}^* = \arg\max_{\boldsymbol{s}\in S}\langle \boldsymbol{s}, \boldsymbol{w}\rangle$ and $\boldsymbol{s}^{\boldsymbol{u}} = \arg\max_{\boldsymbol{s}\in S}\langle \boldsymbol{s}, \boldsymbol{u}\rangle$. Since $\langle \boldsymbol{s}^*, \boldsymbol{u}\rangle \le \langle \boldsymbol{s}^{\boldsymbol{u}}, \boldsymbol{u}\rangle$, we have

$$\begin{aligned} h_S(\boldsymbol{w}) - h_S(\boldsymbol{u}) &= \sup_{\boldsymbol{s}\in S}\langle \boldsymbol{s}, \boldsymbol{w}\rangle - \sup_{\boldsymbol{s}\in S}\langle \boldsymbol{s}, \boldsymbol{u}\rangle = \langle \boldsymbol{s}^*, \boldsymbol{w}\rangle - \langle \boldsymbol{s}^{\boldsymbol{u}}, \boldsymbol{u}\rangle \\ &\le \langle \boldsymbol{s}^*, \boldsymbol{w} - \boldsymbol{u}\rangle, \end{aligned}$$

which implies the proposition.

□

3.2.3 Online convex optimization

In this subsection we briefly review online convex optimization with some known results. See, e.g., [48, 27] for more details.

An online convex optimization (OCO) problem is specified by (W, F), where $W \subseteq \mathbb{R}^d$ is a compact convex set called the decision set and $F \subseteq \{f : W \to \mathbb{R}\}$ is a set of convex functions over W called the loss function set. The OCO problem (W, F) is described by the following protocol between the learner and the adversarial environment. For each round $t = 1, 2, \ldots, T$, the learner chooses a decision vector $\boldsymbol{w}_t \in W$ and then receives from the environment a loss function $f_t \in F$. In this round, the learner incurs the loss given by $f_t(\boldsymbol{w}_t)$. The goal is to make the cumulative loss of the learner nearly as small as the cumulative loss of the best fixed decision. To be more specific, the goal is to minimize the following regret:

$$\mathrm{Regret}_{(W,F)}(T) = \sum_{t=1}^{T} f_t(\boldsymbol{w}_t) - \min_{\boldsymbol{w} \in W} \sum_{t=1}^{T} f_t(\boldsymbol{w}).$$

Here we add the subscript (W, F) to distinguish from the regret for online load balancing.

Any OCO problem can be reduced to an online linear optimization (OLO) problem, which is an OCO problem with linear loss functions. More precisely, an OLO problem is specified by (W, G), where $G \subseteq \mathbb{R}^d$ is the set of cost vectors such that the loss function at round t is $\langle \boldsymbol{g}_t, \cdot \rangle$ for some cost vector $\boldsymbol{g}_t \in G$. For the OLO problem (W, G), the regret of the learner is thus given by

$$\mathrm{Regret}_{(W,G)}(T) = \sum_{t=1}^{T} \langle \boldsymbol{g}_t, \boldsymbol{w}_t \rangle - \min_{\boldsymbol{w} \in W} \sum_{t=1}^{T} \langle \boldsymbol{g}_t, \boldsymbol{w} \rangle.$$

The reduction from OCO to OLO is simple. Run any algorithm for OLO (W, G) with $\boldsymbol{g}_t \in \partial f_t(\boldsymbol{w}_t)$, and then it achieves $\mathrm{Regret}_{(W,F)}(T) \leq \mathrm{Regret}_{(W,G)}(T)$, provided that G is large enough, i.e., $G \supseteq \bigcup_{f \in F, \boldsymbol{w} \in W} \partial f(\boldsymbol{w})$.

A standard FTRL (follow-the-regularized-leader) strategy for the OLO problem (W, G) is to choose $\boldsymbol{w}_t$ as

$$\boldsymbol{w}_t = \arg\min_{\boldsymbol{w} \in W} \left(\sum_{s=1}^{t-1} \langle \boldsymbol{g}_s, \boldsymbol{w} \rangle + \eta_t R(\boldsymbol{w}) \right), \tag{3.2}$$

where $R : W \to \mathbb{R}$ is a strongly convex function called the regularizer and $\eta_t \in \mathbb{R}_+$ is a parameter. Using the strategy (3.2) the following regret bound is known.

Proposition 2 ([48])**.** *Suppose that the regularizer $R : W \to \mathbb{R}$ is σ-strongly convex w.r.t. some*

norm $\|\cdot\|$*, i.e., for any* $\boldsymbol{w}, \boldsymbol{u} \in W$*, for any* $\boldsymbol{z} \in \partial R(\boldsymbol{w})$*,* $R(\boldsymbol{u}) \geq R(\boldsymbol{w}) + \langle \boldsymbol{z}, \boldsymbol{u} - \boldsymbol{w} \rangle + \frac{\sigma}{2}\|\boldsymbol{u} - \boldsymbol{w}\|^2$*. Then, for the OLO problem* (W, G)*, the regret of the strategy (3.2) satisfies*

$$\mathrm{Regret}_{(W,G)}(T) = O(D_R L_G \sqrt{T/\sigma}),$$

where $D_R = \sqrt{\max_{\boldsymbol{w} \in W} R(\boldsymbol{w})}$*,* $L_G = \max_{\boldsymbol{g} \in G} \|\boldsymbol{g}\|_*$ *and* $\eta_t = (L_G/D_R)\sqrt{T/\sigma}$*.*

Note however that the strategy does not consider the computational feasibility at all. For efficient reduction, we need an efficient algorithm that computes a sub-gradient $\boldsymbol{g} \in \partial f(\boldsymbol{w})$ when given (a representation of) $f \in F$ and $w \in W$, and an efficient algorithm for solving the convex optimization problem (3.2).

For a particular OLO problem (W, G) with L_1 ball decision set $W = \{\boldsymbol{w} \in \mathbb{R}^d \mid \|\boldsymbol{w}\|_1 \leq 1\}$, an algorithm called EG$^{\pm}$ [32] finds in linear time the optimal solution of (3.2) with an entropic regularizer and achieves the following regret.

Theorem 3 ([33])**.** *For the OLO problem* (W, G) *with* $W = \{\boldsymbol{w} \in \mathbb{R}^d \mid \|\boldsymbol{w}\|_1 \leq 1\}$ *and* $G = \{\boldsymbol{g} \in \mathbb{R}^d \mid \|\boldsymbol{g}\|_\infty \leq M\}$*, EG*$^{\pm}$ *achieves*

$$\mathrm{Regret}_{(W,G)}(T) \leq M\sqrt{2T\ln(2d)}.$$

3.3 Main result

In this section, we propose a meta-algorithm for online load balancing, which is obtained by combining a reduction to two independent OLO problems and an OLO algorithm (as an oracle) for the reduced problems. Note that the reduced OLO problems depend on the choice of norm for online load balancing, and the OLO problems are further reduced to some optimization problems defined in terms of the norm. For efficient implementation, we assume that the optimization problems are efficiently solved.

Now we consider the online load balancing problem on K servers with respect to a norm $\|\cdot\|$ defined over $\mathbb{R}^K$ that satisfies Assumption 1. The reduction we show consists of three reductions, the first reduction is to a repeated game with vector payoffs, the second one is to an OCO problem, and the last one is to two OLO problems. In the subsequent subsections, we give these reductions, respectively.

3.3.1 Reduction to a vector payoff game

We will show that the online load balancing problem can be reduced to the following repeated game with vector payoffs, denoted by $P = (A, B, r, S, \mathrm{dist})$, where

- $A = \Delta(K), \quad B = [0,1]^K$,
- $r : A \times B \to \mathbb{R}^K \times \mathbb{R}^K$ is the payoff function defined as $r(\boldsymbol{\alpha}, \boldsymbol{l}) = (\boldsymbol{\alpha} \odot \boldsymbol{l}, \boldsymbol{l})$,
- $S = \{(\boldsymbol{x}, \boldsymbol{y}) \in [0,1]^K \times [0,1]^K \mid \|\boldsymbol{x}\| \leq C^*(\boldsymbol{y})\}$, and
- dist is the metric over $\mathbb{R}^K \times \mathbb{R}^K$ defined as $\mathrm{dist}(\boldsymbol{r}, \boldsymbol{s}) = \|\boldsymbol{r} - \boldsymbol{s}\|^+$, where $\|\cdot\|^+$ is the norm over $\mathbb{R}^K \times \mathbb{R}^K$ defined as

$$\|(\boldsymbol{x}, \boldsymbol{y})\|^+ = \|\boldsymbol{x}\| + \|\boldsymbol{y}\| .$$

Here we use the convention that $\mathbb{R}^{2K} = \mathbb{R}^K \times \mathbb{R}^K$. Note that the target set S is convex since $\|\cdot\|$ is convex and C^* is concave by our assumption. Note also that it is easy to verify that $\|\cdot\|^+$ is a norm whenever $\|\cdot\|$ is a norm, and its dual is

$$\|(\boldsymbol{x}, \boldsymbol{y})\|_*^+ = \max\{\|\boldsymbol{x}\|_* , \|\boldsymbol{y}\|_*\}. \tag{3.3}$$

The reduction is similar to that in [19], but they consider a fixed norm $\|\cdot\|_2$ to define the metric, no matter what norm is used for online load balancing.

Proposition 3. *Assume that we have an algorithm for the repeated game P that achieves convergence rate $\gamma(T)$. Then, the algorithm, when directly applied to the online load balancing problem, achieves*

$$\mathrm{Regret}(T) \leq T\gamma(T).$$

Proof. Let $\mathcal{A}$ denote an algorithm for the repeated game P with convergence rate $\gamma(T)$. Assume that when running $\mathcal{A}$ against the environment of online load balancing, we observe, in each round t, $\alpha_t \in \Delta(K)$ output from $\mathcal{A}$ and $\boldsymbol{l}_t \in [0,1]^K$ output from the environment.

Let $(\boldsymbol{x}, \boldsymbol{y}) = \arg\min_{(\boldsymbol{x},\boldsymbol{y}) \in S} \|\bar{r}_T - (\boldsymbol{x}, \boldsymbol{y})\|^+$, where $\bar{r}_T = (1/T) \sum_{t=1}^T r(\boldsymbol{\alpha}_t, \boldsymbol{l}_t)$ is the average payoff. Note that by the assumption of $\mathcal{A}$, we have $\|\bar{r}_T - (\boldsymbol{x}, \boldsymbol{y})\|^+ \leq \gamma(T)$. For simplicity, let $L_T^{\mathcal{A}} = (1/T) \sum_{t=1}^T \boldsymbol{\alpha}_t \odot \boldsymbol{l}_t$ and $L_T = (1/T) \sum_{t=1}^T \boldsymbol{l}_t$.

Then, we have

$$\begin{aligned}
(1/T)\mathrm{Regret}(T) &= \left\|L_T^{\mathcal{A}}\right\| - C^*(L_T) \\
&= \left[\|\boldsymbol{x}\| - C^*(\boldsymbol{y})\right] + \left[\left\|L_T^{\mathcal{A}}\right\| - \|\boldsymbol{x}\|\right] + \left[C^*(\boldsymbol{y}) - C^*(L_T)\right] \\
&\le \left\|L_T^{\mathcal{A}} - \boldsymbol{x}\right\| + \left[\min_{\boldsymbol{\alpha}\in\Delta(K)} \|\boldsymbol{\alpha}\odot\boldsymbol{y}\| - \min_{\boldsymbol{\alpha}\in\Delta(K)} \|\boldsymbol{\alpha}\odot L_T\|\right] \\
&\le \left\|L_T^{\mathcal{A}} - \boldsymbol{x}\right\| + \max_{\boldsymbol{\alpha}\in\Delta(K)} \left[\|\boldsymbol{\alpha}\odot\boldsymbol{y}\| - \|\boldsymbol{\alpha}\odot L_T\|\right] \\
&\le \left\|L_T^{\mathcal{A}} - \boldsymbol{x}\right\| + \max_{\boldsymbol{\alpha}\in\Delta(K)} \|\boldsymbol{\alpha}\odot(\boldsymbol{y} - L_T)\| \\
&\le \left\|L_T^{\mathcal{A}} - \boldsymbol{x}\right\| + \|\boldsymbol{y} - L_T\| \\
&= \left\|(L_T^{\mathcal{A}}, L_T) - (\boldsymbol{x},\boldsymbol{y})\right\|^+ = \|\bar{r}_T - (\boldsymbol{x},\boldsymbol{y})\|^+ \le \gamma(T),
\end{aligned}$$

where the first inequality is from the definition of S and the triangle inequality, the third inequality is from the triangle inequality, and the fourth inequality is from the monotonicity of the norm. □

3.3.2 Reduction to an OCO problem

Next we give the second sub-reduction from the repeated game P to an OCO problem. We just follow a general reduction technique of Shimkin [50] as given in the next theorem.

Theorem 4 ([50]). *Let $(A, B, r, S, \mathrm{dist})$ be a repeated game with vector payoffs, where $\mathrm{dist}(\boldsymbol{r},\boldsymbol{s}) = \|\boldsymbol{r}-\boldsymbol{s}\|$ for some norm $\|\cdot\|$ over $\mathbb{R}^d$. Assume that we have an algorithm $\mathcal{A}$ that witnesses the Blackwell condition, i.e., when given $\boldsymbol{w}\in\mathbb{R}^d$, $\mathcal{A}$ finds $\boldsymbol{a}\in A$ such that $\langle \boldsymbol{w}, r(\boldsymbol{a},\boldsymbol{b})\rangle \le h_S(\boldsymbol{w})$ for any $\boldsymbol{b}\in B$. Assume further that we have an algorithm $\mathcal{B}$ for the OCO problem (W,F), where $W = \{\boldsymbol{w}\in\mathbb{R}^d \mid \|\boldsymbol{w}\|_* \le 1\}$ and $F = \{f : \boldsymbol{w}\mapsto \langle -r(\boldsymbol{a},\boldsymbol{b}),\boldsymbol{w}\rangle + h_S(\boldsymbol{w}) \mid \boldsymbol{a}\in A, \boldsymbol{b}\in B\}$. Then, we can construct an algorithm for the repeated game such that its convergence rate $\gamma(T)$ satisfies*

$$\gamma(T) \le \frac{\mathrm{Regret}_{(W,F)}(T)}{T}.$$

Moreover, the algorithm runs in polynomial time (per round) if $\mathcal{A}$ and $\mathcal{B}$ are polynomial time algorithms.

For completeness, the reduction algorithm(Algorithm 3) is as follow.

The rest to show in this subsection is to ensure the existence of algorithm $\mathcal{A}$ required for the reduction as stated in the theorem above. In other words, we show that the Blackwell condition

Algorithm 3 Reduction from game $(A, B, r, S, \mathrm{dist})$ with $\mathrm{dist}(\boldsymbol{a}, \boldsymbol{b}) = \|\boldsymbol{a} - \boldsymbol{b}\|$ to OCO [50]

Require: An algorithm $\mathcal{A}$ that, when given $\boldsymbol{w}$, finds $\boldsymbol{a} \in A$ such that $\langle \boldsymbol{w}, r(\boldsymbol{a}, \boldsymbol{b})\rangle \leq h_S(\boldsymbol{w})$ for any $\boldsymbol{b} \in B$.

Require: An algorithm $\mathcal{B}$ for the OCO problem (W, F), where $W = \{\boldsymbol{w} \mid \|\boldsymbol{w}\|_* \leq 1\}$ and $F = \{f : \boldsymbol{w} \mapsto \langle -r(\boldsymbol{a}, \boldsymbol{b}), \boldsymbol{w}\rangle + h_S(\boldsymbol{w}) \mid \boldsymbol{a} \in A, \boldsymbol{b} \in B\}$.

for $t = 1, 2, \ldots, T$ **do**

1. Obtain $\boldsymbol{w}_t \in W$ from $\mathcal{B}$.
2. Run $\mathcal{A}(\boldsymbol{w}_t)$ and obtain $\boldsymbol{a}_t \in A$.
3. Output $\boldsymbol{a}_t \in A$ and observe $\boldsymbol{b}_t \in B$.
4. Construct the loss function $f_t : \boldsymbol{w} \mapsto \langle -r(\boldsymbol{a}_t, \boldsymbol{b}_t), \boldsymbol{w}\rangle + h_S(\boldsymbol{w})$ and feed it to $\mathcal{B}$.

end for

holds for our game $P = (\Delta(K), [0,1]^K, r, S, \mathrm{dist})$, where $r(\boldsymbol{\alpha}, \boldsymbol{l}) = (\boldsymbol{\alpha} \odot, \boldsymbol{l}, \boldsymbol{l}) \in \mathbb{R}^K \times \mathbb{R}^K$, $S = \{(\boldsymbol{x}, \boldsymbol{y}) \in [0,1]^K \times [0,1]^K \mid \|\boldsymbol{x}\| \leq C^*(\boldsymbol{y})\}$, and $\mathrm{dist}(\boldsymbol{r}, \boldsymbol{s}) = \|\boldsymbol{r} - \boldsymbol{s}\|^+$.

Lemma 1. *The Blackwell condition holds for game P. That is, for any $\boldsymbol{w} \in \mathbb{R}^K \times \mathbb{R}^K$, we have*

$$\min_{\boldsymbol{\alpha} \in \Delta(K)} \max_{\boldsymbol{l} \in [0,1]^K} \langle \boldsymbol{w}, r(\boldsymbol{\alpha}, \boldsymbol{l})\rangle \leq h_S(\boldsymbol{w}).$$

Proof of Lemma 1. Let $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2) \in \mathbb{R}^K \times \mathbb{R}^K$. By the definition of r, the inner product in the Blackwell condition can be rewritten as a bilinear function

$$f(\boldsymbol{\alpha}, \boldsymbol{l}) = \langle \boldsymbol{w}, r(\boldsymbol{\alpha}, \boldsymbol{l})\rangle = \sum_{i=1}^{K} w_{1,i} \alpha_i l_i + \sum_{i=1}^{K} w_{2,i} l_i$$

over $\Delta(K) \times [0,1]^K$. Therefore, f meets the condition of Minimax Theorem of von Neumann. and we have

$$\min_{\boldsymbol{\alpha} \in \Delta(K)} \max_{\boldsymbol{l} \in [0,1]^K} f(\boldsymbol{\alpha}, \boldsymbol{l}) = \max_{\boldsymbol{l} \in [0,1]^K} \min_{\boldsymbol{\alpha} \in \Delta(K)} f(\boldsymbol{\alpha}, \boldsymbol{l}).$$

Let $\boldsymbol{l}^* = \arg\max_{\boldsymbol{l} \in [0,1]^K} \min_{\boldsymbol{\alpha} \in \Delta(K)} f(\boldsymbol{\alpha}, \boldsymbol{l})$ and $\boldsymbol{\alpha}^* = \arg\min_{\boldsymbol{\alpha} \in \Delta(K)} \|\boldsymbol{\alpha} \odot \boldsymbol{l}^*\|$. Note that by the definition of S, we have $(\boldsymbol{\alpha}^* \odot \boldsymbol{l}^*, \boldsymbol{l}^*) \in S$. Hence we get

$$\begin{aligned}
\min_{\boldsymbol{\alpha} \in \Delta(K)} \max_{\boldsymbol{l} \in [0,1]^K} f(\boldsymbol{\alpha}, \boldsymbol{l}) &= \max_{\boldsymbol{l} \in [0,1]^K} \min_{\boldsymbol{\alpha} \in \Delta(K)} f(\boldsymbol{\alpha}, \boldsymbol{l}) \\
&= f(\boldsymbol{\alpha}^*, \boldsymbol{l}^*) \\
&= \langle \boldsymbol{w}, ((\boldsymbol{\alpha}^* \odot \boldsymbol{l}^*), \boldsymbol{l}^*)\rangle \\
&\leq \sup_{\boldsymbol{s} \in S} \langle \boldsymbol{w}, \boldsymbol{s}\rangle \\
&= h_S(\boldsymbol{w}),
\end{aligned}$$

which completes the lemma. □

The lemma ensures the existence of algorithm $\mathcal{A}$. On the other hand, for an algorithm $\mathcal{B}$ we need to consider the OCO problem (W, F), where the decision set is

$$W = \{\boldsymbol{w} \in \mathbb{R}^K \times \mathbb{R}^K \mid \|\boldsymbol{w}\|_*^+ \leq 1\}, \tag{3.4}$$

and the loss function set is

$$F = \{f : \boldsymbol{w} \mapsto \langle -r(\boldsymbol{\alpha}, \boldsymbol{l}), \boldsymbol{w}\rangle + h_S(\boldsymbol{w}) \mid \boldsymbol{\alpha} \in \Delta(K), \boldsymbol{l} \in [0,1]^K\}. \tag{3.5}$$

Since W is a compact and convex set and F consists of convex functions, we could apply a number of existing OCO algorithms to obtain $\mathrm{Regret}_{(W,F)}(T) = O(\sqrt{T})$. In the next subsection, we show that the problem can be simplified to two OLO problems.

3.3.3 Reduction to two OLO problems

Consider the OCO problem (W, F) given by (3.4) and (3.5). Following the standard reduction technique from OCO to OLO stated in Section 3.2.3, we obtain an OLO problem (W, G) to cope with, where $G \subseteq \mathbb{R}^K \times \mathbb{R}^K$ is any set of cost vectors that satisfies

$$G \supseteq \bigcup_{\substack{f \in F \\ \boldsymbol{w} \in W}} \partial f(\boldsymbol{w}) = \left\{ -r(\boldsymbol{\alpha}, \boldsymbol{l}) + \boldsymbol{s} \;\middle|\; \boldsymbol{\alpha} \in \Delta(K), \boldsymbol{l} \in [0,1]^K, \boldsymbol{s} \in \textstyle\bigcup_{\boldsymbol{w} \in W} \partial h_S(\boldsymbol{w}) \right\}. \tag{3.6}$$

By (3.3), the decision set W can be rewritten as $W = B_*(K) \times B_*(K)$ where $B_*(K) = \{\boldsymbol{w} \in \mathbb{R}^K \mid \|\boldsymbol{w}\|_* \leq 1\}$ is the K-dimensional unit ball with respect to the dual norm $\|\cdot\|_*$. By Proposition 1, any $\boldsymbol{s} \in \partial h_S(\boldsymbol{w})$ is in the target set S, which is a subset of $[0,1]^K \times [0,1]^K$. Moreover, $r(\boldsymbol{\alpha}, \boldsymbol{l}) = (\boldsymbol{\alpha} \odot \boldsymbol{l}, \boldsymbol{l}) \in [0,1]^K \times [0,1]^K$ for any $\boldsymbol{\alpha} \in \Delta(K)$ and $\boldsymbol{l} \in [0,1]^K$. Therefore, $G = [-1,1]^K \times [-1,1]^K$ satisfies (3.6).

Thus, $(B_*(K) \times B_*(K), [-1,1]^K \times [-1,1]^K)$ is a suitable OLO problem reduced from the OCO problem (W, F). Furthermore, we can break the OLO problem into two independent OLO problems $(B_*(K), [-1,1]^K)$ in the straightforward way: Make two copies of an OLO algorithm $\mathcal{C}$ for $(B_*(K), [-1,1]^K)$, denoted by $\mathcal{C}_1$ and $\mathcal{C}_2$, and use them for predicting the first half and second half decision vectors, respectively. More precisely, for each trial t, (1) receive predictions $\boldsymbol{w}_{t,1} \in B_*(K)$ and $\boldsymbol{w}_{t,2} \in B_*(K)$ from $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, (2) output their concatenation $\boldsymbol{w}_t = (\boldsymbol{w}_{t,1}, \boldsymbol{w}_{t,2}) \in W$, (3) receive a cost vector $\boldsymbol{g}_t = (\boldsymbol{g}_{t,1}, \boldsymbol{g}_{t,2}) \in [-1,1]^K \times$

$[-1,1]^K$ from the environment, (4) feed $\boldsymbol{g}_{t,1}$ and $\boldsymbol{g}_{t,2}$ to $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, to make them proceed.

It is clear that the procedure above ensures the following lemma.

Lemma 2. *The OCO problem (W, F) defined as (3.4) and (3.5) can be reduced to the OLO problem $(B_*(K), [-1,1]^K)$, and*

$$\mathrm{Regret}_{(W,F)}(T) \le 2\mathrm{Regret}_{(B_*(K),[-1,1]^K)}(T).$$

3.3.4 Putting all the pieces together

Combining all reductions stated in the previous subsections, we get an all-in-one algorithm as described in Algorithm 4.

Algorithm 4 An OLO-based online load balancing algorithm

Require: An algorithm $\mathcal{A}$ that, when given $\boldsymbol{w}$, finds $\boldsymbol{\alpha} = \arg\min_{\boldsymbol{\alpha}\in\Delta(K)} \max_{\boldsymbol{l}\in[0,1]^K} \langle \boldsymbol{w}, (\boldsymbol{\alpha}\odot\boldsymbol{l}, \boldsymbol{l})\rangle$.
Require: An algorithm $\mathcal{B}$ that, when given $\boldsymbol{w}$, finds $\boldsymbol{s} \in \partial h_S(\boldsymbol{w})$.
Require: Two copies of an algorithm, $\mathcal{C}_1$ and $\mathcal{C}_2$, for the OLO problem $(B_*(K), [-1,1]^K)$.
for $t = 1, 2, \ldots, T$ **do**
1. Obtain $\boldsymbol{w}_{t,1}$ and $\boldsymbol{w}_{t,2}$ from $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively, and let $\boldsymbol{w}_t = (\boldsymbol{w}_{t,1}, \boldsymbol{w}_{t,2})$.
2. Run $\mathcal{A}(\boldsymbol{w}_t)$ and obtain $\boldsymbol{\alpha}_t \in \Delta(K)$.
3. Output $\boldsymbol{\alpha}_t$ and observe $\boldsymbol{l}_t \in [0,1]^K$.
4. Run $\mathcal{B}(\boldsymbol{w}_t)$ and obtain $\boldsymbol{s}_t = (\boldsymbol{s}_{t,1}, \boldsymbol{s}_{t,2})$.
5. Let $\boldsymbol{g}_{t,1} = -\boldsymbol{\alpha}_t \odot \boldsymbol{l}_t + \boldsymbol{s}_{t,1}$ and $\boldsymbol{g}_{t,2} = -\boldsymbol{l}_t + \boldsymbol{s}_{t,2}$.
6. Feed $\boldsymbol{g}_{t,1}$ and $\boldsymbol{g}_{t,2}$ to $\mathcal{C}_1$ and $\mathcal{C}_2$, respectively.

end for

It is clear that combining Proposition 3, Theorem 4 and Lemma 2, we get the following regret bound of Algorithm 4.

Theorem 5. *Algorithm 4 achieves*

$$\mathrm{Regret}(T) \le 2\mathrm{Regret}_{(B_*(K),[-1,1]^K)}(T),$$

where the regret in the right hand side is the regret of algorithm $\mathcal{C}_1$ (and $\mathcal{C}_2$ as well). Moreover, if $\mathcal{A}$, $\mathcal{B}$ and $\mathcal{C}_1$ runs in polynomial time, then Algorithm 4 runs in polynomial time (per round).

By applying the FTRL as in (3.2) to the OLO problem $(B_*(K), [-1,1]^K)$ with a strongly convex regularizer R, Proposition 2 implies the following regret bound.

Corollary 1. *Assume that there exists a regularizer $R : B_*(K) \to \mathbb{R}$ that is σ-strongly convex w.r.t. L_1-norm. Then, there exists an algorithm for the online load balancing problem that achieves*

$$\mathrm{Regret}(T) = O(D_R\sqrt{T/\sigma}),$$

where $D_R = \sqrt{\max_{\boldsymbol{w}\in B_*(K)} R(\boldsymbol{w})}$.

In particular, for the OLO problem $(B_1(K), [-1,1]^K)$, algorithm EG$^{\pm}$ achieves $\sqrt{2T\ln 4K}$ regret bound as shown in Theorem 3. Thus we have $O(\sqrt{T\ln K})$ regret bound for online load balancing with respect to L_∞-norm (i.e., w.r.t. makespan), which improves the bound of [19] by a factor of $\sqrt{\ln K}$. Moreover, for L_∞-norm, it turns out that we have polynomial time algorithms for $\mathcal{A}$ and $\mathcal{B}$, which we will give in the next section. We thus obtain the following corollary.

Corollary 2. *There exists a polynomial time (per round) algorithm for the online load balancing problem with respect to L_∞-norm that achieves*

$$\mathrm{Regret}(T) \le 2\sqrt{2T\ln 4K}.$$

3.4 Algorithmic details for L_∞-norm

In this section we give details of Algorithm 4 for the makespan problem, i.e., for L_∞-norm.

3.4.1 Computing α_t

First, we give details of implementation of $\mathcal{A}$ in Algorithm 4. Specifically, on the round t, we need to choose $\boldsymbol{\alpha}_t$, which is the optimal solution of the problem in Lemma 1. That is,

$$\min_{\boldsymbol{\alpha}\in\Delta(K)} \max_{\boldsymbol{l}\in[0,1]^K} \langle \boldsymbol{w}_1, (\boldsymbol{\alpha}\odot\boldsymbol{l})\rangle + \langle \boldsymbol{w}_2, \boldsymbol{l}\rangle, \tag{3.7}$$

where we set that $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2)$ and $\boldsymbol{w}_1$ and $\boldsymbol{w}_2$ are K-dimensional vectors, respectively. We see that the optimization of this objective function is defined by $l_i = 0$ if $w_{1,i}\cdot\alpha_i + w_{2,i} \le 0$, otherwise we let $l_i = 1$. Hence we can convert our problem to choose $\boldsymbol{\alpha}$ as

$$\min_{\boldsymbol{\alpha}\in\Delta(K)} \max_{\boldsymbol{l}\in[0,1]^K} \langle \boldsymbol{w}_1, (\boldsymbol{\alpha}\odot\boldsymbol{l})\rangle + \langle \boldsymbol{w}_2, \boldsymbol{l}\rangle = \min_{\boldsymbol{\alpha}\in\Delta(K)} \sum_{i=1}^{K} \max\left\{0, \alpha_i w_{1,i} + w_{2,i}\right\},$$

which is equivalent to

$$\min_{\boldsymbol{\alpha}\in\Delta(K),\boldsymbol{\beta}\geq\mathbf{0}} \sum_{i=1}^{K} \beta_i \quad \text{s.t. } \beta_i \geq w_{1,i}\alpha_i + w_{2,i} \quad \forall i = 1,\ldots,K.$$

The above problem is a linear program with $O(K)$ variables and $O(K)$ linear constraints. Thus, computing $\boldsymbol{\alpha}_t$ in the problem (3.7) can be solved in polynomial time.

3.4.2 Computing subgradients g_t for the ∞-norm

The second component of Algorithm 4 is the algorithm $\mathcal{B}$, which computes subgradients $\boldsymbol{s}_t \in \partial h_S(\boldsymbol{w}_t)$. By Proposition 1, we have $\boldsymbol{s}_t = \arg\max_{\boldsymbol{s}\in S}\langle \boldsymbol{s}, \boldsymbol{w}_t\rangle$. Recall that $S = \{(\boldsymbol{x},\boldsymbol{y}) \in [0,1]^K \times [0,1]^K \mid \|\boldsymbol{x}\|_\infty \leq C^*_\infty(\boldsymbol{y})\}$. In particular, the condition that $\|\boldsymbol{x}\|_\infty \leq C^*_\infty(\boldsymbol{y})$ can be represented as

$$\max_i x_i \leq \min_{\boldsymbol{\alpha}\in\Delta(K)} \|\boldsymbol{\alpha}\odot\boldsymbol{y}\|_\infty \iff x_i \leq \frac{1}{\sum_{j=1}^K \frac{1}{y_j}}, \forall i.$$

Therefore, the computation of the subgradient $\boldsymbol{s}_t$ is formulated as

$$\max_{\boldsymbol{x},\boldsymbol{y}\in[0,1]^K} \langle \boldsymbol{w}_1, \boldsymbol{x}\rangle + \langle \boldsymbol{w}_2, \boldsymbol{y}\rangle \quad \text{s.t. } x_i \leq \frac{1}{\sum_j \frac{1}{y_j}}, \quad \forall i = 1,\ldots,K. \tag{3.8}$$

Now we show that there exists an equivalent second order cone programming (SOCP) formulation (e.g., [41]) for this problem.

First we give the definition of the second order cone programming, and then we give a proposition, which states that our optimization problem is equivalent to the second order cone programming.

Definition 4. *The standard form for the second order conic programming (SOCP) model is as follows:*

$$\min_{\boldsymbol{x}} \langle \boldsymbol{c}, \boldsymbol{x}\rangle \text{ s.t. } A\boldsymbol{x} = \boldsymbol{b}, \|C_i\boldsymbol{x} + \boldsymbol{d}_i\|_2 \leq \boldsymbol{e}_i^\top \boldsymbol{x} + \boldsymbol{f}_i \quad \text{for } i = 1,\cdots,m,$$

where the problem parameters are $\boldsymbol{c} \in \mathbb{R}^n$, $C_i \in \mathbb{R}^{n_i\times n}$, $\boldsymbol{d}_i \in \mathbb{R}^{n_i}$, $\boldsymbol{e} \in \mathbb{R}^n$, $\boldsymbol{f}_i \in \mathbb{R}$, $A \in \mathbb{R}^{p\times n}$, *and* $\boldsymbol{b} \in \mathbb{R}^p$. $\boldsymbol{x} \in \mathbb{R}^n$ *is the optimization variable.*

Then we obtain the following proposition.

Proposition 4. $\sum_{i=1}^{K} \frac{x^2}{y_i} \leq x, x \geq 0$ *and* $y_i \geq 0$ *is equivalent to* $x^2 \leq y_i z_i$, *where* $y_i, z_i \geq 0$ *and* $\sum_{i=1}^{K} z_i = x$.

Proof. On the direction " $\Rightarrow$ "

From $\sum_{i=1}^{k} \frac{x^2}{y_i} \leq x$ we obtain that $\sum_{i=1}^{k} \frac{x}{y_i} \leq 1$. By setting

$$z_i = x \cdot \frac{\frac{1}{y_i}}{\sum_i \frac{1}{y_i}},$$

we can have that $x^2 \leq y_i z_i$, and $\sum_{i=1}^{k} z_i = x$.

On the other direction " $\Leftarrow$ " Due to $x^2 \leq y_i z_i$, we have $\frac{x^2}{y_i} \leq z_i$. So we have that

$$\sum_{i=1}^{k} \frac{x^2}{y_i} \leq \sum_{i=1}^{k} z_i = x.$$

□

Again in our case we need find to the optimal vector $\boldsymbol{s} \in S$, which satisfies that $\boldsymbol{s}_t = \arg\max_{\boldsymbol{s} \in S} \langle \boldsymbol{w}_t, \boldsymbol{s} \rangle$. Then we can reduce our problem in following theorem.

Theorem 6. *The optimization problem (3.8) can be solved by the second order cone programming.*

Proof. To prove this theorem we only need to represent the original problem (3.8) as a standard form of the SOCP problem. Note that we only consider the case that $y_i \neq 0$ for all $i = 1, \ldots, K$. The case where $y_i = 0$ for some i is trivial. To see this, by definition of S, we know that for all i, $x_i = 0$. Then, the resulting problem is a linear program, which is a special case of the SOCP. Now we assume that $y_i \neq 0$ for $i = 1, \ldots, K$. For $x_i \leq \frac{1}{\sum_j \frac{1}{y_j}}$, we multiply x_i on both sides and rearrange the inequality:

$$\sum_{j=1}^{K} \frac{x_i^2}{y_j} \leq x_i.$$

By Proposition 4, this is equivalent with

$$y_j z_{i,j} \geq x_i^2, \quad y_j, z_{i,j} \geq 0, \quad \sum_{j=1}^{K} z_{i,j} = x_i.$$

By [41], we may rewrite it as follows: For each i,

$$x_i^2 \leq y_j z_{i,j}; \quad y_j, z_{i,j} \geq 0 \Longleftrightarrow \|(2x_i, y_j - z_{i,j})\|_2 \leq y_j + z_{i,j} \quad \forall j = 1, \ldots, K. \tag{3.9}$$

The above equivalence is trivial. On the other hand, since $x_i \leq \frac{1}{\sum_j \frac{1}{y_j}}$, and $y_i \in [0,1]$, naturally we have $x_i \in [0,1]$. So we need only constrain that $y_i \in [0,1]$. We can apply the face that if y_i is positive so $|y_i| = y_i$, and if $y_i \leq 1$, so $|y_i| \leq 1$. Therefore we may give a $(K^2+2K)\times(K^2+2K)$-matrix C_i in SOCP, and the variable vector is composed as follows:

$$\tilde{\boldsymbol{x}} = (x_1, \cdots, x_K, y_1, \cdots, y_K, z_{1,1}, \cdots, z_{1,K}, \cdots, z_{K,1} \cdots, z_{K,K}), \tag{3.10}$$

where for $z_{i,j}$, i is corresponding to x_i.

Now we may give the second order cone programming of our target problem as follows:

$$\begin{aligned} &\min_{\tilde{\boldsymbol{x}}} \langle -(\boldsymbol{w}_1, \boldsymbol{w}_2, 0, \cdots, 0), \tilde{\boldsymbol{x}} \rangle \\ &\text{s.t.} \|C_i \tilde{\boldsymbol{x}}\|_2 \leq \boldsymbol{e}_i^\top \tilde{\boldsymbol{x}} + \boldsymbol{d}_i \quad \forall i = 1, \cdots, K^2 + 2K, \\ &A\tilde{\boldsymbol{x}} = \boldsymbol{b}. \end{aligned} \tag{3.11}$$

where C_i, $\boldsymbol{e}_i$, A and $\boldsymbol{b}$ are defined as follows:

Firstly the matrix C for hyperbolic constraints are given as: For a fixed $s \in [K]$, where $[K] = \{1, \cdots, K\}$ in matrix C_i, where $i \in [(s-1)K, sK]$ we let $(C_i)_{1,s} = 2$, $(C_i)_{K+i,K+i} = 1$, $(C_i)_{2K+(s-1)K+i,2K+(s-1)K+i} = -1$, and others are 0. $\boldsymbol{e}_i$ is defined as $(\boldsymbol{e}_i)_{K+i} = 1$ and $(\boldsymbol{e}_i)_{2K+(s-1)K+i} = 1$, others are 0.

Next we need to constrain that y_i is less than 1. For $i \in [K^2, K^2+K]$ we let that $(C_i)_{K+i,K+i} = 1$ and others are 0. And we let that $\boldsymbol{e}_i$ is a zero vector and $\boldsymbol{d}_i = 1$. It means that $\|y_i\| \leq 1$. For $i \in [K^2+K, K^2+2K]$, we set $(C_i)_{K+i,K+i} = 1$ $\boldsymbol{e}_{K+i} = 1$, and $\boldsymbol{d}_i = 0$

At last we need to constrain that $\sum_{j=1}^K z_j = x_i$ in equation 3.9: Let $A \in \mathbb{R}^{K\times(3K+K^2)}$ for each row vector A_j, where $j \in [K]$, we have that $(A_j)_j = 1$ and $(A_j)_{2K+(j-1)j+m} = -1$, for all $m = 1, \cdots, K$. No w the matrix A is composed by the row vectors A_j. and $\boldsymbol{b}$ is a zero vector. □

3.5 Conclusion

In this paper we give a framework for online load balancing problem by reducing it to two OLO problems. Moreover, for online load balancing problem with respect to L_∞-norm we achieve the best known regret bound in polynomial time. Firstly, we reduce online load balancing with $\|\cdot\|$ norm to a vector payoff game measured by combination norm $\|\cdot\|^+$. Next due to [50] this vector payoff game is reduced to an OCO problem. At last, we can reduce this OCO problem

to two independent OLO problems. Especially, for makespan, we give an efficient algorithm, which achieves the best known regret bound $O(\sqrt{T \ln K})$, by processing linear programming and second order cone programming in each trial. Recently Kwon [37] proposed a similar reduction with other type of induced norm instead of our combination norm.

There are some open problems left in this topic. For instance, an efficient algorithm for online load balancing with respect to general norm or p-norm is still an open problem. Kwon's reduction [37] might be helpful to this discussion. Furthermore, the lower bound of online load balancing is still unknown.

Chapter 4

Expert advice problem with noisy low rank loss

4.1 Introduction

The expert advice problem with low rank loss [29] is an extension of the standard expert problem by considering a latent structure in losses. In this problem, we model the expert advice with d-rank loss as follows: the environment chooses a full rank $N \times d$ matrix U, called kernel. On each round $t \in [T]$, the algorithm picks a prediction $\boldsymbol{w}_t$ in an N-dimensional simplex over the set of N experts. Then the environment gives a d-rank loss vector $\boldsymbol{l}_t \in [-1, 1]^N$, where $\boldsymbol{l}_t = U\boldsymbol{v}_t$ to the algorithm. At last the algorithm suffers the loss $\boldsymbol{w}_t \cdot \boldsymbol{l}_t$. We measure the performance of this algorithm based difference between the cumulative loss and the best expert strategy in hindsight, which is defined as regret in the following equation:

$$\text{Regret}_T = \sum_{t=1}^{T} \boldsymbol{w}_t \cdot \boldsymbol{l}_t - \min_{i \in [N]} \sum_{t=1}^{T} \boldsymbol{l}_t(i). \tag{4.1}$$

This low rank loss setting is popular in recommendation system, especially when experts give losses based on a latent structure [36]. For instance, these experts share some common information, or their prediction methods are similar and depended on only few factors. As a consequence, the loss vectors given by experts are in fact in a relatively lower dimensional space, which implies that $d \leq N$.

Compared with the original expert advice problem, whose regret bound is $\Theta(\sqrt{T \ln N})$ with Hedge algorithm [21, 13], the expert advice problem with d-rank loss incurs an upper bound as $O(d\sqrt{T})$ [29], and $O(\sqrt{dT \ln T})$ [35], respectively. Moreover, the optimal bound is

shown to be $\Theta(\sqrt{dT})$ if the kernel is known to the algorithm [29]. The work of Hazan et al. [29] combines two parts, recovering the kernel U and predicting the $\boldsymbol{w}_t$. Hence, this algorithm not only provides a more precise upper bound, if $d \leq o(\sqrt{\ln N})$, but also re-constructs the kernel while running the algorithm. Equivalently, kernel U is explored and exploited in the learning process. Actually, in the algorithm of Hazan et al. [29], Online mirror descent (OMD) is designed to predict $\boldsymbol{w}_t$ utilising the recovered U. Without U, OMD can give only a loose bound as $O(\sqrt{NT})$, even if the loss is d-ranked. In contrast, if the kernel is acquired in advance, OMD achieves the optimal bound $O(\sqrt{dT})$.

After this pioneering work [29], Barman et al. [8] considered a case that, the d-rank loss $\boldsymbol{l}_t$ is corrupted by a L_2-norm bounded $\boldsymbol{\epsilon}_t$ noise, i.e., the environment gives the loss vector $\tilde{\boldsymbol{l}}_t = \boldsymbol{l}_t + \boldsymbol{\epsilon}_t = U\boldsymbol{v}_t + \boldsymbol{\epsilon}_t$, which is near to the d-dimensional space. In their work, Barman et al. [8] gave a regret bound $O\left(\sqrt{(d+\epsilon)T}\right)$ and a lower bound as $\Omega\left(\sqrt{T(d+\frac{\epsilon}{2^d})}\right)$, where $\|\boldsymbol{\epsilon}_t\|_2^2 \leq \epsilon$. However, there is a strong assumption in their work that the algorithm needs know both U and ϵ.

In this paper, we release the strong assumption in Barman et al. [8], and assume that the kernel is unknown to the algorithm, while the low rank loss is corrupted by $\boldsymbol{\epsilon}_t$. This problem setting is more realistic in two points: One is the loss vectors are not always strictly well-structured. Although the experts share some common prediction methods, there is still some slight disturbance, which damages the low rank structure loss. The other is the specific structure is unknown to the algorithm before it receives the loss vectors. It is natural that the recommendation system needs to confront new users who are recently registered without any prior information. Thus, the system is required to explore the characteristics of users while recommending their goods.

From the above works, we can see how the low rank structure plays the important role in the algorithms. Either, the kernel is known to the algorithm, then the algorithm can overcome the noised loss vectors easily; or, the loss is precisely low ranked, then the algorithm can extract the kernel from the received losses, even if the kernel is unknown. However, if we apply the algorithms of Hazan et al. [29], and Koren and Livni [35] in our problem setting directly, it is equivalent to run OMD on expert advice directly, and the regret bound is $O(\sqrt{NT})$, since the low rank loss no longer exists. Hence, one of the essential problems is how to deal with the "noisy low rank loss" and recover the underlying kernel U. Although recovering a low rank matrix is not a novel concept for machine learning [24, 20, 6, 53], to the best of the authors knowledge, there is no such research for online expert advice problem with noise attached loss vectors. The most tricky obstacle is that, it is impossible to re-construct the kernel U exactly

with the corrupted loss vectors. Thus, in our proposed algorithm, we attempt to approximate this kernel from the received loss vectors.

In our algorithm, we assume that there is no prior information about the kernel. Thus, we select at most d highly-independent loss vectors (see details in latter section) to construct a pseudo-kernel $\widetilde{U}$ in our learning process. Unlike the selection criteria in work [6], which recovers the kernel with a randomised algorithm stochastically, in our paper, each selected vector is supposed to be not only "long" enough, but also far enough to the current sub-space, spanned by all previously selected vectors. Therefore, our algorithm can approximate the kernel of the low rank loss deterministically and obtains a worst case guarantee. Instead of the unknown U in [29], we next utilise the pseudo-kernel, which is spanned by the selected vectors, in OMD to update the predictions. Our algorithm obtains an upper bound with respect to $\tilde{\boldsymbol{l}}_t$ as $O(\sqrt{T}(d + d^{4/3}(N\epsilon)^{1/3}))$. In practice, we can consider our algorithm and the Hedge algorithm as two meta-experts and run another Hedge algorithm on top of them, which performs nearly as well as our algorithm and the Hedge algorithm with additional negligeable ($O(\sqrt{T})$) regret.

The following table shows the relationship between our work and previous work.

Table 4.1: Comparison with noisy and non-noisy low rank loss

	known kernel	unknown kernel
without noise	$\Theta(\sqrt{dT})$ [29]	$O(d\sqrt{T})$ [29] $O(\sqrt{dT\ln T})$ [35]
ϵ-noise case	$O(\sqrt{(d+\epsilon)T})$ $\Omega\left(\sqrt{T(d+\frac{\epsilon}{2^d})}\right)$ [8]	$O((1+\epsilon)\sqrt{T}(d + d^{4/3}(N\epsilon)^{1/3}))$ (Our work)

This paper is composed as follows. In the second section, we define our expert advice with noisy d-rank loss formally and some necessary notations. In the third section, the proposed algorithm is given and we show the upper bound. In section 4, we compare the proposed algorithm and previous work under synthetic environments[1].

4.2 Preliminaries

We denote the set $\{1, \cdots, T\}$ by $[T]$. We denote the N-dimensional vector with all 1 elements by $\mathbb{I}_N$ and $N \times N$ identity matrix by I_N. We define a norm $\|\boldsymbol{x}\|_H$ with respect to a positive definite matrix H, as $\|\boldsymbol{x}\|_H = \sqrt{\boldsymbol{x}^T H \boldsymbol{x}}$, for a vector $\boldsymbol{x}$, and the dual norm of

[1]The code is available at https://github.com/2015211217/LowRankStructureC-.git

$\|\cdot\|_H$ is defined as $\|\boldsymbol{x}\|_H^* = \sqrt{\boldsymbol{x}^T H^{-1}\boldsymbol{x}}$. We define the angle between two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ as $\theta(\boldsymbol{u},\boldsymbol{v}) = \arccos\frac{\langle \boldsymbol{u},\boldsymbol{v}\rangle}{\|\boldsymbol{u}\|_2\|\boldsymbol{v}\|_2}$. If $\boldsymbol{V}$ is a subspace then $\theta(\boldsymbol{u},\boldsymbol{V}) = \min_{\boldsymbol{v}\in\boldsymbol{V}\setminus\{0\}}\theta(\boldsymbol{u},\boldsymbol{v})$, and $\theta(\boldsymbol{U},\boldsymbol{V}) = \max_{\boldsymbol{u}\in U}\theta(\boldsymbol{u},\boldsymbol{V})$. At last, we denote the orthogonal projection from a vector $\boldsymbol{l}$ to a subspace $\boldsymbol{U}$ as $\mathcal{P}_{\boldsymbol{U}}\boldsymbol{l}$. Moreover, the Euclidean distance from a vector $\boldsymbol{l}$ to a sub-space $\boldsymbol{U}$ is defined as $\|\boldsymbol{l}-\mathcal{P}_{\boldsymbol{U}}\boldsymbol{l}\|_2 = \|\boldsymbol{l}\|_2\sin\theta(\boldsymbol{l},\boldsymbol{U}) = \|\boldsymbol{l}\|_2\sin\theta(\boldsymbol{l},\mathcal{P}_{\boldsymbol{U}}\boldsymbol{l})$. A linear space spanned by all columns from a matrix $V\in\mathbb{R}^{N\times k}$ can be represented as $\mathrm{span}(V)$. In the following part, we simplify $\mathrm{span}(V)$ as $\boldsymbol{V}$, when it leads no ambiguity.

4.2.1 Problem Setting

Firstly we define the d-rank loss: For a sequence of $\boldsymbol{l}_t\in[-1,1]^N$, for $t\in[T]$ we define $L_T\in\mathbb{R}^{N\times T}$ as follows:

$$L_T = [\boldsymbol{l}_1,\boldsymbol{l}_2,\cdots,\boldsymbol{l}_T], \tag{4.2}$$

where each loss vector $\boldsymbol{l}_t$ is a column in L_T. We define that the sequence $\{\boldsymbol{l}_1,\cdots,\boldsymbol{l}_T\}$ is d-rank loss if and only if $\mathrm{rank}(L_T)=d$. Equivalently, for the sequence $\{\boldsymbol{l}_1,\cdots,\boldsymbol{l}_t\}$, there exists a kernel $U\in\mathbb{R}^{N\times d}$ such that $\boldsymbol{l}_t = U\boldsymbol{v}_t$, where $\mathrm{rank}(U)=d$. If $d\ll N$, we can call $\boldsymbol{l}_t$ as low rank loss vector.

From d-rank loss $\boldsymbol{l}_t$, we define the noisy d-rank loss $\tilde{\boldsymbol{l}}_t$: $\tilde{\boldsymbol{l}}_t = \boldsymbol{l}_t+\boldsymbol{\epsilon}_t$, where $\|\boldsymbol{\epsilon}_t\|_2\le\epsilon,\forall t\in[T]$. We call $\boldsymbol{\epsilon}_t$ as the noise vector.

In this paper we consider the following learning problem of online expert advice with noisy d-rank loss. On round $t=1,\cdots,T$, an algorithm gives a prediction $\boldsymbol{w}_t\in\Delta(N)$; Then an environment gives a loss vector $\tilde{\boldsymbol{l}}_t\in[-1-\epsilon,1+\epsilon]^N$, note that $\tilde{\boldsymbol{l}}_t = \boldsymbol{l}_t+\boldsymbol{\epsilon}_t$. It implies that there exists an underlying $\boldsymbol{l}_t\in[-1,1]^N$ as d-rank loss and $\boldsymbol{\epsilon}_t$ as ϵ-noise. However, $\boldsymbol{l}_t, U$ and $\boldsymbol{\epsilon}_t$ are unknown to the algorithm. At last the algorithm suffers the loss as $\boldsymbol{w}_t\cdot\tilde{\boldsymbol{l}}_t$. Next we define the regret as follows:

$$\mathrm{Regret}_T = \sum_{t=1}^{T}\boldsymbol{w}_t\cdot\tilde{\boldsymbol{l}}_t - \min_{i\in[N]}\sum_{t=1}^{T}\tilde{\boldsymbol{l}}_t(i). \tag{4.3}$$

4.2.2 Online mirror descent

Online mirror descent (OMD) is a basic algorithm utilised in both Hazan et al. [29] and Barman et al. [8], as well as in this paper. We give OMD with the time-varying matrix norms as follows:

Algorithm 5 Online Mirror descent

Require: $H_0, \cdots, H_{T-1} \succ 0$, $\{\eta_t\}_{t=1}^T$, and $\boldsymbol{w}_1 \in \Delta(N)$.
  for $t = 1, \cdots, T$ **do**
    Receive $\boldsymbol{l}_t$,
    Suffer cost $\boldsymbol{w}_t \cdot \boldsymbol{l}_t$,
    Update $\boldsymbol{w}_{t+1} = \arg\min_{\boldsymbol{w} \in \Delta(N)} \boldsymbol{l}_t \cdot \boldsymbol{w} + \eta_t^{-1} \|\boldsymbol{w} - \boldsymbol{w}_t\|_{H_{t-1}}^2$.
  end for

Theorem 7 (Orabona et al. [45])**.** *The T-round regret of OMD is bounded as follow:*

$$\sum_{t=1}^{T} \boldsymbol{w}_t \cdot \boldsymbol{l}_t - \sum_{t=1}^{T} \boldsymbol{l}_t \cdot \boldsymbol{w}^* \leq \frac{1}{\eta_T} \|\boldsymbol{w}_1 - \boldsymbol{w}^*\|_{H_T}^2 + \frac{1}{2} \sum_{t=1}^{T} (\eta_t \|\boldsymbol{l}_t\|_{H_t}^*)^2 + \sum_{t=1}^{T} \eta_t^{-1} (\|\boldsymbol{w}_t\|_{H_{t-1}}^2 - \|\boldsymbol{w}_t\|_{H_t}^2).$$

4.2.3 Ellipsoid approximation of convex bodies

Given a positive semi-definite matrix $M \succeq 0$, we define the ellipsoid with respect to M as follows:

$$\mathcal{E}(M) = \{\boldsymbol{x} : \boldsymbol{x}^T M^\dagger \boldsymbol{x} \leq 1\}, \tag{4.4}$$

where $M^\dagger$ is the Moore-Penrose pseudo-inverse matrix of M. In the following theorem we give a result for minimum volume enclosing ellipsoid (MVEE) to a central symmetric convex body.

Theorem 8 (John's Theorem [7])**.** *Let K be a convex body in $\mathbb{R}^d$ that is symmetric around zero. Let $\mathcal{E}$ be an ellipsoid with minimum volume enclosing K. Then:*

$$\frac{1}{\sqrt{d}} \mathcal{E} \subseteq K \subseteq \mathcal{E}. \tag{4.5}$$

Moreover for a polytope defined as $P_A = \{\boldsymbol{x} : \|A\boldsymbol{x}\|_\infty \leq 1, \boldsymbol{x} \in \mathbb{R}^d\}$, corresponding to a given matrix $A \in \mathbb{R}^{N \times d}$, we have the following theorem.

Theorem 9 (Grötschel et al. [25])**.** *There exists a poly-time procedure* $\mathrm{MVEE}(A)$ *that receives as input a matrix $A \in \mathbb{R}^{N \times d}$ and returns a matrix M such that*

$$\frac{1}{\sqrt{2d}} \mathcal{E}(M) \subseteq P_A \subseteq \mathcal{E}(M).$$

4.3 Algorithm for no prior information about kernel

In this section, we consider the case that there is no prior information about the kernel in the online expert advice with noisy d-rank loss problem. Under this assumption, we construct a

pseudo kernel, $\widetilde{U}$, in our learning process, when the d-rank structure is corrupted by $\boldsymbol{\epsilon}_t$. Note that in this algorithm, we construct this pseudo kernel cumulatively. We denote it as $\widetilde{U}^k \in \mathbb{R}^{N\times k}$ for convenience.

Concisely, our algorithm will select some loss vectors to construct the pseudo kernel and run OMD with respect to from $\tilde{U}^k$ processed matrix H^k. The criteria to select $\tilde{\boldsymbol{l}}_t$ is twofold: firstly, the L_2-norm of $\tilde{\boldsymbol{l}}_t$ is supposed to be large enough; then, the Euclidean distance from $\tilde{\boldsymbol{l}}_t$ to the current pseudo kernel spanned space $\mathrm{span}(\widetilde{U}^k)$ should be far enough. Thus, for some $k \in [d]$ and corresponding parameters s_k, γ_k, the basic idea of our algorithm is as follows:

1. $\|\tilde{\boldsymbol{l}}_t\|_2 \leq 2s_k \rightarrow$ Do OMD with respect to matrix H^k.
2. $\|\tilde{\boldsymbol{l}}_t\|_2 \geq 2s_k$ and $\|\tilde{\boldsymbol{l}}_t - \mathcal{P}_{\mathrm{span}(\widetilde{U}^k)}\tilde{\boldsymbol{l}}_t\|_2 \leq 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon) \rightarrow$ Do OMD with respect to H^k.
3. $\|\tilde{\boldsymbol{l}}_t\|_2 \geq 2s_k$ and $\|\tilde{\boldsymbol{l}}_t - \mathcal{P}_{\mathrm{span}(\widetilde{U}^k)}\tilde{\boldsymbol{l}}_t\|_2 \geq 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon) \rightarrow$ Adding $\tilde{\boldsymbol{l}}_t$ as a column to $\widetilde{U}^k$, and reset k as $k+1$, then do OMD with respect to H^k.

Remark 2. *If we have that $\|\tilde{\boldsymbol{l}}_t\| \geq 2s_k > 2\epsilon$, then we obtain the following result: Since $\|\tilde{\boldsymbol{l}}_t - \boldsymbol{l}_t\|_2 \leq \epsilon$, $\theta(\tilde{\boldsymbol{l}}_t, \boldsymbol{l}_t) \leq 2\sin\theta(\tilde{\boldsymbol{l}}_t, \boldsymbol{l}_t) \leq \frac{\epsilon}{s_k}$*

According to this criteria, we can observe that we need to guarantee that $K = \max k \leq d$, to ensure that our algorithm effective, since the algorithm from [29] updates N-times for low rank noisy loss vector. It implies that the previous algorithm in [29] is OMD and obtains $O(\sqrt{(N+\epsilon)T})$ actually.

We confirm s_k, γ_k according to the following proposition.

Proposition 5. *Define a sequence s_i and non-decreasing sequence γ_i such that $\gamma_i \in (0, \frac{\pi}{2})$ and*

$$\frac{\gamma_i}{\gamma_{i-1} - (i-1)\frac{\beta\epsilon}{s_{i-1}\gamma_{i-1}}} \cdot \frac{\pi}{2\beta} + \frac{s_i\gamma_i}{s_i\gamma_{i-1}}(i-1) \leq i \quad \forall i \in \{2, \cdots, k\}, \tag{4.6}$$

and

$$\gamma_i - \frac{\beta\epsilon}{s_i\gamma_i} i > 0 \quad \forall i \in \{1, \cdots, k\}. \tag{4.7}$$

Given any space $\boldsymbol{W} \subseteq \mathbb{R}^N$: Let $\boldsymbol{U}^k = \mathrm{span}\{\boldsymbol{W}, \boldsymbol{l}_1, \cdots, \boldsymbol{l}_k\}$ and $\widetilde{\boldsymbol{U}}^k = \mathrm{span}\{\boldsymbol{W}, \tilde{\boldsymbol{l}}_1, \cdots, \tilde{\boldsymbol{l}}_k\}$ be two subspace such that $\theta(\boldsymbol{l}_i, \tilde{\boldsymbol{l}}_i) \leq \epsilon/s_i$ for all $i \in \{2, \cdots, k\}$.

If $\theta(\tilde{\boldsymbol{l}}_i, \widetilde{\boldsymbol{U}}^{i-1}) > \gamma_i$, and $\beta \geq \frac{\pi}{2}$ then we have that

$$\theta(\boldsymbol{U}^k, \widetilde{\boldsymbol{U}}^k) < \beta k \frac{\epsilon}{s_k\gamma_k}. \tag{4.8}$$

Proof. For $i = 1$, due to Lemma 5, by setting that $\theta(l_1, \emptyset) = \frac{\pi}{2}$, it is trivial that

$$\theta(\boldsymbol{U}^1, \widetilde{\boldsymbol{U}}^1) \leq \frac{\pi\epsilon}{2s_1\gamma_1} \leq \beta\frac{\epsilon}{s_1\gamma_1}. \tag{4.9}$$

Here $\theta(\tilde{\boldsymbol{l}}_1, \boldsymbol{l}_1) \leq \frac{\epsilon}{s_1}$, and $\theta(\tilde{\boldsymbol{l}}_1, \widetilde{\boldsymbol{U}}^0) = \theta(\tilde{\boldsymbol{l}}_1, \boldsymbol{W}) \geq \gamma_1$. If $\boldsymbol{W} = \emptyset$ then we have that

$$\theta(\boldsymbol{U}^1, \widetilde{\boldsymbol{U}}^1) = \theta(\boldsymbol{l}_1, \tilde{\boldsymbol{l}}_1) \leq \frac{\epsilon}{s_1} \leq \frac{\pi}{2} \times \frac{\epsilon}{\gamma_1 s_1},$$

if $\gamma_1 \leq \frac{\pi}{2}$.

Then if it holds for $i-1$: Let us involve $\boldsymbol{U}_0^i = \text{span}\{\boldsymbol{U}^{i-1}, \tilde{\boldsymbol{l}}_i\}$, note that

$$\theta(\boldsymbol{U}_0^i, \widetilde{\boldsymbol{U}}^i) = \theta(\text{span}\{\boldsymbol{U}^{i-1}, \tilde{\boldsymbol{l}}_i\}, \text{span}\{\widetilde{\boldsymbol{U}}^{i-1}, \tilde{\boldsymbol{l}}_i\}) \tag{4.10}$$

Since the induction hypothesis so we have that

$$\theta(\text{span}\{\boldsymbol{W}, \boldsymbol{l}_1, \cdots, \boldsymbol{l}_{i-1}\}, \text{span}\{\boldsymbol{W}, \tilde{\boldsymbol{l}}_1 \cdots, \tilde{\boldsymbol{l}}_{i-1}\}) \leq (i-1)\frac{\beta\epsilon}{s_{i-1}\gamma_{i-1}}. \tag{4.11}$$

then we have that

$$\begin{aligned}\theta(\boldsymbol{U}^i, \widetilde{\boldsymbol{U}}^i) &\leq \theta(\boldsymbol{U}^i, \boldsymbol{U}_0^i) + \theta(\boldsymbol{U}_0^i, \widetilde{\boldsymbol{U}}^i) \\ &\leq \frac{\pi}{2}\frac{\theta(\tilde{\boldsymbol{l}}_i, \boldsymbol{l}_i)}{\theta(\tilde{\boldsymbol{l}}_i, \boldsymbol{U}^{i-1})} + \theta(\boldsymbol{U}_0^i, \widetilde{\boldsymbol{U}}^i)\end{aligned}$$

The first inequality is from Remark 3. The second inequality is due to Lemma 5.

Now due to the Lemma 7 we have that

$$\theta(\boldsymbol{U}^i, \widetilde{\boldsymbol{U}}^i) \leq \frac{\pi}{2}\frac{\theta(\boldsymbol{l}_i, \tilde{\boldsymbol{l}}_i)}{\theta(\tilde{\boldsymbol{l}}_i, \widetilde{\boldsymbol{U}}^{i-1}) - \theta(\boldsymbol{U}^{i-1}, \widetilde{\boldsymbol{U}}^{i-1})} + \theta(\boldsymbol{U}_0^i, \widetilde{\boldsymbol{U}}^i) \tag{4.12}$$

Since the fact that $\theta(\tilde{\boldsymbol{l}}_k, \widetilde{\boldsymbol{U}}^{k-1}) > \gamma_k$, and the induction hypothesis we obtain that

$$\begin{aligned}\theta(\boldsymbol{U}^i, \widetilde{\boldsymbol{U}}^i) &< \frac{\pi}{2}\frac{\frac{\epsilon}{s_i}}{\gamma_{i-1} - \beta(i-1)\frac{\epsilon}{s_{i-1}\gamma_{i-1}}} + (i-1)\frac{\beta\epsilon}{s_{i-1}\gamma_{i-1}} \\ &= \frac{\epsilon\beta}{s_i\gamma_i}\left(\frac{\gamma_i}{\gamma_{i-1} - (i-1)\frac{\beta\epsilon}{s_{i-1}\gamma_{i-1}}} \times \frac{\pi}{2\beta} + (i-1)\frac{s_i\gamma_i}{s_{i-1}\gamma_{i-1}}\right)\end{aligned} \tag{4.13}$$

Since Equation 4.6 we have that

$$\frac{\epsilon\beta}{s_i\gamma_i}\left(\frac{\gamma_i}{\gamma_{i-1}-(i-1)\frac{\beta\epsilon}{s_{i-1}\gamma_{i-1}}}\times\frac{\pi}{2\beta}+(i-1)\frac{s_i\gamma_i}{s_{i-1}\gamma_{i-1}}\right)\leq\frac{\beta\epsilon}{s_i\gamma_i}i \tag{4.14}$$

Thus we have our conclusion. □

The description of the algorithm is shown in Algorithm 6.

Algorithm 6 Mirror descent with l_2 noise

Require: $\boldsymbol{w}_1=\frac{1}{N}\mathbb{I}_N$, $\tau=0$, $k=0$, $\widetilde{U}=\{\}$, a sequence of $\{s_k\}$ and a non decreasing sequence $\{\gamma_k\}$ for all $\gamma_k\in(0,\frac{\pi}{2})$ such that γ_k and s_k satisfies the conditions in Equation (4.6) and (4.7). Setting $m_k=\max\{2s_k,6\epsilon+\gamma_k\sqrt{N}\}$.

for $t=1,\cdots,T$ **do**

Observe $\tilde{\boldsymbol{l}}_t$, suffer loss $\boldsymbol{w}_t\cdot\tilde{\boldsymbol{l}}_t$.

if $\|\tilde{\boldsymbol{l}}_t\|_2\geq 2s_k$ **then**

if $\|\tilde{\boldsymbol{l}}_t-\mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t\|_2\geq 2\epsilon+\gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2+\epsilon)$ **then**

Add $\tilde{\boldsymbol{l}}_t$ as a new column of $\widetilde{U}^k$, reset $\tau=0$ and set $k\leftarrow k+1$.

Compute $M=\mathrm{MVEE}(\widetilde{U}^k)$ and $H^k=I_N+\widetilde{U}^kM(\widetilde{U}^k)^T$.

end if

end if

let $\tau\leftarrow\tau+1$ and $\eta_t=\sqrt{8k/(1+m_k)^2\tau}$ and set:

$$\boldsymbol{w}_{t+1}=\operatorname*{arg\,min}_{\boldsymbol{w}\in\Delta(N)}\tilde{\boldsymbol{l}}_t\cdot\boldsymbol{w}+\eta_t^{-1}\|\boldsymbol{w}-\boldsymbol{w}_t\|_{H^k}^2.$$

end for

According to our proposed algorithm, we see that our updating rule divides the T rounds into K epoches, where the final size of pseudo kernel, $K\leq d$ (will be show in the following Theorem). In each epoch, OMD procees with respect to H^k, a fixed matrix. Before we give the regret bound, we need some useful lemmata.

Lemma 3. *Let* $\|\tilde{\boldsymbol{l}}_t-\boldsymbol{l}_t\|_2\leq\epsilon$, *if* $\theta(\boldsymbol{l}_t,\mathrm{span}(\widetilde{U}^k))\leq\gamma_k$ *then we have*

$$\|\tilde{\boldsymbol{l}}_t-\mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t\|_2\leq 2\epsilon+\gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2+\epsilon). \tag{4.15}$$

Proof. Firstly we obtain that

$$\|\tilde{\boldsymbol{l}}_t-\mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t\|_2\leq\|\tilde{\boldsymbol{l}}_t-\boldsymbol{l}_t\|+\|\boldsymbol{l}_t-\mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\boldsymbol{l}_t\|+\|\mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\boldsymbol{l}_t-\mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t\|.$$

Since the fact that $\|\tilde{\boldsymbol{l}}_t - \boldsymbol{l}_t\| \leq \epsilon$, and $\|\boldsymbol{l}_t - \mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\boldsymbol{l}_t\| \leq \gamma_k\|\boldsymbol{l}_t\| \leq \gamma_k(\|\tilde{\boldsymbol{l}}_t\| + \epsilon)$, we have our conclusion. □

Lemma 3 states the criteria of distance between $\tilde{\boldsymbol{l}}_t$ and current space $\widetilde{\boldsymbol{U}}^k$.

Lemma 4. *Given $\widetilde{U}^k \in \mathbb{R}^{N\times k}$, and $\mathrm{span}(\widetilde{U}^k) \subset \mathbb{R}^N$, for any $\tilde{\boldsymbol{l}}_t$ such that $\|\tilde{\boldsymbol{l}}_t\|_2 \leq 2s_k$ or $\|\tilde{\boldsymbol{l}}_t - \mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t\|_2 \leq 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon)$, there exists a unique $\boldsymbol{v}_t \in \mathbb{R}^k$, and $\boldsymbol{e}_t$ such that $\tilde{\boldsymbol{l}}_t = \widetilde{U}^k\boldsymbol{v}_t + \boldsymbol{e}_t$, where $\theta(\boldsymbol{e}_t, \widetilde{\boldsymbol{U}}^k) = \frac{\pi}{2}$. Therefore we have that*

$$\|\boldsymbol{e}_t\|_2 \leq \max\{2s_k, 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon)\}. \tag{4.16}$$

In particular, we set that $\forall t \in [T]$

$$m_k = \max\{2s_k, 6\epsilon + \gamma_k\sqrt{N}\} \geq \max\{2s_k, 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon)\}. \tag{4.17}$$

Proof. Since the fact that $\|\boldsymbol{e}_t\|_2 \leq \|\tilde{\boldsymbol{l}}_t\|_2$ and the updating rule, we have that

$$\|\boldsymbol{e}_t\|_2 \leq \max\{2s_k, 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon)\}. \tag{4.18}$$

Due to our setting that $\gamma_k \leq \frac{\pi}{2} \leq 2$, and $\|\tilde{\boldsymbol{l}}_t\|_2 \leq \sqrt{N} + \epsilon$, thus we obtain Equation (4.17). □

Theorem 10. *Running Algorithm 6 on T-rounds, denoting that K is the final size of pseudo kernel, and $K \leq d$, we have*

$$\mathrm{Regret}_T \leq O\left((1+\epsilon)\sqrt{T}\left(K + \sqrt{\sum_{k=1}^{K} k m_k^2}\right)\right). \tag{4.19}$$

Proof. Firstly let us prove that $K \leq d$. Compared with $\widetilde{U}^k$ with k loss vectors of $\tilde{\boldsymbol{l}}_t$, we define a matrix U^k with respect to $\boldsymbol{l}_t$ corresponding to $\tilde{\boldsymbol{l}}_t$.

Due to our updating rule in algorithm, we have that

$$\theta(\tilde{\boldsymbol{l}}_t, \mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t) \geq \gamma_k,$$

since if $\|\tilde{\boldsymbol{l}}_t - \mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t\|_2 \geq 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon)$, then we have that

$$\theta(\tilde{\boldsymbol{l}}_t, \mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t) \geq \sin\theta(\tilde{\boldsymbol{l}}_t, \mathcal{P}_{\widetilde{\boldsymbol{U}}^k}\tilde{\boldsymbol{l}}_t) \geq \frac{2\epsilon + \gamma_k(\|\tilde{l}_t\|_2 + \epsilon)}{\|\tilde{\boldsymbol{l}}_t\|_2} \geq \gamma_k.$$

Therefore by Proposition 5 we have that $\theta(\mathrm{span}(U^k), \mathrm{span}(\widetilde{U}^k)) \leq \frac{\beta k \epsilon}{\gamma_k s_k}$. So we have that

$$\theta(\boldsymbol{l}_t, \mathrm{span}(U^k)) \geq \theta(\boldsymbol{l}_t, \mathrm{span}(\widetilde{U}^k)) - \theta(\mathrm{span}(U^k), \mathrm{span}(\widetilde{U}^k)) \geq \gamma_k - \frac{\beta\epsilon}{s_k \gamma_k} k > 0.$$

The first inequality is from Lemma 7. Since Lemma 3 we have that $\theta(\boldsymbol{l}_t, \mathrm{span}(\widetilde{U}^k)) \geq \gamma_k$. It implies that if $\|\tilde{\boldsymbol{l}}_t - \mathcal{P}_{\widetilde{\boldsymbol{U}}^k} \tilde{\boldsymbol{l}}_t\|_2 \geq 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_2\|_2 + \epsilon)$, then we have $\boldsymbol{l}_t \notin \mathrm{span}(U^k)$. So we have that $\mathrm{rank}(\widetilde{U}^K) \leq \mathrm{rank}(L_T) = d$.

Now let us prove the second part. Due to above result we can divide the total learning rounds T into K episodes. $T = \sum_{k=1}^{K} |T_k|$ and $T_k \cup T_{k'} = \emptyset$, if $k \neq k'$. For a given k, if $t \in T_k$, we obtain that $\|\tilde{\boldsymbol{l}}_t - \mathcal{P}_{\widetilde{\boldsymbol{U}}^k} \tilde{\boldsymbol{l}}_t\|_2 \leq 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_2\|_2 + \epsilon)$ or $\|\tilde{\boldsymbol{l}}_t\|_2 \leq 2s_k$. Thus we need give upper bound of $(\|\tilde{\boldsymbol{l}}_t\|_{H^k}^*)^2$, and $\|\boldsymbol{w}_1 - \boldsymbol{w}^*\|_{H^k}$.

Given a k-dimensional polytope as

$$P = \{\boldsymbol{v} \in \mathbb{R}^k : \|\widetilde{U}^k \boldsymbol{v}\|_\infty \leq 1 + \epsilon\},$$

we are able to have

$$\mathcal{E}(\frac{1}{2k} M) \subseteq P \subseteq \mathcal{E}(M), \tag{4.20}$$

due to the MVEE procedure, and Theorem 9 in previous section. For each $\tilde{\boldsymbol{l}}_t$ we give a unique orthogonal decomposition of $\tilde{\boldsymbol{l}}_t$ upon $\mathrm{span}(\widetilde{U}^k)$. Thus we obtain $\tilde{\boldsymbol{l}}_t = \widetilde{U}^k \boldsymbol{v}_t + \boldsymbol{e}_t$. Since $\tilde{\boldsymbol{l}}_t \in [0,1]^N$, and we have that $\boldsymbol{v}_t \in P$, and $\theta(\boldsymbol{e}_t, \widetilde{\boldsymbol{U}}^k) = \frac{\pi}{2}$. Therefore we have following inequality:

$$(\|\tilde{\boldsymbol{l}}_t\|_{H^k}^*)^2 = (\|\widetilde{U}^k \boldsymbol{v}_t + \boldsymbol{e}_t\|_{H^k}^*)^2 \leq (\|\widetilde{U}^k \boldsymbol{v}_t\|_{H^k}^* + \|\boldsymbol{e}_t\|_{H^k}^*)^2 \leq 2(\|\widetilde{U}^k \boldsymbol{v}_t\|_{H^k}^*)^2 + 2(\|\boldsymbol{e}_t\|_{H^k}^*)^2.$$

Firstly we give upper bound of $(\|\widetilde{U}^k \boldsymbol{v}_t\|_{H^k}^*)^2$.

$$\begin{aligned}(\|\widetilde{U}^k \boldsymbol{v}_t\|_{H^k}^*)^2 &= (\widetilde{U}^k \boldsymbol{v}_t)^T (I_N + \widetilde{U}^k M (\widetilde{U}^k)^T)^{-1} (\widetilde{U}^k \boldsymbol{v}_t) \\ &\leq (\widetilde{U}^k \boldsymbol{v}_t)^T (\widetilde{U}^k M (\widetilde{U}^k)^T)^+ (\widetilde{U}^k \boldsymbol{v}_t) = \boldsymbol{v}_t^T M^{-1} \boldsymbol{v}_t \leq 1.\end{aligned}$$

The last equality is due to Lemma 6 in Appendix.

Next we try to bound $(\|\boldsymbol{e}_t\|_{H^k}^*)^2$. We obtain that

$$(\|\boldsymbol{e}_t\|_{H^k}^*)^2 = \boldsymbol{e}_t^T H^{-1} \boldsymbol{e}_t = \boldsymbol{e}_t^T (I_N + \widetilde{U}^k M (\widetilde{U}^k)^T)^{-1} \boldsymbol{e}_t \leq \boldsymbol{e}_t^T I_n \boldsymbol{e}_t \leq \|\boldsymbol{e}_t\|_2^2. \tag{4.21}$$

Based on previous discussion we have that $\|\boldsymbol{e}_t\|_2 \leq \max\{2s_k, 2\epsilon + \gamma_k(\|\tilde{\boldsymbol{l}}_t\|_2 + \epsilon)\}$. Therefore

we have that

$$\|\boldsymbol{e}_t\|_2 \leq \max\{2s_k, 2\epsilon + \gamma_k(\sqrt{N} + \epsilon)\}. \tag{4.22}$$

Due to above Lemma 4, we have that $\|\boldsymbol{e}\|_2 \leq m_k$. In conclusion we have:

$$(\|\tilde{\boldsymbol{l}}_t\|_{H^k}^*)^2 \leq 2(1 + m_k)^2. \tag{4.23}$$

Now we show the bound of $\|\boldsymbol{w}_1 - \boldsymbol{w}^*\|_{H^k}$. Since $\|\boldsymbol{w}_1 - \boldsymbol{w}^*\|_{H^k} \leq 2\max_{\boldsymbol{w}\in\Delta(N)} \|\boldsymbol{w}\|_{H^k}$, it suffices to bound that $\max_{\boldsymbol{w}\in\Delta(N)} \|\boldsymbol{w}\|_{H^k}$. Since $\|\boldsymbol{w}\|_{H^k}^2 \leq 1 + 2k\|\boldsymbol{w}\|_{H(k)'}^2$ with $H(k)' = \frac{1}{2k}\widetilde{U}^k M(\widetilde{U}^k)^T$.

Given a convex set P in $\mathbb{R}^k$, so the dual set P^* is defined as

$$P^* = \{\boldsymbol{x} : \sup_{\boldsymbol{p}\in P} |\boldsymbol{x} \cdot \boldsymbol{p}| \leq 1\}.$$

The dual of an ellipsoid $\mathcal{E}(M)$ is given by $(\mathcal{E}(M))^* = \mathcal{E}(M^{-1})$ and since Equation (4.20) it is standard to show that

$$(\mathcal{E}(M))^* \subseteq P^* \subseteq (\mathcal{E}(\frac{1}{2k}M))^*. \tag{4.24}$$

It implies that $P^* \subseteq \mathcal{E}(2kM^{-1})$. Note that due to the definition of P^* we have that for all $i \in [N]$, $(1+\epsilon)^{-1}\tilde{\boldsymbol{u}}_i \cdot \boldsymbol{v} \leq 1$, where $\tilde{\boldsymbol{u}}_i$ is the i-th row of $\widetilde{U}^k$. So each row of $\widetilde{U}^k$ are in P^*, thus we have for each $\tilde{\boldsymbol{u}}_i$:

$$\begin{aligned}(1+\epsilon)^{-1}\tilde{\boldsymbol{u}}_i^T \left(2kM^{-1}\right)^{-1} (1+\epsilon)\tilde{\boldsymbol{u}}_i \leq 1 &\Leftrightarrow (1+\epsilon)^{-1}\tilde{\boldsymbol{u}}_i^T M (1+\epsilon)^{-1}\tilde{\boldsymbol{u}}_i \leq 2k \\ &\Leftrightarrow \|(1+\epsilon)^{-1}\tilde{\boldsymbol{u}}_i\|_M^2 \leq 2k\end{aligned} \tag{4.25}$$

Since $\boldsymbol{w} \in \Delta(N)$, we have that

$$\|\boldsymbol{w}\|_{H(k)'}^2 = \frac{1}{2k}\|\widetilde{U}\boldsymbol{w}\|_M^2 \leq \frac{1}{2k}\max_i \|\boldsymbol{u}_i\|_M^2 \leq (1+\epsilon)^2.$$

Therefore we have that $\|\boldsymbol{w}\|_{H^k}^2 \leq 1+(1+\epsilon)^2 2k$. Moreover we obtain that $\|\boldsymbol{w}\|_{H^k} \leq 2(1+\epsilon)\sqrt{k}$.

Given $\boldsymbol{w}^* = \arg\min_{t=1}^{T} \sum_{t=1}^{T} \boldsymbol{w} \cdot \tilde{\boldsymbol{l}}_t$, then we have that

$$\begin{aligned}
\text{Regret}_{T_k} &= \sum_{t \in T_k} \boldsymbol{w}_t \cdot \tilde{\boldsymbol{l}}_t - \sum_{t \in T_k} \tilde{\boldsymbol{l}}_t \cdot \boldsymbol{w}^* \\
&\leq \frac{1}{\eta_{T_k}} \|\boldsymbol{w}_{T_{k-1}} - \boldsymbol{w}^*\|_{H^k}^2 + \frac{1}{2} \sum_{t \in T_k} \eta_t (\|\tilde{\boldsymbol{l}}_t\|_{H^k}^*)^2 \\
&\leq \frac{4}{\eta_{T_k}} \max_{\boldsymbol{w} \in \Delta(N)} \|\boldsymbol{w}\|_{H^k}^2 + \frac{1}{2} \sum_{t \in T_k} \eta_t (\|\tilde{\boldsymbol{l}}_t\|_{H^k}^*)^2 \\
&\leq \frac{8k(1+\epsilon)^2}{\eta_{T_k}} + \sum_{t \in T_k} \eta_t (1+m_k)^2,
\end{aligned}$$

where we set that $\boldsymbol{w}_{T_{k-1}}$ as last term in episode T_{k-1}, the first inequality is due to Theorem 7.

Particularly, we can choose $\eta_t = \sqrt{8k(1+\epsilon)^2/(1+m_k^2)t}$, then we get the regret bound as

$$\text{Regret}_{\text{T}_\text{k}} \leq O(\sqrt{(8k(1+\epsilon)^2)(1+m_k)^2 T_k}).$$

Thus, we have the regret as

$$\text{Regret}_T \leq O\left((1+\epsilon)\sqrt{T}\left(K + \sqrt{\sum_{k=1}^{K} k m_k^2}\right)\right) \tag{4.26}$$

□

4.3.1 Parameter optimization

In this sub-section we are going to give an optimal setting of parameter γ_i, m_i and s_i. Since Equation (4.17), we can simplify

$$m_i = \max\{2s_i, 6\epsilon + \gamma_i\sqrt{N}\} \leq 2s_i + 2\gamma_i\sqrt{N}, \tag{4.27}$$

if we assume that $6\epsilon \leq \gamma_i\sqrt{N}, \forall i \in [K]$. Then we need to solve the following optimal problem:

$$\begin{aligned}
&\min s_i + \gamma_i\sqrt{N} \\
&\text{s.t.} \gamma_i - \frac{\beta\epsilon}{s_i\gamma_i} i > 0 \quad \wedge \quad \beta \geq \frac{\pi}{2} \geq \gamma_i > 0 \quad \forall i \in \{1, \cdots, K\}.
\end{aligned} \tag{4.28}$$

where the latter inequality is due to Proposition 5.

It is equivalent to solve:

$$\begin{aligned} &\min s_i + \gamma_i\sqrt{N} \\ &\text{s.t.} s_i \geq (1+x)\frac{\beta\epsilon i}{\gamma_i} \quad \wedge \quad \beta \geq \frac{\pi}{2} \geq \gamma_i > 0 \quad \forall i \in \{1, \cdots, K\}, \forall x > 0. \end{aligned} \tag{4.29}$$

With simple calculation we have that

$$\begin{cases} s_i = \left(N(1+x)\beta i\epsilon\right)^{1/3}, \\ \gamma_i = \left((1+x)\beta i\epsilon\right)^{1/3} N^{-1/6}, \end{cases} \tag{4.30}$$

for any $x > 0$.

Let us re-consider the constraint in Equation (4.6)

$$\frac{\gamma_i}{\gamma_{i-1} - (i-1)\frac{\beta\epsilon}{s_{i-1}\gamma_{i-1}}} \cdot \frac{\pi}{2\beta} + \frac{s_i\gamma_i}{s_i\gamma_{i-1}}(i-1) \leq i \quad \forall i \in \{2, \cdots, K\},$$

with above s_i and γ_i.

We can simplify this equation as

$$\frac{\pi}{2\beta} \leq i^{1/3}(i-1)^{1/3}(i^{1/3} - (i-1)^{1/3}) \cdot x, \forall i \in \{2, \cdots, K\}. \tag{4.31}$$

Without loss the generality, we can set that $x = 1$ and $\beta = 10$, according to Lemma 8.

Therefore if we set that $s_k = (20kN\epsilon)^{1/3}$ and $\gamma_k = \sqrt{20k\epsilon/s_k}$, in Algorithm 6, we have the following bound:

$$\text{Regret}_T \leq O\left((1+\epsilon)\sqrt{T}(K + K^{4/3}(N\epsilon)^{1/3})\right), \tag{4.32}$$

with the constraint that $6\epsilon \leq \gamma_k\sqrt{N} \Leftrightarrow \epsilon^2 \leq \frac{20kN}{216}$, $\gamma_k \leq \frac{\pi}{2} \Leftrightarrow \epsilon^2 \leq \frac{64\pi^6 N}{400k^2}$ and $s_k \geq \epsilon \Leftrightarrow \epsilon \leq 20kN$ for all $k \in [K]$, where the last constraint is due to Remark 2.

Corollary 3. *If* $\epsilon^2 \leq \min\{\frac{64\pi^6 N}{400k^2}, 20kN, \frac{20kN}{216}\}, \forall k \in [d]$, *setting* $s_k = (20kN\epsilon)^{1/3}$ *and* $\gamma_k = \sqrt{20k\epsilon/s_k}$, *running our algorithm for* T *times we have the regret bound as*

$$\text{Regret}_T \leq O\left((1+\epsilon)\sqrt{T}(K + K^{4/3}(N\epsilon)^{1/3})\right) \leq O\left((1+\epsilon)\sqrt{T}(d + d^{4/3}(N\epsilon)^{1/3})\right). \tag{4.33}$$

4.4 Experiments

We perform preliminary experiments using a synthetic environment. In our experiments, we construct the environment as follows: Firstly, we produce a $N \times d$ matrix U, if the rank of U is not d, then produce another one, until we obtain a d-rank matrix U as kernel.

To approach the maximal regret in our experiment, before the algorithms start, we randomly produce $T \times M$ seed vectors $\boldsymbol{v}_t^j$, where $\boldsymbol{v}_t^j(i) \in [-1, 1], \forall i \in [d]$, $t \in [T]$ and $j \in [M]$. Next we denote $\boldsymbol{l}_t^j = (U\boldsymbol{v}_t^j)/\|U\boldsymbol{v}_t^j\|_\infty$, and the noise vector $\boldsymbol{\epsilon}_t = s\tilde{\boldsymbol{\epsilon}}_t/(\|\tilde{\boldsymbol{\epsilon}}_t\|_2)$, where $\tilde{\boldsymbol{\epsilon}}_t(i)$ is randomly produced between $[-1, +1]$. We define $\tilde{\boldsymbol{l}}_t^j = \boldsymbol{l}_t^j + \boldsymbol{\epsilon}_t$. Then, we process Hedge, Hazan's and our algorithm(Algorithm 6) with the same input sequences accordingly, and by going this procedure for M sequence respectively, we can obtain M regret results for each algorithm, then choosing the maximum value among those M as follows:

On round t, for any algorithm $\mathcal{A}$, we record maximum of the regret with respect to M loss sequence as $\mathrm{Regret}_{\mathcal{A}}(t)$

$$\mathrm{Regret}_{\mathcal{A}}(t) = \max_{j\in[M]} \left\{ \sum_{s=1}^{t} \boldsymbol{w}_s^j \cdot \tilde{\boldsymbol{l}}_s^j - \min_{i\in[N]} \sum_{s=1}^{t} \tilde{\boldsymbol{l}}_s^j \right\}, \tag{4.34}$$

where we denote $\boldsymbol{w}_s^j$ as output of $\mathcal{A}$ with j-th loss sequence on round s.

Firstly, we set $T = 1000, M = 5, N = 300, d = 2.$ and plot $\mathrm{Regret}_{\mathcal{A}}(T)$ of each algorithm as a function of ϵ for $T = 1000$ in Figure 4.1.

As can be seen in Figure 4.1, our algorithm performs more robustly than others for different choices of noise ϵ. In particular, we set ϵ(i.e., setting s) as $\{0, 0.2, 0.4, 0.6, 0.8, 1\}$, respectively, and other parameters remain, and detailed graphs for each ϵ is in the Appendix.

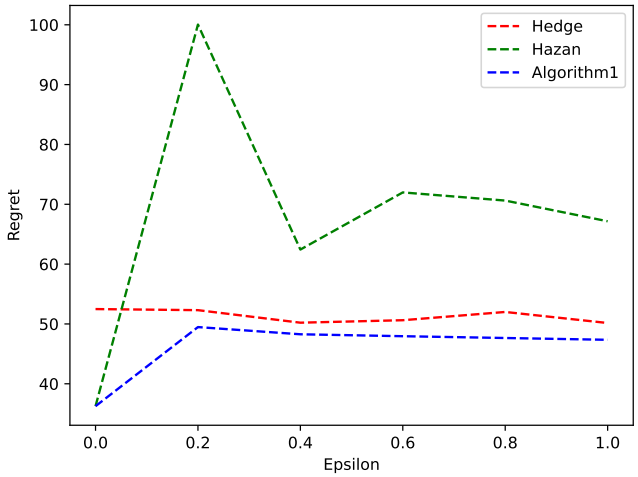
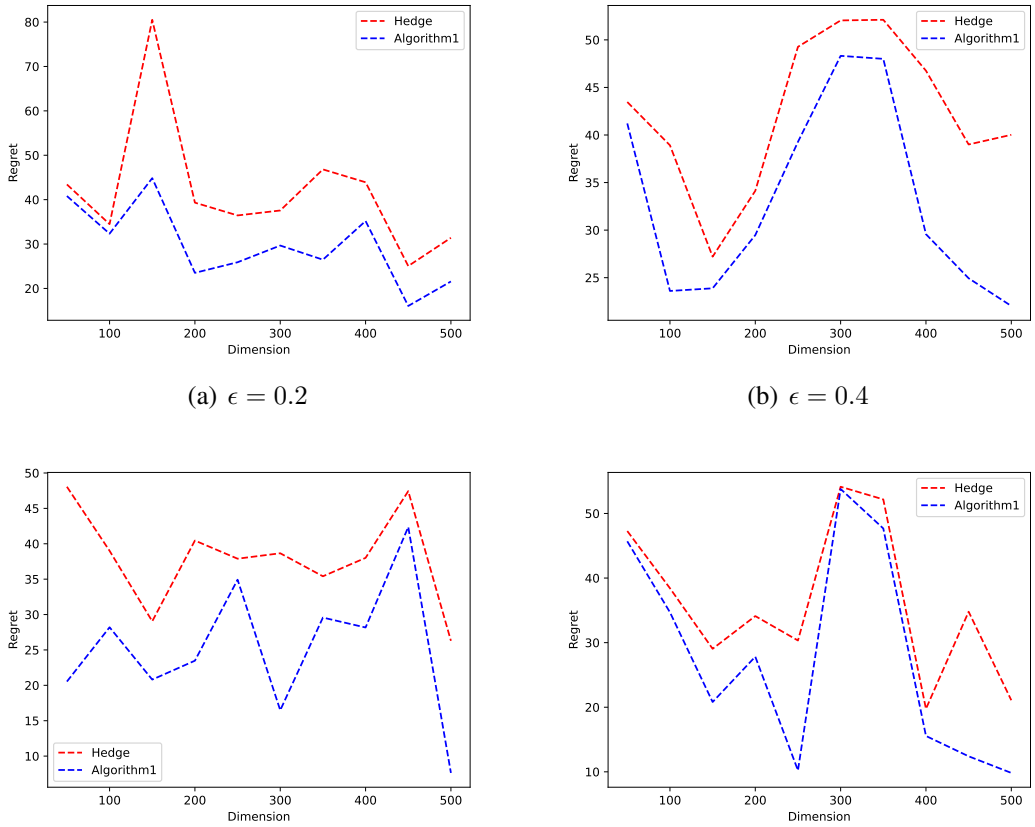


Figure 4.1: $\text{Regret}_{\mathcal{A}}(T)$ as a function of ϵ for $T = 1000$

Secondly, in order to see the results based on the different dimensions, we construct the experiments and set $N = 50, 100, 150, ..., 500$, and $\epsilon = 0.2, 0.4, 0.6, 0.8$, since the unsatisfying performance of Hazan's algorithm shown in Figure 4.1, and rapidly increased running time with respect to dimension, we discard it to make sure the experiments can be done in an acceptable amount of time.



(a) $\epsilon = 0.2$ (b) $\epsilon = 0.4$

(c) $\epsilon = 0.6$ (d) $\epsilon = 0.8$

Figure 4.2: Results for different choices of N and ϵ

4.5 Concluding remarks

In this paper, we construct an algorithm for expert advice with noisy low rank loss. This algorithm is designed for the problem where the algorithm obtains no prior information about the low rank structure but only the noise bound ϵ. Theoretically, we achieve a regret bound as $O(\sqrt{T}(d + d^{4/3}(N\epsilon)^{1/3}))$, however, in the experiments, our algorithm performs better even if $\epsilon \geq \Omega\left(\frac{1}{N}\right)$, which indicates that there might be a gap between our bound and the optimal one.

4.6 Appendix

4.6.1 Appendix A. Necessary Lemmata

Lemma 5 ([6]). *Let* $\boldsymbol{W} = \operatorname{span}\{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_{k-1}\}$, $\boldsymbol{U} = \operatorname{span}\{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_{k-1}, \boldsymbol{u}\}$, *and* $\widetilde{\boldsymbol{U}} = \operatorname{span}\{\boldsymbol{w}_1, \cdots, \boldsymbol{w}_{k-1}, \tilde{\boldsymbol{u}}\}$ *be subspaces spanned by vectors in* $\mathbb{R}^m$. *Then*

$$\theta(\boldsymbol{U}, \widetilde{\boldsymbol{U}}) \leq \frac{\pi}{2} \frac{\theta(\tilde{\boldsymbol{u}}, \boldsymbol{u})}{\theta(\tilde{\boldsymbol{u}}, \boldsymbol{W})}$$

Lemma 6 ([29] Lemma 11). *Let* $M \in \mathbb{R}^{k \times k}$, $U \in \mathbb{R}^{N \times d}$ *such that* $M \succ 0$ *and* U. *Then*

$$U^T (UMU^T)^+ U = M^{-1}. \tag{4.35}$$

Lemma 7. *For any* $\boldsymbol{l}_t \in \mathbb{R}^N$, *and two sub-spaces* $\widetilde{\boldsymbol{U}}^k, \boldsymbol{U}^k \subseteq \mathbb{R}^N$. *We have that*

$$\theta(\boldsymbol{l}_t, \boldsymbol{U}^k) \geq \theta(\boldsymbol{l}_t, \widetilde{\boldsymbol{U}}^k) - \theta(\boldsymbol{U}^k, \widetilde{\boldsymbol{U}}^k). \tag{4.36}$$

Proof. Naturally we obtain that $\theta(l_t, \tilde{x}) \leq \theta(x, \tilde{x}) + \theta(l_t, x), \forall x, \tilde{x}$.

Given $\tilde{x}^* = \arg\min_{\tilde{x} \in \widetilde{U}^k} \theta(x, \tilde{x})$, then we have that

$$\min_{\tilde{x} \in \widetilde{U}^k} \theta(l_t, \tilde{x}) \leq \theta(l_t, \tilde{x}^*) \leq \theta(x, \tilde{x}^*) + \theta(l_t, x) \tag{4.37}$$

Rearranging the terms we obtain that

$$\min_{\tilde{x} \in \widetilde{U}^k} \theta(l_t, \tilde{x}) - \min_{\tilde{x} \in \widetilde{U}^k} \theta(x, \tilde{x}) \leq \theta(l_t, x) \quad \forall x. \tag{4.38}$$

Since the definition we have that $\max_{x \in U^k} \min_{\tilde{x} \in \widetilde{U}^k} \theta(x, \tilde{x}) = \theta(U^k, \widetilde{U}^k)$, we have

$$\min_{\tilde{x} \in \widetilde{U}^k} \theta(l_t, \tilde{x}) - \theta(U^k, \widetilde{U}^k) \leq \min_{\tilde{x} \in \widetilde{U}^k} \theta(l_t, \tilde{x}) - \min_{\tilde{x} \in \widetilde{U}^k} \theta(x, \tilde{x}) \leq \theta(l_t, x) \quad \forall x. \tag{4.39}$$

At last, setting $x = \arg\min_{x \in U^k} \theta(l_t, x)$, we have that

$$\theta(l_t, \widetilde{U}^k) - \theta(U^k, \widetilde{U}^k) = \min_{\tilde{x} \in \widetilde{U}^k} \theta(l_t, \tilde{x}) - \theta(U^k, \widetilde{U}^k) \leq \min_{x \in U^k} \theta(l_t, x) = \theta(l_t, U^k). \tag{4.40}$$

□

Remark 3. *For any $\boldsymbol{l}_t \in \boldsymbol{U}^k \subseteq \mathbb{R}^N$ and two spaces $\boldsymbol{U}_0^k, \widetilde{\boldsymbol{U}}^k \subseteq \mathbb{R}^N$. We have*

$$\theta(\boldsymbol{U}^k, \boldsymbol{U}_0^k) + \theta(\boldsymbol{U}_0^k, \widetilde{\boldsymbol{U}}^k) \geq \theta(\boldsymbol{U}^k, \widetilde{\boldsymbol{U}}^k). \tag{4.41}$$

Proof. According to Lemma 7 we have that

$$\theta(l_t, U_0^k) \geq \theta(l_t, \widetilde{U}^k) - \theta(U_0^k, \widetilde{U}^k). \tag{4.42}$$

Rearranging the equation we have that

$$\theta(l_t, U_0^k) + \theta(U_0^k, \widetilde{U}^k) \geq \theta(l_t, \widetilde{U}^k). \tag{4.43}$$

Letting $l_t' = \arg\max_{l_t \in U^k} \theta(l_t, \widetilde{U}^k)$, we obtain that $\theta(l_t', U_0^k) + \theta(U_0^k, \widetilde{U}^k) \geq \theta(U^k, \widetilde{U}^k)$. Since $\theta(U^k, U_0^k) = \max_{l_t \in U^k} \theta(l_t, U_0^k) \geq \theta(l_t', U_0^k)$, we have our conclusion that

$$\theta(U^k, U_0^k) + \theta(U_0^k, \widetilde{U}^k) \geq \theta(U^k, \widetilde{U}^k). \tag{4.44}$$

□

Lemma 8. *For any $k \geq 2$, we have that*

$$\left(\frac{k}{k-1}\right)^{2/3} \left(\pi \cdot \left(\frac{k-1}{k}\right)^{1/3} + 10(k-1)\right) \leq 10k. \tag{4.45}$$

Proof. First we have that:

$$\begin{aligned}
&\left(\frac{k}{k-1}\right)^{2/3} \left(\pi \cdot \left(\frac{k-1}{k}\right)^{1/3} + 10(k-1)\right) \leq 10k \\
&\Leftrightarrow \left(\frac{k}{k-1}\right)^{1/3} \pi + 10k^{2/3}(k-1)^{1/3} \leq 10k \\
&\Leftrightarrow \pi \leq (10k - 10k^{2/3}(k-1)^{1/3}) \cdot \left(\frac{k-1}{k}\right)^{1/3} \\
&\Leftrightarrow \pi \leq 10k^{2/3}(k^{1/3} - (k-1)^{1/3}) \cdot k^{-1/3} \cdot (k-1)^{1/3} \\
&\Leftrightarrow \pi \leq 10k^{1/3}(k-1)^{1/3}(k^{1/3} - (k-1)^{1/3}).
\end{aligned}$$

We set that $g(k) = k^{1/3}(k-1)^{1/3}(k^{1/3} - (k-1)^{1/3})$, then we have that

$$g^{'}(k) = \frac{-3k(k-1)^{1/3} + 3k^{4/3} - 2k^{1/3} + (k-1)^{1/3}}{3k^{2/3}(k-1)^{2/3}}. \tag{4.46}$$

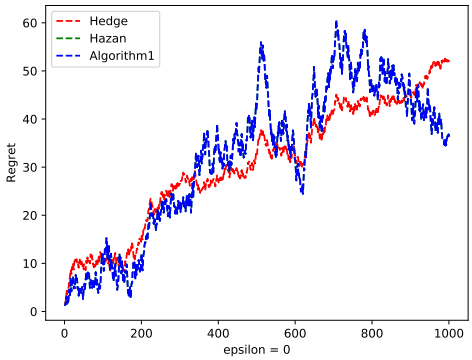
Next we can show that

$$
\begin{aligned}
&-3k(k-1)^{1/3}+3k^{4/3}-2k^{1/3}+(k-1)^{1/3}>0\\
&\Leftrightarrow k^{1/3}(3x-1)>(k-1)^{1/3}(3x-1)\\
&\Leftrightarrow k(3x-2)^3>(k-1)(3k-1)^3\\
&\Leftrightarrow 2k-1>0.
\end{aligned}
$$

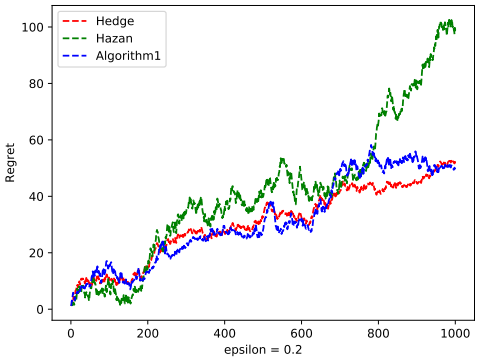
Above equation implies that $\forall k \geq 2$, then we have $g^{'}(k) \geq 0$, and $g(k)$ is an increasing function. Meanwhile since $\pi \leq 10 \cdot g(2)$, we obtain our conclusion. □

4.6.2 Appendix B. Detailed experimental results for Figure 4.1

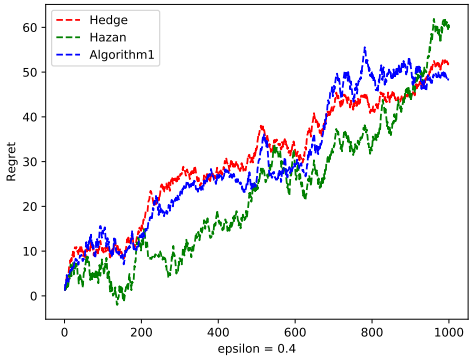
In the following pictures we show the details for the experiment for Figure 4.1.



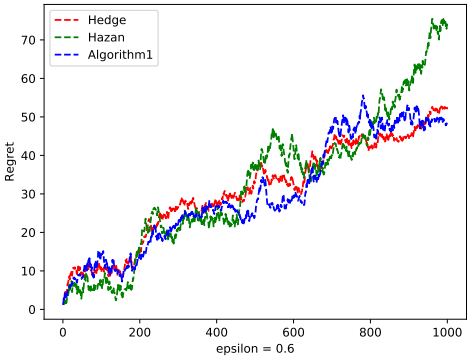
(a) $\epsilon = 0$



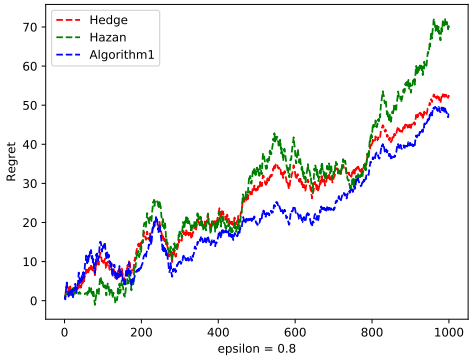
(b) $\epsilon = 0.2$



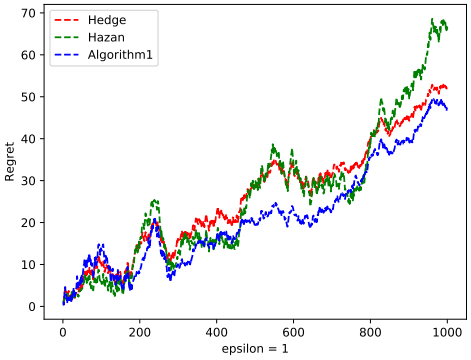
(c) $\epsilon = 0.4$



(d) $\epsilon = 0.6$



(e) $\epsilon = 0.8$



(f) $\epsilon = 1$

Figure 4.3: Results for different choices of ϵ when $N = 300$ and $d = 2$

Chapter 5

An online semi-definition programming with a generalised log-determinant regularizer and its applications

5.1 Introduction

Online binary matrix completion (OBMC) is a natural formulation of online matrix completion, extensively studied in machine learning community [30, 31, 52, 9]. Intuitively, the OBMC problem is to predict a given entry of an unknown $m \times n$ binary matrix. More precisely, the problem is formulated as a repeated game between the algorithm and the adversarial environment as described below: On each round t, (i) the environment presents an entry $(i_t, j_t) \in [m] \times [n]$, (ii) the algorithm predicts $\hat{y}_t \in \{-1, 1\}$, and then (iii) the environment reveals the true value $y_t \in \{-1, 1\}$. The goal of the algorithm is to minimise the number of mistakes $\sum_{t=1}^{T} \mathbb{I}_{\hat{y}_t \neq y_t}$.

Recently, Herbster et al. generalise the problem by considering side information available [31]. The side information brings some information about the target matrix, or more generally, about a comparator matrix $\boldsymbol{U}$ that is hopefully a good approximation to the target matrix. To be more specific, assume that $\boldsymbol{U} = \mathbb{R}^{m \times n}$ can be factorized into $\boldsymbol{U} = \boldsymbol{P}\boldsymbol{Q}^\top$ for some matrices $\boldsymbol{P} \in \mathbb{R}^{n \times d}$ and $\boldsymbol{Q} \in \mathbb{R}^{m \times d}$ for some $d \geq 1$ such that $\|\boldsymbol{P}_i\| = \|\boldsymbol{Q}_j\| = 1$ for all i and j, where $\boldsymbol{P}_i$ is the i-th row vector of $\boldsymbol{P}$ (interpreted as a linear classifier associated with row i of $\boldsymbol{U}$) and $\boldsymbol{Q}_j$ is the j-th row vector of $\boldsymbol{Q}$ (interpreted as a feature vector associated with column j of $\boldsymbol{U}$). In other words, $z_t = y_t U_{i_t, j_t}$ can be viewed as the margin of the labeled instance $(\boldsymbol{Q}_{j_t}, y_t)$ with respect to a hyperplane $\boldsymbol{P}_{i_t}$, from which we can define the hinge loss as $[1 - z_t/\gamma]_+$ for a given margin parameter $\gamma > 0$, where $[x]_+$ is x if $x > 0$ and 0 otherwise.

Note that the hinge loss represents the quality of predictiveness of the comparator matrix $\boldsymbol{U}$. Moreover, side information is formally represented as a pair of symmetric and positive definite matrices $\boldsymbol{M} \in \mathbb{R}^{m\times m}$ and $\boldsymbol{N} \in \mathbb{R}^{n\times n}$, and its quality is measured by the sum of trace norms $\mathcal{D} = \mathrm{Tr}(\boldsymbol{P}^\top \boldsymbol{M}\boldsymbol{P}) + \mathrm{Tr}(\boldsymbol{Q}^\top \boldsymbol{N}\boldsymbol{Q})$. Note that Herbster et al. introduce a notion of *quasi-dimension* of a comparator $\boldsymbol{U}$, defined as the minimum of $\mathcal{D}$ over all factorizations $\boldsymbol{P}$ and $\boldsymbol{Q}$ such that $\boldsymbol{U} = \gamma \boldsymbol{P}\boldsymbol{Q}^\top$. But it turns out that we do not need the notion in this paper. Then, they prove a mistake bound given by the total hinge loss of $\boldsymbol{U}$ with an additional term expressed in terms of γ, m, n, and $\mathcal{D}$. In particular, for the *realizable case* where the total hinge loss of $\boldsymbol{U}$ is zero, the bound is of the form $O(\mathcal{D}\ln(m+n)/\gamma^2)$. They consider a simple realizable case where $\boldsymbol{U}$ has a (k,l)-biclustered structure (see Appendix for details) and some information about the structure is given to the algorithm as the side information. Then, they show that the mistake bounds becomes $O(kl\ln(m+n))$. Unfortunately, however, there still remains a logarithmic gap from a lower bound of $\Omega(kl)$ [30].

In this paper, we obtain a mistake bound of $O(\mathcal{D}/\gamma^2)$ in the realizable case, which improves the bound of Herbster et al.'s by a logarithmic factor and thus implies an optimal $O(kl)$ mistake bounds when $\boldsymbol{U}$ has a (k,l)-biclustered structure. The basic idea is to reduce the OBMC problem with side information to a variant of an online semi-definite programming (OSDP) problem, where the loss matrices are sparse and the decision space consists of symmetric and positive semi-definite matrices $\boldsymbol{W}$ such that its $\boldsymbol{\Gamma}$-trace norm $\mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{W}\boldsymbol{\Gamma})$ and diagonal entries $W_{i,i}$ are both bounded, where $\boldsymbol{\Gamma}$ is a symmetric and positive definite matrix transformed from the side information $(\boldsymbol{M}, \boldsymbol{N})$ through our reduction. Note that the standard OSDP problems studied in the literature correspond to the case where $\boldsymbol{\Gamma} = \boldsymbol{E}$. Then we employ a standard follow-the-regularised-leader (FTRL) framework (see, e.g., [13, 48, 26]) for designing and analyzing our algorithm for the generalized OSDP problem. Note that to obtain a good algorithm we choose a specialized regulariser as stated later.

The FTRL approach to solving the standard OSDP problems have been widely utilised for various problems of online matrix prediction, such as online gambling [2, 28], online collaborative filtering [49, 14, 34], online similarity prediction [22], and especially a *non-binary version* of online matrix completion with no side information [28, 44]. Note that for these problems the performance of the algorithm is now measured by the *regret*, defined as the cumulative loss of the algorithm minus the cumulative loss of the best fixed comparator matrix in hindsight. Let us briefly review the last-mentioned results about non-binary online matrix completion with no side information. In the seminal paper of Hazan et al. [28], they first propose a reduction

from the problem to a standard OSDP problem, which is similar to but quite different from our reduction presented in this paper, and then they give an FTRL-based algorithm with an *entropic regularizer* for the reduced OSDP problem, resulting in a sub-optimal regret bound though. On the other hand, Moridomi et al. [44] observe that the loss matrices obtained in the reduction are sparse, and by following the result of [15] they find out that the *log-determinant regularizer* performs better, resulting in a better regret bound.

Now let us return to the OBMC problem with side information. It seems that Herbster et al. [31] implicitly reduce the problem to another variant of OSDP problem where the decision space is less restricted than ours, and employ an FTRL-based algorithm with an entropic regularizer for the reduced OSDP problem. Note that they do not give a regret analysis for their OSDP problem in a general form but give it only for the particular OSDP problem instance obtained from the reduction. We believe that the sub-optimality of their mistake bound is mainly due to the choice of entropic regularizer. On the other hand, our reduction yields sparse loss matrices and thus it is highly expected that the log-determinant regularizer performs better.

For our OSDP problem, we first examine a standard log-determinant regularizer $R(\boldsymbol{W}) = -\ln\det(\boldsymbol{W} + \epsilon\boldsymbol{E})$, but we have not succeeded to obtain a good regret bound. Next we try a natural and apparently straightforward reduction to a standard OSDP problem, for which a regret bound is known, and derive a regret bound for our OSDP problem from the known bound. Unfortunately, as seen in the later section, this approach also fails. This is due to the fact that the reduction does not preserve the sparsity of loss matrices and the bound of the diagonal entries of decision matrices. Finally we try a specialized log-determinant regularizer $R(\boldsymbol{W}) = -\ln\det(\boldsymbol{\Gamma}\boldsymbol{W}\boldsymbol{\Gamma} + \epsilon\boldsymbol{E})$ and succeed to derive a better regret bound. Therefore, we not only demonstrate the power of log-determinant regularizer, which has not been well explored as the standard entropic or Frobenius-norm regularizer; but also suggest to use the appropriate regularizer depending on side information as well as on the decision space. Note that to derive the bound we carefully follow the analysis of Moridomi et al. [44] with non-trivial generalizations.

Furthermore, we apply our online algorithm in the statistical (batch) learning setting by the standard online-to-batch conversion framework (see, e.g., Mohri et al. [42]) and derive a generalization error bound with side information. Our generalised error bound is similar to the known margin-based bound of SVMs (e.g.,Mohri et al. [42]) with the best kernel when the side information is vacuous. It is remarkable that we can not only obtain such a bound without knowing the best kernel, but also implies that the error bound in batch learning setting can be improved when the side information is given to the learner in advance.

Our main contribution is summarised as follows:

1. Firstly, we establish a generalized OSDP problem parameterized by a symmetric and positive definite matrix $\mathbf{\Gamma}$, and give an FTRL-based algorithm with a specialized log-determinant regularizer with a regret bound. Note that our result recovers the previously known bound [44] in the case where $\mathbf{\Gamma}$ is the identity matrix.

2. We apply the result above to the online OBMC problem with side information and the online similarity prediction with side information, and improve the previously known mistake bounds by logarithmic factors for the both problems. In particular, for the former problem, our mistake bound is optimal.

3. We give a conversion from online learning setting to the batch setting. On the one hand, our error bound recovers the best margin-based bound when the side information is vacuous, on the other hand, our bound implies an improvement to the batch setting with extra side information.

This paper is organized as follows. In section 2, we formally describe the problem formulation of the generalised OSDP and give a naive reduction to the standard OSDP, which yields a worse regret bound. The main algorithm with its regret bound for the generalised OSDP is given in section 3. In section 4 we apply our algorithm to the OBMC problem with side information and give a mistake bound. Moreover, we show that the mistake bound is optimal in the realizable case where the comparator matrix has a biclustered structure. In section 5, we derive the batch setting to the OBMC problem with side information. In the appendix A1, we describe some of proofs for our main proposition. Further, we give the definition and application to the (k, l)-biclustered structural comparator matrix in the OMBC problem, the online similarity problem with side information, necessary Lemmata, and proofs in appendix A2, and A3.

5.2 Preliminaries

For a positive integer N, let $[N]$ denote the set $\{1, 2, \ldots, N\}$. Let $\mathbb{S}^{N\times N}$, $\mathbb{S}_{+}^{N\times N}$ and $\mathbb{S}_{++}^{N\times N}$ denote the sets of $N \times N$ symmetric matrices, symmetric positive semi-definite matrices and symmetric strictly positive definite matrices, respectively. We define $\boldsymbol{E}$ as the identity matrix. For an $m \times n$ matrix $\boldsymbol{X} \in \mathbb{R}^{m\times n}$ and $(i, j) \in [m] \times [n]$, let $\boldsymbol{X}_i$, $\boldsymbol{X}_{i,j}$ and $\mathrm{vec}(\boldsymbol{X})$ denote the i-th row vector of $\boldsymbol{X}$, the (i, j) entry of $\boldsymbol{X}$, and the vector of mn dimension obtained by arranging

all entries $\boldsymbol{X}_{i,j}$ of $\boldsymbol{X}$ in some order. For matrices $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{R}^{m\times n}$, $\boldsymbol{X} \bullet \boldsymbol{Y} = \mathrm{Tr}(\boldsymbol{X}^\top \boldsymbol{Y}) = \mathrm{vec}(\boldsymbol{X})^\top \mathrm{vec}(\boldsymbol{Y})$ denotes the Frobenius inner product of them. For $\boldsymbol{X} \in \mathbb{S}_+^{N\times N}$, we denote by $\mathrm{Tr}(\boldsymbol{X}) = \sum_{i=1}^N |\lambda_i(\boldsymbol{X})| = \sum_{i=1}^N \boldsymbol{X}_{i,i}$ the trace norm of $\boldsymbol{X}$, where $\lambda_i(\boldsymbol{X})$ is the i-th largest eigenvalue of $\boldsymbol{X}$. Furthermore, for $\boldsymbol{\Gamma} \in \mathbb{S}_{++}^{N\times N}$, we define the $\boldsymbol{\Gamma}$-trace norm of $\boldsymbol{X}$ as $\mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\Gamma})$. For a vector $\boldsymbol{x}$, the p-norm $\boldsymbol{x}$ is denoted by $\|\boldsymbol{x}\|_p$.

5.2.1 Generalised OSDP problem with bounded Γ-trace norm

Our generalised OSDP problem with respect to a matrix $\boldsymbol{\Gamma} \in \mathbb{S}_{++}^{N\times N}$ is specified by a pair $(\mathcal{K}, \mathcal{L})$, where

$$\mathcal{K} = \{\boldsymbol{W} \in \mathbb{S}_+^{N\times N} : \mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{W}\boldsymbol{\Gamma}) \le \tau, \forall i \in [N], |\boldsymbol{W}_{i,i}| \le \beta\} \tag{5.1}$$

is called the decision space, and

$$\mathcal{L} = \{\boldsymbol{L} \in \mathbb{S}^{N\times N} : \|\mathrm{vec}(\boldsymbol{L})\|_1 \le g\} \tag{5.2}$$

is called the loss space, where $\tau > 0$, $\beta > 0$ and $g > 0$ are parameters. The generalised OSDP problem $(\mathcal{K}, \mathcal{L})$ is a repeated game between the algorithm and the adversary as described below: On each round $t \in [T]$,

1. The algorithm chooses a matrix $\boldsymbol{W}_t \in \mathcal{K}$,
2. The adversary gives a loss matrix $\boldsymbol{L}_t \in \mathcal{L}$, and
3. The algorithm incurs a loss given by $\boldsymbol{W}_t \bullet \boldsymbol{L}_t$.

The goal of the algorithm is to minimise the following regret

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) = \sum_{t=1}^T \boldsymbol{W}_t \bullet \boldsymbol{L}_t - \min_{\boldsymbol{W}\in\mathcal{K}} \sum_{t=1}^T \boldsymbol{W} \bullet \boldsymbol{L}_t. \tag{5.3}$$

Note that the standard OSDP problem corresponds to the special case where $\boldsymbol{\Gamma} = \boldsymbol{E}$.

Since the decision space is convex and the loss function is linear, the problem is categorized in online linear optimization and thus we can employ a standard FTRL algorithm, as Moridomi et al. [44] did for the standard OSDP problem. Given a convex function $R : \mathcal{K} \to \mathbb{R}$ as the regularizer, the FTRL algorithm produces a matrix $\boldsymbol{W}_t$ in each round t according to

$$\boldsymbol{W}_t = \operatorname*{arg\,min}_{\boldsymbol{W}\in\mathcal{K}} \left(R(\boldsymbol{W}) + \eta \sum_{s=1}^{t-1} \boldsymbol{L}_s \bullet \boldsymbol{W} \right). \tag{5.4}$$

In particular, Moridomi et al. choose the log-determinant regularizer defined as

$$R(\boldsymbol{W}) = -\ln\det(\boldsymbol{W} + \epsilon\boldsymbol{E}), \tag{5.5}$$

where $\epsilon > 0$ is a parameter and derive the following regret bound for the standard OSDP problem.

Theorem 11 ([44])**.** *For the standard OSDP problem $(\mathcal{K}, \mathcal{L})$ with $\boldsymbol{\Gamma} = \boldsymbol{E}$, The FTRL algorithm with the log-determinant regularizer achieves*

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) = O(g\sqrt{\tau\beta T}). \tag{5.6}$$

5.2.2 A naive reduction

There is a natural reduction to a standard OSDP problem $(\widetilde{\mathcal{K}}, \widetilde{\mathcal{L}})$ where

$$\widetilde{\mathcal{K}} = \{\boldsymbol{W} \in \mathbb{S}_+^{N\times N} : \mathrm{Tr}(\boldsymbol{W}) \le \tau, \forall i \in [N], W_{i,i} \le \beta'\}, \quad \widetilde{\mathcal{L}} = \{\boldsymbol{L} \in \mathbb{S}^{N\times N} : \|\mathrm{vec}(\boldsymbol{L})\|_1 \le g'\}$$

for some parameters $\beta' > 0$ and $g' > 0$. The reduction consists of two transformations: One is to transform the decision matrix $\widetilde{\boldsymbol{W}}_t \in \widetilde{\mathcal{K}}$ produced from an algorithm for the standard OSDP problem to the decision matrix $\boldsymbol{W}_t = \boldsymbol{\Gamma}^{-1}\widetilde{\boldsymbol{W}}_t\boldsymbol{\Gamma}^{-1}$ and the other is to transform the loss matrices $\boldsymbol{L}_t \in \mathcal{L}$ chosen by the adversary to $\widetilde{\boldsymbol{L}}_t = \boldsymbol{\Gamma}^{-1}\boldsymbol{L}_t\boldsymbol{\Gamma}^{-1}$, which is fed to the algorithm for the standard OSDP problem. Note that the loss is preserved under this reduction, that is, $\boldsymbol{W}_t \bullet \boldsymbol{L}_t = \mathrm{Tr}(\boldsymbol{W}_t\boldsymbol{L}_t) = \mathrm{Tr}(\boldsymbol{\Gamma}^{-1}\widetilde{\boldsymbol{W}}_t\boldsymbol{\Gamma}^{-1}\boldsymbol{\Gamma}\widetilde{\boldsymbol{L}}_t\boldsymbol{\Gamma}) = \mathrm{Tr}(\widetilde{\boldsymbol{W}}_t\widetilde{\boldsymbol{L}}_t) = \widetilde{\boldsymbol{W}}_t \bullet \widetilde{\boldsymbol{L}}_t$. Moreover, the $\boldsymbol{\Gamma}$-trace norm of $\boldsymbol{W}_t$ is the trace norm of $\widetilde{\boldsymbol{W}}_t$, i.e., $\mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{W}_t\boldsymbol{\Gamma}) = \mathrm{Tr}(\widetilde{\boldsymbol{W}}_t)$. Therefore, if β' and g' are large enough so that $\boldsymbol{\Gamma}\boldsymbol{W}\boldsymbol{\Gamma}_{i,i} \le \beta'$ for any $\boldsymbol{W} \in \mathcal{K}$ and $\|\mathrm{vec}(\boldsymbol{\Gamma}^{-1}\boldsymbol{L}\boldsymbol{\Gamma}^{-1})\|_1 \le g'$ for any $\boldsymbol{L} \in \mathcal{L}$, we have that $\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) \le \mathrm{Regret}_{\mathrm{OSDP}}(T, \widetilde{\mathcal{K}}, \widetilde{\mathcal{L}})$. Moreover, using the FTRL algorithm with the log-determinant regularizer for the standard OSDP problem, we immediately have

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) = O(g'\sqrt{\tau\beta' T}).$$

by Theorem 11.

In the following part, we give lower bounds on β' and g' by showing an example, which implies that the above reduction yields a worse regret bound.

Example 1. *Define* $\mathbf{\Gamma} \in \mathbb{S}_{++}^{N\times N}$ *as*

$$\mathbf{\Gamma} = \begin{bmatrix} N & -1 & \cdots & -1 \\ -1 & N & \cdots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \cdots & N \end{bmatrix} \quad \text{with} \quad \mathbf{\Gamma}^{-1} = \begin{bmatrix} \frac{2}{N+1} & \frac{1}{N+1} & \cdots & \frac{1}{N+1} \\ \frac{1}{N+1} & \frac{2}{N+1} & \cdots & \frac{1}{N+1} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{N+1} & \frac{1}{N+1} & \cdots & \frac{2}{N+1} \end{bmatrix}$$

and let $\tau = N^3 + N^2 - N$, $\beta = 1$ *and* $g = 1$ *so that* $\boldsymbol{E} \in \mathcal{K}$ *and* $\boldsymbol{L} \in \mathcal{L}$ *with* $L_{i,j} = 1$ *if* $(i, j) = (1, 1)$ *and 0 otherwise.*

Then, with a simple calculation we get $|\mathbf{\Gamma}\boldsymbol{E}\mathbf{\Gamma}|_{i,i} = N^2 + N - 1$ *for all* $i \in [N]$ *and* $\|\mathrm{vec}(\mathbf{\Gamma}^{-1}\boldsymbol{L}\mathbf{\Gamma}^{-1})\|_1 = 1$, *which implies that we need* $\beta' \geq N^2 + N - 1$ *and* $g' \geq 1$. *In other words, the regret bound obtained by the naive reduction above is not smaller than the order of* $N\sqrt{\tau T}$. *On the other hand, using our algorithm described in the next section, we have a regret bound of* $O(\sqrt{\tau T})$ *for this example problem, which comes from* $\rho = \max_{i,j} |(\mathbf{\Gamma}^{-1}\mathbf{\Gamma}^{-1})_{i,j}| \leq 1$. *So our algorithm is significantly better than the naive reduction method.*

5.3 Algorithm for the generalised OSDP problem

Throughout this section, we consider the generalised OSDP problem $(\mathcal{K}, \mathcal{L})$ specified by (5.1) and (5.2). for some $\mathbf{\Gamma} \in \mathbb{S}_{++}^{N\times N}$, and parameters $\tau > 0$, $\beta > 0$ and $g > 0$. We use the FTRL algorithm (5.4) with the following regularizer.

$$R(\boldsymbol{W}) = -\ln\det(\mathbf{\Gamma}\boldsymbol{W}\mathbf{\Gamma} + \epsilon\boldsymbol{E}), \tag{5.7}$$

which we call the $\mathbf{\Gamma}$-*calibrated log-determinant regularizer*, where $\epsilon > 0$ is a parameter. The next theorem gives a regret bound of our algorithm.

Theorem 12 (Main Theorem)**.** *Let* $\rho = \max_{i,j} |(\mathbf{\Gamma}^{-1}\mathbf{\Gamma}^{-1})_{i,j}|$. *Then, the FTRL algorithm with the* $\mathbf{\Gamma}$*-calibrated log-determinant regularizer achieves*

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) = O\left(g^2(\beta + \rho\epsilon)^2 T\eta + \frac{\tau}{\epsilon\eta}\right).$$

In particular, letting $\eta = \sqrt{\frac{\tau}{g^2(\beta+\rho\epsilon)^2\epsilon T}}$ *and* $\epsilon = \beta/\rho$, *we have*

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) = O\left(g\sqrt{\beta\rho\tau T}\right). \tag{5.8}$$

Note that we can recover the same regret bound of Theorem 11 by letting $\mathbf{\Gamma} = \boldsymbol{E}$.

The proof is based on the analysis of strong convexity of our regularizer with respect to loss space.

Definition 5. *For a decision space $\mathcal{K}$ and a real number $s \geq 0$, a regularizer $R : \mathcal{K} \to \mathbb{R}$ is said to be s-strongly convex with respect to the loss space $\mathcal{L}$ if for any $\alpha \in [0, 1]$, any $\boldsymbol{X}, \boldsymbol{Y} \in \mathcal{K}$ and any $\boldsymbol{L} \in \mathcal{L}$, the following holds*

$$R(\alpha \boldsymbol{X} + (1-\alpha)\boldsymbol{Y}) \leq \alpha R(\boldsymbol{X}) + (1-\alpha)R(\boldsymbol{Y}) - \frac{s}{2}\alpha(1-\alpha)|\boldsymbol{L} \bullet (\boldsymbol{X} - \boldsymbol{Y})|^2. \tag{5.9}$$

This is equivalent to the following condition: for any $\boldsymbol{X}, \boldsymbol{Y} \in \mathcal{K}$ and $\boldsymbol{L} \in \mathcal{L}$,

$$R(\boldsymbol{X}) \geq R(\boldsymbol{Y}) + \nabla R(\boldsymbol{Y}) \bullet (\boldsymbol{X} - \boldsymbol{Y}) + \frac{s}{2}|\boldsymbol{L} \bullet (\boldsymbol{X} - \boldsymbol{Y})|^2. \tag{5.10}$$

Note that the notion of strong convexity defined above is quite different from the standard one: Usually, the strong convexity is defined with respect to some norm $\|\cdot\|$, but now it is defined with respect to the loss space. Moridomi et al. [44] give a regret bound of the FTRL with a strongly convex regularizer for any OSDP problem in a general form.

Lemma 9. *[44] Let $R : \mathcal{K} \to \mathbb{R}$ be an s-strongly convex regularizer with respect to a decision space $\mathcal{L}$ for a decision space $\mathcal{K}$. Then the FTRL with the regularizer R applied to $(\mathcal{K}, \mathcal{L})$ achieves*

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}) \leq \frac{H_0}{\eta} + \frac{\eta}{s}T, \tag{5.11}$$

where $H_0 = \max_{\boldsymbol{W}, \boldsymbol{W}' \in \mathcal{K}}(R(\boldsymbol{W}) - R(\boldsymbol{W}'))$.

Due to the lemma above, it suffices to analyze the strong convexity of our $\mathbf{\Gamma}$-calibrated log-determinant regularizer with respect to our loss space (5.2). We give the result in the next proposition.

Proposition 6 (Main proposition)**.** *The $\mathbf{\Gamma}$-calibrated log-determinant regularizer $R(W) = -\ln\det(\mathbf{\Gamma}\boldsymbol{W}\mathbf{\Gamma} + \epsilon\boldsymbol{E})$ is s-strongly convex with respect to $\mathcal{L}$ for $\mathcal{K}$ with $s = 1/(1152\sqrt{e}(\beta + \rho\epsilon)^2 g^2)$, where $\rho = \max_{i,j}|(\mathbf{\Gamma}^{-1}\mathbf{\Gamma}^{-1})_{i,j}|$.*

The proof is given in Appendix A.

Proof sketch of Theorem 12: According to Lemma 9 and main proposition, we only need to bound H_0. With simple calculation we can bound $H_0 \leq \frac{\tau}{\epsilon}$ from the definition of R. A

detailed derivation is found in supplementary material. So the theorem follows. Note that the regret bound obtained is apparently irrelevant to the size of matrix N.

5.4 Application to OBMC with side information

In this section, we show that the OBMC with side information can be reduced to our OSDP problem $(\mathcal{K}, \mathcal{L})$. The reduction is twofold: Firstly reduce it to an online matrix prediction(OMP) problem with side information and then further reduce it to the generalised OSDP problem.

We first define the problem of OBMC with side information formally with some necessary notations.

5.4.1 The problem statement

We basically follow the problem statement by Herbster et al. [31] with some simplification.

Let m and n be natural numbers. Assume that matrices $\boldsymbol{M} \in \mathbb{S}_{++}^{m\times m}$ and $\boldsymbol{N} \in \mathbb{S}_{++}^{n\times n}$ are given to the algorithm. We call the pair $(\boldsymbol{M}, \boldsymbol{N})$ the side information.

The problem is a repeated game between the algorithm and the adversary, which is described as follows: On each round t,

1. the adversary presents $(i_t, j_t) \in [m] \times [n]$,
2. the algorithm produces $\hat{y}_t \in \{-1, +1\}$,
3. the adversary reveals $y_t \in \{-1, 1\}$.

The goal of the algorithm is to minimize the number of mistakes $M = \sum_{t=1}^{T} \mathbb{I}_{y_t \neq \hat{y}_t}$. In particular, we want to give a mistake bound in terms of the side information $(\boldsymbol{M}, \boldsymbol{N})$, so that the bound is small when the side information is useful in some sense.

Let the sequence from the adversary be denoted by $\mathcal{S} = ((i_1, j_1), y_1), \ldots, ((i_T, j_T), y_T) \subseteq ([m] \times [n] \times \{-1, 1\})^T$.

The problem can be interpreted as the prediction of given entries (i_t, j_t) of an unknown target matrix. But we do not assume the existence of such a matrix, that is, it can happen $y_t \neq y_{t'}$ even if $(i_t, j_t) = (i_{t'}, j_{t'})$.

To apply the FTRL framework to the problem, we consider a convex surrogate loss function, instead of 0-1 loss. In particular, we define the hinge loss function $h_\gamma : \mathbb{R} \to \mathbb{R}$ as

$$h_\gamma(x) = \begin{cases} 0 & \text{if } \gamma \leq x, \\ 1 - x/\gamma & \text{otherwise,} \end{cases}$$

for a given margin parameter $\gamma > 0$. Now we consider any matrices $\boldsymbol{P} \in \mathbb{R}^{m \times d}$ and $\boldsymbol{Q} \in \mathbb{R}^{n \times d}$ for some d so that $\boldsymbol{P}\boldsymbol{Q}^\top \in \mathbb{R}^{m \times n}$ can be interpreted as a *comparator matrix* for the sequence $\mathcal{S}$. We define the hinge loss of the sequence $\mathcal{S}$ with respect to the pair $(\boldsymbol{P}, \boldsymbol{Q})$ and γ as

$$\mathrm{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma) = \sum_{t=1}^{T} h_\gamma \left(\frac{y_t \boldsymbol{P}_{i_t} \boldsymbol{Q}_{j_t}^\top}{\|\boldsymbol{P}_{i_t}\|_2 \|\boldsymbol{Q}_{j_t}\|_2} \right). \tag{5.12}$$

The hinge loss measures how well the comparator matrix $\boldsymbol{P}\boldsymbol{Q}^\top$ predicts the true label y_t. In what follows, we assume without loss of generality that each row of $\boldsymbol{P}$ and $\boldsymbol{Q}$ is normalised, that is, $\|\boldsymbol{P}_i\|_2 = \|\boldsymbol{Q}_j\|_2 = 1$ for every $(i, j) \in [m] \times [n]$. Moreover, we sometimes call the pair $(\boldsymbol{P}, \boldsymbol{Q})$ as the comparator matrix.

Now we define the notion of the *quasi-dimension* of a comparator matrix which measures the usefulness of the side information. Specifically, the quasi-dimension of a comparator matrix $(\boldsymbol{P}, \boldsymbol{Q})$ with respect to the side information $(\boldsymbol{M}, \boldsymbol{N})$ is defined as

$$\mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{P}, \boldsymbol{Q}) = \alpha_{\boldsymbol{M}} \operatorname{Tr}\left(\boldsymbol{P}^\top \boldsymbol{M} \boldsymbol{P}\right) + \alpha_{\boldsymbol{N}} \operatorname{Tr}\left(\boldsymbol{Q}^\top \boldsymbol{N} \boldsymbol{Q}\right),$$

where $\alpha_{\boldsymbol{M}} = \max_{i \in [m]} (\boldsymbol{M}^{-1})_{i,i}$ and $\alpha_{\boldsymbol{N}} = \max_{j \in [n]} (\boldsymbol{N}^{-1})_{j,j}$. Note that if $\boldsymbol{M}$ and $\boldsymbol{N}$ are the identity matrices, then the quasi-dimension is $m + n$ for any comparator matrix, which corresponds to the case where the side information is vacuous. On the other hand, if the rows of $\boldsymbol{P}$ and/or the columns of $\boldsymbol{Q}$ are correlated and $\boldsymbol{M}$ and/or $\boldsymbol{N}$ capture the correlation well, then the quasi-dimension will be smaller.

Note that the notion of quasi-dimension is defined in a different way in [31].

5.4.2 Reduction from OBMC with side information to an online matrix prediction (OMP)

First we describe an OMP problem, to which our problem is reduced. The problem is specified by a decision space $\mathcal{X} \subseteq [-1, 1]^{m \times n}$ and a margin parameter $\gamma > 0$, and again it is formulated

as a repeated game: On each round $t \in [T]$,

1. the algorithm chooses a matrix $\boldsymbol{X}_t \in \mathbb{R}^{m \times n}$,
2. the adversary gives a triple $(i_t, j_t, y_t) \in [m] \times [n] \times \{-1, 1\}$, and
3. the algorithm suffers a loss given by $h_\gamma(y_t \boldsymbol{X}_{t,(i_t,j_t)})$.

The goal of the algorithm is to minimise the regret:

$$\mathrm{Regret}_{\mathrm{OMP}}(T, \mathcal{X}, \boldsymbol{X}^*) = \sum_{t=1}^{T} h_\gamma(y_t \boldsymbol{X}_{t,(i_t,j_t)}) - \min_{\boldsymbol{X}^* \in \mathcal{X}} \sum_{t=1}^{T} h_\gamma(y_t \boldsymbol{X}^*_{i_t,j_t}),$$

Note that unlike the standard setting of online prediction, we do not require $\boldsymbol{X}_t \in \mathcal{X}$.

For any matrix $\boldsymbol{A} \in \mathrm{R}^{k \times l}$, we define

$$\bar{\boldsymbol{A}} = \mathrm{diag}\left(\frac{1}{\|\boldsymbol{A}_1\|_2}, \cdots, \frac{1}{\|\boldsymbol{A}_k\|_2}\right) \boldsymbol{A}.$$

That is, $\bar{\boldsymbol{A}}$ is a matrix obtained from $\boldsymbol{A}$ by normalising all row vectors.

Below we show that the OBMC problem with side information $(\boldsymbol{M}, \boldsymbol{N})$ can be reduced to the OMP problem with the following decision space:

$$\mathcal{X} = \{\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top : \boldsymbol{P}\boldsymbol{Q}^\top \in \mathbb{R}^{m \times n}, \mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}(\bar{\boldsymbol{P}}, \bar{\boldsymbol{Q}}) \leq \widehat{\mathcal{D}}\},$$

where $\widehat{\mathcal{D}}$ is an arbitrary parameter. Below we give the reduction. Assume that we have an algorithm $\mathcal{A}$ for the OMP problem $(\mathcal{X}, \gamma)$.

Run the algorithm $\mathcal{A}$ and receive the first prediction matrix $\boldsymbol{X}_1$ from $\mathcal{A}$. Then, in each round $t \in [T]$,

1. observe an index pair $(i_t, j_t) \in [m] \times [n]$,
2. predict $\hat{y}_t = \mathrm{sgn}(\boldsymbol{X}_{t,(i_t,j_t)})$,
3. observe a true label $y_t \in \{-1, 1\}$,
4. if $\hat{y}_t = y_t$ then $\boldsymbol{X}_{t+1} = \boldsymbol{X}_t$, and if $\hat{y}_t \neq y_t$, then feed (i_t, j_t, y_t) to $\mathcal{A}$ to let it proceed and receive $\boldsymbol{X}_{t+1}$.

Note that we run the algorithm $\mathcal{A}$ in the mistake-driven manner, and hence $\mathcal{A}$ runs for $M = \sum_{t=1}^{T} \mathbb{I}_{\hat{y}_t \neq y_t}$ rounds, where M is the number of mistakes of the reduction algorithm above.

The next lemma shows the performance of the reduction.

Lemma 10. *Let* $\mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \boldsymbol{X}^*)$ *denote the regret of the algorithm* $\mathcal{A}$ *in the reduction above for a competitor matrix* $\boldsymbol{X}^* \in \mathcal{X}$*, where* $M = \sum_{t=1}^{T} \mathbb{I}(\hat{y}_t \neq y_t)$*. Then,*

$$\begin{aligned} M &\leq \inf_{\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^T \in \mathcal{X}} (\mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^T) + \mathrm{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma)) \\ &\leq \mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}) + \mathrm{hloss}(\mathcal{S}, \gamma), \end{aligned} \tag{5.13}$$

where we define

$$\mathrm{hloss}(\mathcal{S}, \gamma) = \min_{\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^T \in \mathcal{X}} \mathrm{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma). \tag{5.14}$$

Remark 4. *If* $\boldsymbol{M}$ *and* $\boldsymbol{N}$ *are identity matrices, then we have* $\mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}(\bar{\boldsymbol{P}}, \bar{\boldsymbol{Q}}) = m + n$*, and thus the decision space is an unconstrained set* $\mathcal{X} = \{\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top : \boldsymbol{P}\boldsymbol{Q}^\top \in \mathbb{R}^{m \times n}\}$*.*

Proof. Let $\boldsymbol{P}$ and $\boldsymbol{Q}$ be arbitrary matrices such that $\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top \in \mathcal{X}$. Since $\mathbb{I}(\mathrm{sgn}(x) \neq y) \leq h_\gamma(yx)$ for any $x \in \mathbb{R}$ and $y \in \{-1, 1\}$, we have

$$\begin{aligned} M &= \sum_{t=1}^{T} \mathbb{I}(\hat{y}_t \neq y_t) \leq \sum_{\{t:\hat{y}_t \neq y_t\}} h_\gamma(y_t \boldsymbol{X}_{t,(i_t,j_t)}) \\ &= \mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top) + \sum_{\{t:\hat{y}_t \neq y_t\}} h_\gamma(y_t(\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i_t,j_t}) \\ &\leq \mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top) + \sum_{t=1}^{T} h_\gamma(y_t(\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i_t,j_t}) \\ &= \mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top) + \mathrm{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma), \end{aligned}$$

where the second equality follows from the definition of regret, and the third equality follows from the fact that $(\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i,j} = \boldsymbol{P}_i\boldsymbol{Q}_j^\top/(\|\boldsymbol{P}_i\|_2\|\boldsymbol{Q}_j\|_2)$. Since the choice of $\boldsymbol{P}$ and $\boldsymbol{Q}$ is arbitrary, we get the first inequality of the lemma.

Now, let $\boldsymbol{P}$ and $\boldsymbol{Q}$ be the matrices that attain (5.14). Then, the inequality above implies that

$$M \leq \mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top) + \mathrm{hloss}(\mathcal{S}, \gamma) \leq \sup_{\boldsymbol{X}^* \in \mathcal{X}} \mathrm{Regret}_{\mathrm{OMP}}(M, \mathcal{X}, \boldsymbol{X}^*) + \mathrm{hloss}(\mathcal{S}, \gamma),$$

which proves the second inequality of the lemma.

□

5.4.3 Reduction from OMP to the generalised OSDP problem

A similar technique is used in [30] and [28]. For side information matrix $\boldsymbol{M}, \boldsymbol{N}$ we define a matrix $\boldsymbol{\Gamma}$ for our generalised OSDP as follows:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \sqrt{\alpha_{\boldsymbol{M}}}\boldsymbol{M} & 0 \\ 0 & \sqrt{\alpha_{\boldsymbol{N}}}\boldsymbol{N} \end{bmatrix}. \tag{5.15}$$

Next we define the decision space $\mathcal{K}$. Let $N = m + n$, and for any matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ such that $\boldsymbol{P}\boldsymbol{Q}^\top \in \mathbb{R}^{m\times n}$, we define

$$\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}} = \begin{bmatrix} \bar{\boldsymbol{P}} \\ \bar{\boldsymbol{Q}} \end{bmatrix} \begin{bmatrix} \bar{\boldsymbol{P}}^\top & \bar{\boldsymbol{Q}}^\top \end{bmatrix} = \begin{bmatrix} \bar{\boldsymbol{P}}\bar{\boldsymbol{P}}^\top & \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top \\ \bar{\boldsymbol{Q}}\bar{\boldsymbol{P}}^\top & \bar{\boldsymbol{Q}}\bar{\boldsymbol{Q}}^\top \end{bmatrix}.$$

Note that $\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}$ is an $N \times N$ symmetric and positive semi-definite matrix with its upper right $m \times n$ component matrix $\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top$ is a decision matrix for the OMP problem. So, intuitively, $\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}$ can be viewed as a positive semi-definite embedding of $\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top \in \mathcal{X}$. Next, we need to find a decision space as a convex set $\mathcal{K} \in \mathbb{S}_{++}^{N\times N}$ which satisfies

$$\mathcal{K} \supseteq \{\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}} : \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top \in \mathcal{X}\}.$$

Due to the following Lemma:

Lemma 11 (Lemma 8 [31])**.** *Given side information matrices $\boldsymbol{M}, \boldsymbol{N} \in \mathbb{S}_{++}^{N\times N}$, we define $\boldsymbol{\Gamma}$ as in Equation (5.15). Then we obtain that*

$$\mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}\boldsymbol{\Gamma}) = \alpha_{\boldsymbol{M}}\mathrm{Tr}\left(\bar{\boldsymbol{P}}^\top\boldsymbol{M}\bar{\boldsymbol{P}}\right) + \alpha_{\boldsymbol{N}}\mathrm{Tr}\left(\bar{\boldsymbol{Q}}^\top\boldsymbol{N}\bar{\boldsymbol{Q}}\right), \tag{5.16}$$

we can choose $\mathcal{K}$ as follows:

$$\mathcal{K} = \{\boldsymbol{W} \in \mathbb{S}_{++}^{N\times N} : \forall i \in [N], \boldsymbol{W}_{i,i} \leq 1 \wedge \mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{W}\boldsymbol{\Gamma}) \leq \widehat{\mathcal{D}}\} \supseteq \{\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}} : \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top \in \mathcal{X}\}. \tag{5.17}$$

Then, we define the loss matrix class $\mathcal{L}$. For any $(i, j) \in [m] \times [n]$, let $\boldsymbol{Z}\langle i, j\rangle \in \mathbb{S}_{+}^{N\times N}$ be a matrix such that the $(i, m + j)$-th and $(m + j, i)$-th components are 1 and the other components are 0. More formally,

$$\boldsymbol{Z}\langle i, j\rangle = \frac{1}{2}\left(\boldsymbol{e}_i\boldsymbol{e}_{m+j}^\top + \boldsymbol{e}_{m+j}\boldsymbol{e}_i^\top\right),$$

where $\boldsymbol{e}_k$ is the k-th basis vector of $\mathbb{R}^N$. Note that when we focus on its upper right $m \times n$

component matrix, then only the (i, j)-th component is 1. Then, $\mathcal{L}$ is

$$\mathcal{L} = \{c\boldsymbol{Z}\langle i, j\rangle : c \in \{-1/\gamma, 1/\gamma\}, i \in [m], j \in [n]\}. \tag{5.18}$$

Now we are ready to describe the reduction from the OMP problem for $\mathcal{X}$ to the OSDP problem $(\mathcal{K}, \mathcal{L})$. Let $\mathcal{A}$ be an algorithm for the OSDP problem.

Run the algorithm $\mathcal{A}$ and receive the first prediction matrix $\boldsymbol{W}_1 \in \mathcal{K}$ from $\mathcal{A}$.

In each round t,

1. let $\boldsymbol{X}_t$ be the upper right $m \times n$ component matrix of $\boldsymbol{W}_t$.
   // $\boldsymbol{X}_{t,(i,j)} = \boldsymbol{W}_t \bullet \boldsymbol{Z}\langle i, j\rangle$
2. observe a triple $(i_t, j_t, y_t) \in [m] \times [n] \times \{-1, 1\}$,
3. suffer loss $\ell_t(\boldsymbol{W}_t)$ where $\ell_t : \boldsymbol{W} \mapsto h_\gamma(y_t(\boldsymbol{W} \bullet \boldsymbol{Z}\langle i_t, j_t\rangle))$,
4. let $\boldsymbol{L}_t = \nabla_{\boldsymbol{W}} \ell_t(\boldsymbol{W}_t) = \begin{cases} -\frac{y_t}{\gamma}\boldsymbol{Z}\langle i_t, j_t\rangle & \text{if } y_t\boldsymbol{X}_{t,(i,j)} \le \gamma \\ 0 & \text{otherwise} \end{cases}$,
5. feed $\boldsymbol{L}_t$ to the algorithm $\mathcal{A}$ to let it proceed and receive $\boldsymbol{W}_{t+1}$.

Since the loss function ℓ_t is convex, a standard linearization argument ([48]) gives

$$\ell_t(\boldsymbol{W}_t) - \ell_t(\boldsymbol{W}^*) \le \boldsymbol{W}_t \bullet \boldsymbol{L}_t - \boldsymbol{W}^* \bullet \boldsymbol{L}_t$$

for any $\boldsymbol{W}^* \in \mathcal{K}$. Moreover, since $\ell_t(\boldsymbol{W}_t) = h_\gamma(y_t\boldsymbol{X}_{t,(i_t,j_t)})$ and $\ell_t(\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}) = h_\gamma(y_t(\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i_t,j_t})$, the following lemma immediately follows.

Lemma 12. *Let* $\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}, \boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}) = \sum_{t=1}^{T}(\boldsymbol{W}_t - \boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}) \bullet \boldsymbol{L}_t$ *denote the regret of the algorithm* $\mathcal{A}$ *in the reduction above for a competitor matrix* $\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}$ *and* $\mathrm{Regret}_{\mathrm{OMP}}(T, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top) = \sum_{t=1}^{T}(h_\gamma(y_t\boldsymbol{X}_{t,(i_t,j_t)}) - h_\gamma(y_t(\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i_t,j_t})$ *denote the regret of the reduction algorithm for* $\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top$. *Then,*

$$\mathrm{Regret}_{\mathrm{OMP}}(T, \mathcal{X}, \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top) \le \mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}, \boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}).$$

Combining Lemma 10 and Lemma 12, we have the following corollary.

Corollary 4. *There exists an algorithm for the OBMC problem with side information with the following mistake bounds.*

$$\begin{aligned} M &\leq \inf_{\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top \in \mathcal{X}} (\mathrm{Regret}_{\mathrm{OSDP}}(M, \mathcal{K}, \mathcal{L}, \boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}) + \mathrm{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma)) \\ &\leq \mathrm{Regret}_{\mathrm{OSDP}}(M, \mathcal{K}, \mathcal{L}) + \mathrm{hloss}(\mathcal{S}, \gamma). \end{aligned}$$

5.4.4 Application to matrix completion

According to the above two reductions, we can reduce OBMC with side information $\boldsymbol{M}$ and $\boldsymbol{N}$ to a generalised OSDP problem $(\mathcal{K}, \mathcal{L})$ with bounded $\boldsymbol{\Gamma}$-trace norm defined in (5.17) and (5.18), where $\boldsymbol{\Gamma}$ is respect to side information matrices $\boldsymbol{M}$ and $\boldsymbol{N}$, defined as in (5.15), hence we can apply FTRL algorithm with the generalised log-determinant regularizer defined in (5.7). Again, the generalised log-determinant regularizer becomes the regular form as $-\ln\det(\boldsymbol{W} + \epsilon\boldsymbol{E})$, when the side information is vacuous.

Remark 5. *Since the definition of* $\boldsymbol{\Gamma}$ *in Equation (5.15), we have that* $\rho = 1$.

Thus we set $\beta = 1$, $g = 1/\gamma$, $\epsilon = \rho = 1$,$\tau = \widehat{\mathcal{D}}$, and $\boldsymbol{\Gamma}$ is given as in Equation (5.15), then utilise Theorem 12, so we get the following result

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}, \boldsymbol{W}^*) = O\left(\frac{T\eta}{\gamma^2} + \frac{\widehat{\mathcal{D}}}{\eta}\right). \tag{5.19}$$

Before stating our improved mistake bound, we give in Algorithm 7 the algorithm for the OBMC problem with side information $\boldsymbol{M}, \boldsymbol{N}$ which is obtained by putting together the two reductions with the FTRL algorithm (5.4).

Theorem 13. *Running Algorithm 7 with parameter* $\eta = \sqrt{\gamma^2\widehat{\mathcal{D}}/T}, \gamma \in (0, 1]$ *the hinge loss of OBMC with side information is bounded as follows:*

$$\sum_{t=1}^{T} h_\gamma(y_t \cdot \hat{y}_t) - \sum_{t=1}^{T} h_\gamma(y_t \cdot (\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i_t,j_t}) \leq O\left(\sqrt{\frac{\widehat{\mathcal{D}}T}{\gamma^2}}\right). \tag{5.20}$$

Compared with [31], our regret bound with hinge loss is improved with $\ln(m + n)$.

Meanwhile, according to our mistake-driven technique, the horizon T is set to be the number of mistakes M, through the reduction, which is unknown in advance. Then, by choosing η independent of M we can derive a good mistake bound due to above theorem.

Algorithm 7 Online binary matrix completion with side information algorithm

Parameters: $\gamma > 0$, $\eta > 0$, side information matrices $\boldsymbol{M} \in \mathbb{S}_{++}^{m \times m}$ and $\boldsymbol{N} \in \mathbb{S}_{++}^{n \times n}$. Quasi dimension estimator $1 \leq \widehat{\mathcal{D}}$. $\boldsymbol{\Gamma}$ is composed as in Equation (5.15), and decision set $\mathcal{K}$ is given as (5.17).
Initialize $\forall \boldsymbol{W} \in \mathcal{K}$, set $\boldsymbol{W}_1 = \boldsymbol{W}$.
for $t = 1, 2, \ldots, T$ **do**
  Receive $(i_t, j_t) \in [m] \times [n]$.
  Let $\boldsymbol{Z}_t = \frac{1}{2}(\boldsymbol{e}_{i_t} \boldsymbol{e}_{m+j_t}^T + \boldsymbol{e}_{m+j_t} \boldsymbol{e}_{i_t}^T)$.
  Predict $\hat{y}_t = \operatorname{sgn}(\boldsymbol{W}_t \bullet \boldsymbol{Z}_t)$ and receive $y_t \in \{-1, 1\}$.
  if $\hat{y}_t \neq y_t$ **then**
    Let $\boldsymbol{L}_t = \frac{-y_t}{\gamma} \boldsymbol{Z}_t$ and $\boldsymbol{W}_{t+1} = \arg\min_{\boldsymbol{W} \in \mathcal{K}} - \ln\det(\boldsymbol{\Gamma} \boldsymbol{W} \boldsymbol{\Gamma} + \boldsymbol{E}) + \eta \sum_{s=1}^{t} \boldsymbol{W} \bullet \boldsymbol{L}_s$.
  else
    Let $\boldsymbol{L}_t = 0$ and $\boldsymbol{W}_{t+1} = \boldsymbol{W}_t$.
  end if
end for

Theorem 14. *Algorithm 7 with $\eta = c\gamma^2$ for some $c > 0$ achieves*

$$M = \sum_{t=1}^{T} \mathbb{I}_{\hat{y}_t \neq y_t} = O\left(\frac{\widehat{\mathcal{D}}}{\gamma^2}\right) + 2\mathrm{hloss}(\mathcal{S}, \gamma). \tag{5.21}$$

Proof. Combining Corollary 4 and the regret bound (5.19), we have

$$M = O\left(\frac{M\eta}{\gamma^2} + \frac{\widehat{\mathcal{D}}}{\eta}\right) + \mathrm{hloss}(\mathcal{S}, \gamma).$$

Choosing $\eta = c\gamma^2$ for sufficiently small constant c, we get

$$M \leq \frac{M}{2} + O\left(\frac{\widehat{\mathcal{D}}}{\gamma^2}\right) + \mathrm{hloss}(\mathcal{S}, \gamma),$$

from which (5.21) follows.

□

Again if the side information is vacuous, which means that $\boldsymbol{M}, \boldsymbol{N}$ are identity matrices, from Remark 4 and Theorem 14, we can set that $\widehat{\mathcal{D}} = m + n$ and obtain the mistake bound as follows:

$$O\left(\frac{m+n}{\gamma^2} + 2\mathrm{hloss}_{\boldsymbol{P}\boldsymbol{Q}^T \in \mathbb{R}^{m \times n}}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma)\right).$$

In contrast, there is a case where side information matters non-trivially. For OBMC with side information $\boldsymbol{M}, \boldsymbol{N}$ we can consider the comparator matrix $\boldsymbol{U}$ as the upper-right block in

an optimal matrix in decision set (5.17) for reduced generalised OSDP problem with $\mathbf{\Gamma}$-trace norm. Then by choosing special $\boldsymbol{M}, \boldsymbol{N}$ the class of comparator matrix $\boldsymbol{U}$ contains meaningful structure, especially, if $\boldsymbol{U}$ contains (k, l)-biclustered structure(the details are in Supplement material) then we obtain that $\widehat{\mathcal{D}} \in O(k + l)$, which is strictly smaller than $O(m + n)$.

Note that in the realizable case, our mistake bound becomes $O\left(\frac{\widehat{\mathcal{D}}}{\gamma^2}\right)$, which improves the previous bound $O\left(\frac{\widehat{\mathcal{D}}}{\gamma^2}\ln(m+n)\right)$ in [31], removing the logarithmic factor $\ln(m+n)$. Furthermore, this bound matches the previously known lower bound of Herbster et.al. [30]. When $\boldsymbol{U}$ contains (k, l)-biclustered structure ($k \geq l$), γ can be set as $\gamma = \frac{1}{\sqrt{l}}$ and our regret bound becomes $O(kl)$. On the other hand, the lower bound of Herbster et.al. is $\Omega(kl)$. Thus, the mistake bound of Theorem 14 is optimal.

5.5 Connection to a batch setting

In this section, we employ the well known online-to-batch conversion technique (see, e.g., [42]) and obtain a batch learning algorithm with generalization error bounds. The results imply that the algorithm performs nearly as well as the SVM running over the optimal feature space.

First we describe our setting formally. We consider the problem in the standard PAC learning framework. The algorithm is given the side information matrix $\boldsymbol{M} \in \mathbb{S}_{++}^{m\times m}$ and $\boldsymbol{N} \in \mathbb{S}_{++}^{n\times n}$ and a sample sequence $\mathcal{S}$:

$$\mathcal{S} = ((i_1, j_1, y_1), (i_2, j_2, y_2), \ldots, (i_T, j_T, y_T))$$

where each triple (i_t, j_t, y_t) is randomly and independently generated according to some unknown probability distribution $\mathcal{V}$ over $[m] \times [n] \times \{-1, 1\}$. Then the algorithm outputs a hypothesis $f : [m] \times [n] \to [-1, 1]$. The goal is to find, with high probability, a hypothesis f that has small generalization error

$$R(f) = \Pr_{(i,j,y)\sim\mathcal{V}}(\mathrm{sgn}(f(i, j)) \neq y).$$

In particular, we consider a hypothesis of the form of

$$f_{\boldsymbol{W}} : (i, j) \mapsto \boldsymbol{W} \bullet \boldsymbol{Z}\langle i, j\rangle$$

where $\boldsymbol{W} \in \mathcal{K} = \{\boldsymbol{W} \in \mathbb{S}_{++}^{N\times N} : \forall i \in [N], \boldsymbol{W}_{i,i} \leq 1 \wedge \mathrm{Tr}(\mathbf{\Gamma}\boldsymbol{W}\mathbf{\Gamma}) \leq \widehat{\mathcal{D}}\}$, where $\mathbf{\Gamma}$ is defined

as in Equation (5.15).

In Algorithm 8 we give the algorithm obtained by the online-to-batch conversion.

Algorithm 8 Binary matrix completion in the batch setting

1: Parameter: $\gamma > 0$
2: Input: a sample $\mathcal{S}$ of size T.
3: Run Algorithm 7 over $\mathcal{S}$ and get its predictions $\boldsymbol{W}_1, \boldsymbol{W}_2, \cdots, \boldsymbol{W}_T$.
4: Choose $\boldsymbol{W}$ from $\{\boldsymbol{W}_1, \boldsymbol{W}_2, \cdots, \boldsymbol{W}_T\}$ uniformly at random.
5: Output $f_{\boldsymbol{W}}$.

To bound the generalization error, we use the following lemma, which is straightforward from Lemma 7.1 of [42].

Lemma 13. *Let $L : [-1, 1] \times \{-1, 1\} \to [-B, B]$ be a function and $\boldsymbol{W}_1, \cdots, \boldsymbol{W}_T$ and $\boldsymbol{W}$ be the matrices obtained in Algorithm 8. Then, for any $\delta > 0$, with probability at least $1 - \delta$, the following holds:*

$$\begin{aligned}\mathbb{E}_{(i,j,y)\sim\mathcal{V},\boldsymbol{W}}[L(f_{\boldsymbol{W}}(i,j),y)] &= \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{(i,j,y)\sim\mathcal{V}}[L(f_{\boldsymbol{W}}(i,j),y)] \\ &\leq \frac{1}{T}\sum_{t=1}^{T}L(f_{\boldsymbol{W}_t}(i_t,j_t),y_t) + B\sqrt{\frac{2\ln 1/\delta}{T}}.\end{aligned}$$

Applying the lemma with the zero-one loss $L(r, y) = \mathbf{1}(\mathrm{sgn}(r) \neq y)$ combined with the mistake bound (5.21) of Theorem 14, we have the following generalization bound.

Theorem 15. *For any $\delta > 0$, with probability at least $1 - \delta$, Algorithm 8 produces $f_{\boldsymbol{W}}$ with the following property:*

$$\mathbb{E}_{\boldsymbol{W}}[R(f_{\boldsymbol{W}})] \leq \frac{O\left(\frac{\widehat{\mathcal{D}}}{\gamma^2} + \mathrm{hloss}(\mathcal{S},\gamma)\right)}{T} + \sqrt{\frac{2\ln 1/\delta}{T}}. \tag{5.22}$$

On the other hand, when applying the lemma with the hinge loss $L(r, y) = h_\gamma(ry)$ combined with an $O(\sqrt{\widehat{\mathcal{D}}T/\gamma^2})$ regret bound of (5.19) with the minimizer $\eta = \sqrt{\gamma^2\widehat{\mathcal{D}}/T}$, then we have

$$\begin{aligned}\mathbb{E}_{\boldsymbol{W}}[R(f_{\boldsymbol{W}})] &\leq \mathbb{E}_{(i,j,y)\sim\mathcal{V},\boldsymbol{W}}[h_\gamma(yf_{\boldsymbol{W}}(i,j))] \\ &\leq O\left(\sqrt{\frac{\widehat{\mathcal{D}}}{\gamma^2 T}} + \frac{\mathrm{hloss}(\mathcal{S},\gamma)}{T}\right) + (1+\gamma)\sqrt{\frac{2\ln 1/\delta}{T}},\end{aligned} \tag{5.23}$$

which is slightly worse than (5.22).

Now we examine some implications of our generalization bounds. First we assume without loss of generality that $m \geq n$, because otherwise we can make everything transposed.

As explained in the aforementioned sections, we can think of each $\bar{\boldsymbol{Q}}_j$ as a feature vector of item j. Assume all feature vectors $\bar{\boldsymbol{Q}} \in \mathbb{R}^{n \times k}$ are given and consider the problem of finding a good linear classifier $\bar{\boldsymbol{P}}_i$ for each user i independently. A natural way is to use the SVM, which solves

$$\inf_{\gamma>0, \boldsymbol{P}_i \in \mathbb{R}^k} \left(1/\gamma^2 + C \sum_{t:i_t=i} h_\gamma(y_t \bar{\boldsymbol{P}}_i \bar{\boldsymbol{Q}}_{j_t}^\top) \right)$$

for every $i \in [m]$ for some constant $C > 0$. Now if we fix γ to be a constant for all i, then the optimization problems above are summarized as

$$\begin{aligned} \inf_{\boldsymbol{P} \in \mathbb{R}^{m \times k}} \sum_{i=1}^{m} \left(1/\gamma^2 + C \sum_{t:i_t=i} h_\gamma(y_t \bar{\boldsymbol{P}}_i \bar{\boldsymbol{Q}}_{j_t}^\top) \right) &= \inf_{\boldsymbol{P} \in \mathbb{R}^{m \times k}} \left(m/\gamma^2 + C \sum_{t} h_\gamma(y_t \bar{\boldsymbol{P}}_{i_t} \bar{\boldsymbol{Q}}_{j_t}^\top) \right) \\ &= \frac{m}{\gamma^2} + C \inf_{\boldsymbol{P}} \operatorname{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma). \end{aligned}$$

So, if we further optimize feature vectors, we get

$$\frac{m}{\gamma^2} + C \inf_{\boldsymbol{P}\boldsymbol{Q}^\top \in \mathbb{R}^{m \times n}} \operatorname{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma) = \frac{m}{\gamma^2} + C \operatorname{hloss}(\mathcal{S}, \gamma) \tag{5.24}$$

which roughly is proportional to our generalization bound (5.22), when the side information is vacuous (i.e., $\boldsymbol{M}$ and $\boldsymbol{N}$ are identity matrices). This result implies that our generalization bound is upper bounded by the objective function value of the SVM running over the optimal choice of feature vectors. Meanwhile, we expect a more refined bound when the side information is not vacuous for the batch setting.

Moreover, a well known generalization bound for linear classifiers (see, e.g., [42]) gives

$$\Pr_{(j,y) \sim \mathcal{V}_i} (\operatorname{sgn}(\bar{\boldsymbol{P}}_i \bar{\boldsymbol{Q}}_j^\top) \neq y) \leq \frac{1}{T_i} \sum_{t:i_t=i} h_\gamma(y_t \bar{\boldsymbol{P}}_{i_t} \bar{\boldsymbol{Q}}_{j_t}^\top) + 2\sqrt{\frac{1}{\gamma^2 T_i}} + \sqrt{\frac{\ln(1/\delta)}{2T_i}}$$

for every $i \in [m]$, where $\mathcal{V}_i$ is the conditional distribution of $\mathcal{V}$ given that the first component is i, and T_i is the number of $t \in [T]$ that satisfies $i_t = i$. Assume for simplicity that $\mathcal{V}(i) =$

$\sum_{j,y} \mathcal{V}(i,j,y) = 1/m$ and $T_i = T/m$ for every i. Then,

$$\begin{aligned} R(f_{\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}}) &= \sum_{i=1}^{m} \mathcal{V}(i) \Pr_{(j,y)\sim\mathcal{V}_i}(\mathrm{sgn}(\bar{\boldsymbol{P}}_i \bar{\boldsymbol{Q}}_j^\top) \neq y) \\ &\leq \sum_{i=1}^{m} \mathcal{V}(i) \left(\frac{1}{T_i} \sum_{t:i_t=i} h_\gamma(y_t \bar{\boldsymbol{P}}_{i_t} \bar{\boldsymbol{Q}}_{j_t}^\top)) + 2\sqrt{\frac{1}{\gamma^2 T_i}} + \sqrt{\frac{\ln(1/\delta)}{2T_i}} \right) \\ &= \frac{1}{T}\mathrm{hloss}(\mathcal{S}, (\boldsymbol{P}, \boldsymbol{Q}), \gamma) + 2\sqrt{\frac{m}{\gamma^2 T}} + \sqrt{\frac{m\ln(1/\delta)}{2T}}. \end{aligned}$$

Minimizing $R(f_{\boldsymbol{W}_{\boldsymbol{P},\boldsymbol{Q}}})$ over all $\boldsymbol{P}$ and $\boldsymbol{Q}$ such that $\boldsymbol{P}\boldsymbol{Q}^\top \in \mathbb{R}^{m\times n}$, the bound obtained is very similar to our bound (5.23). This observation implies that our hypothesis has generalization ability competitive with the optimal linear classifiers $\bar{\boldsymbol{P}}$ over the optimal feature vectors $\bar{\boldsymbol{Q}}$.

5.6 Conclusion

In this paper, on the one hand, we define a generalised OSDP problem with bounded $\boldsymbol{\Gamma}$-trace norm. To solve this problem, we involve FTRL with the generalised log-determinant regularizer and achieve regret bound as $O(g\sqrt{\beta\tau\rho T})$. On the other hand, we utilise our result to OBMC with side information particularly. We reduce OBMC with side information to our new OSDP with bounded $\boldsymbol{\Gamma}$-trace norm and obtain a tighter mistake bound than previous work by removing logarithmic factor.

5.7 Acknowledgement

This work was supported by JSPS KAKENHI Grant Numbers JP19H04174 and JP19H04067, respectively. We thank the reviewers for their suggestions and comments.

5.8 Appendix

5.8.1 Proof of Main Proposition and Main Theorem

Before we prove this theorem, we need to involve some Lemmata and notations.

The negative entropy function over the set of probability distribution P over $\mathbb{R}^N$ is defined as $H(P) = \mathbb{E}_{x\sim P}[\ln(P(x))]$. The total variation distance between probability distribution P

and Q over $\mathbb{R}^N$ is defined as $\frac{1}{2}\int_x |P(x) - Q(x)|dx$. The characteristic function of a probability distribution P over $\mathbb{R}^N$ is defined as $\phi(u) = \mathbb{E}_{x\sim P}[e^{iu^T x}]$ where i is the imaginary unit.

Lemma 14. *Let G_1 and G_2 be two zero mean Gaussian distributions with covariance matrix $\mathbf{\Gamma\Sigma\Gamma}$ and $\mathbf{\Gamma\Theta\Gamma}$. Furthermore $\mathbf{\Sigma}$ and $\mathbf{\Theta}$ are positive definite matrices. If there exists (i, j) such that*

$$|\mathbf{\Sigma}_{i,j} - \mathbf{\Theta}_{i,j}| \geq \delta(\mathbf{\Sigma}_{i,i} + \mathbf{\Theta}_{i,i} + \mathbf{\Sigma}_{j,j} + \mathbf{\Theta}_{j,j}), \tag{5.25}$$

then the total variation distance between G_1 and G_2 is at least $\frac{1}{12e^{1/4}}\delta$.

Proof. Given $\phi_1(u)$ and $\phi_2(u)$ as characteristic function of G_1 and G_2 respectively. Due to Lemma 17 in [44], we have

$$\int_x |G_1(x) - G_2(x)|dx \geq \max_{u\in\mathbb{R}^N} |\phi_1(u) - \phi_2(u)|, \tag{5.26}$$

so we only need to show the lower bound of $\max_{u\in\mathbb{R}^N} |\phi_1(u) - \phi_2(u)|$.

Then we set that characteristic function of G_1 and G_2 are $\phi_1(u) = e^{\frac{-1}{2}u^T\mathbf{\Gamma}^T\mathbf{\Sigma\Gamma}u}$ and $\phi_2(u) = e^{\frac{-1}{2}u^T\mathbf{\Gamma}^T\mathbf{\Theta\Gamma}u}$ respectively. Set that $\alpha_1 = (\mathbf{\Gamma}v)^T\mathbf{\Sigma}(\mathbf{\Gamma}v)$, $\alpha_2 = (\mathbf{\Gamma}v)^T\mathbf{\Theta}(\mathbf{\Gamma}v)$ and $\mathbf{\Gamma}u = \frac{\mathbf{\Gamma}v}{\sqrt{\alpha_1+\alpha_2}}$. Moreover we denote that $\bar{v} = \Gamma v$, for any $\bar{v} \in \mathbb{R}^V$, there exists $v \in \mathbb{R}^V$. $\bar{u} = \Gamma u$ in the same way.

We need only give the lower bound of $\max_{u\in\mathbb{R}^N} |\phi_1(u) - \phi_2(u)|$.

Next we have that

$$\begin{aligned}
&\max_{u\in\mathbb{R}^N} |\phi_1(u) - \phi_2(u)| \\
&= \max_{u\in\mathbb{R}^N} \left| e^{\frac{-1}{2}u^T\mathbf{\Gamma\Sigma\Gamma}u} - e^{\frac{-1}{2}u^T\mathbf{\Gamma\Theta\Gamma}u} \right| \\
&= \max_{u\in\mathbb{R}^V} \left| e^{\frac{-1}{2}(\mathbf{\Gamma}u)^T\mathbf{\Sigma}(\mathbf{\Gamma}u)} - e^{\frac{-1}{2}(\mathbf{\Gamma}u)^T\mathbf{\Theta}(\mathbf{\Gamma}u)} \right| \\
&\geq \max_{\bar{v}\in\mathbb{R}^N} \left| e^{\frac{-\alpha_1}{2(\alpha_1+\alpha_2)}} - e^{\frac{-\alpha_2}{2(\alpha_1+\alpha_2)}} \right| \\
&\geq \max_{\bar{v}\in\mathbb{R}^N} \left| \frac{1}{2e^{1/4}} \frac{\alpha_1 - \alpha_2}{\alpha_1 + \alpha_2} \right|.
\end{aligned} \tag{5.27}$$

Then second inequality is due to Lemma 20, since $\min\{\frac{\alpha_1}{\alpha_1+\alpha_2}, \frac{\alpha_2}{\alpha_1+\alpha_2}\} \in (0, \frac{1}{2}]$.

Due to assumption in the Lemma we obtain for some (i, j) that

$$\begin{aligned}
&\delta(\mathbf{\Sigma}_{i,i} + \mathbf{\Theta}_{i,i} + \Sigma_{j,j} + \mathbf{\Theta}_{j,j}) \leq |\mathbf{\Sigma}_{i,j} - \mathbf{\Theta}_{i,j}| \\
&= \frac{1}{2}|(\boldsymbol{e}_i + \boldsymbol{e}_j)^T(\mathbf{\Sigma} - \mathbf{\Theta})(\boldsymbol{e}_i + \boldsymbol{e}_j) - \boldsymbol{e}_i^T(\mathbf{\Sigma} - \mathbf{\Theta})\boldsymbol{e}_i - \boldsymbol{e}_j^T(\mathbf{\Sigma} - \mathbf{\Theta})\boldsymbol{e}_j|
\end{aligned} \tag{5.28}$$

It implies that one of $(\boldsymbol{e}_i+\boldsymbol{e}_j)^T(\boldsymbol{\Sigma}-\boldsymbol{\Theta})(\boldsymbol{e}_i+\boldsymbol{e}_j)$, $\boldsymbol{e}_i^T(\boldsymbol{\Sigma}-\boldsymbol{\Theta})\boldsymbol{e}_i$ and $\boldsymbol{e}_j^T(\boldsymbol{\Sigma}-\boldsymbol{\Theta})\boldsymbol{e}_j$ has absolute value greater that $\frac{2\delta}{3}(\boldsymbol{\Sigma}_{i,i}+\boldsymbol{\Theta}_{i,i}+\boldsymbol{\Sigma}_{j,j}+\boldsymbol{\Theta}_{j,j})$.

Since Σ, Θ are strictly positive definite matrices, we have that for all $v \in \{\boldsymbol{e}_i+\boldsymbol{e}_j, \boldsymbol{e}_i, \boldsymbol{e}_j\}$

$$v^T(\boldsymbol{\Sigma}+\boldsymbol{\Theta})v \leq 2(\boldsymbol{\Sigma}+\boldsymbol{\Theta})_{i,i} + 2(\boldsymbol{\Sigma}+\boldsymbol{\Theta})_{j,j}. \tag{5.29}$$

and therefore we have that

$$\max_{\bar{v}\in\mathbb{R}^N}\left|\frac{1}{2e^{1/4}}\frac{\alpha_1-\alpha_2}{\alpha_1+\alpha_2}\right| \geq \max_{\bar{v}\in\{\boldsymbol{e}_i+\boldsymbol{e}_j,\boldsymbol{e}_i,\boldsymbol{e}_j\}}\left|\frac{1}{2e^{1/4}}\frac{v^T(\boldsymbol{\Sigma}-\boldsymbol{\Theta})v}{v^T(\boldsymbol{\Sigma}+\boldsymbol{\Theta})v}\right| \geq \frac{\delta}{6e^{1/4}} \tag{5.30}$$

□

Lemma 15. *Let $X, Y \in \mathbb{S}_+^{N\times N}$ be such that*

$$|\boldsymbol{X}_{i,j}-\boldsymbol{Y}_{i,j}| \geq \delta(\boldsymbol{X}_{i,i}+\boldsymbol{Y}_{i,i}+\boldsymbol{X}_{j,j}+\boldsymbol{Y}_{j,j}), \tag{5.31}$$

and $\boldsymbol{\Gamma}$ is a symmetric strictly positive definite matrix. Then the following inequality holds that

$$\begin{aligned}&-\ln\det(\alpha\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\Gamma}+(1-\alpha)\boldsymbol{\Gamma}\boldsymbol{Y}\boldsymbol{\Gamma})\\ &\leq -\alpha\ln\det(\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\Gamma})-(1-\alpha)\ln\det(\boldsymbol{\Gamma}\boldsymbol{Y}\boldsymbol{\Gamma})-\frac{\alpha(1-\alpha)}{2}\frac{\delta^2}{72e^{1/2}}.\end{aligned} \tag{5.32}$$

Proof. Let G_1 and G_2 be zero mean Gaussian distributions with covariance matrix $\boldsymbol{\Gamma}\boldsymbol{\Sigma}\boldsymbol{\Gamma} = \boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\Gamma}$ and $\boldsymbol{\Gamma}\boldsymbol{\Theta}\boldsymbol{\Gamma} = \boldsymbol{\Gamma}\boldsymbol{Y}\boldsymbol{\Gamma}$. In matrix total variation distance between G_1 and G_2 is at least $\frac{\delta}{12e^{1/4}}$, since assumption of this Lemma and result in Lemma 14. We denote that $\tilde{\delta} = \frac{\delta}{12e^{1/4}}$. Consider the entropy of the following probability distribution of v with probability α that $v \sim G_1$ and $v \sim G_2$ otherwise. Its covariance matrix is $\alpha\boldsymbol{\Gamma}\boldsymbol{\Sigma}\boldsymbol{\Gamma}+(1-\alpha)\boldsymbol{\Gamma}\boldsymbol{\Theta}\boldsymbol{\Gamma}$. Due to Lemma A.2 and Lemma A.3 [44] (see in Appendix) we obtain that

$$\begin{aligned}&-\ln\det(\alpha\boldsymbol{\Gamma}\boldsymbol{\Sigma}\boldsymbol{\Gamma}+(1-\alpha)\boldsymbol{\Gamma}\boldsymbol{\Theta}\boldsymbol{\Gamma})\\ &\leq 2H(\alpha G_1+(1-\alpha)G_2)+\ln(2\pi e)^V\\ &\leq 2\alpha H(G_1)+2(1-\alpha)H(G_2)+\ln(2\pi e)^V-\alpha(1-\alpha)\tilde{\delta}^2\\ &= -\alpha\ln\det(\boldsymbol{\Gamma}\boldsymbol{\Sigma}\boldsymbol{\Gamma})-(1-\alpha)\ln\det(\boldsymbol{\Gamma}\boldsymbol{\Theta}\boldsymbol{\Gamma})-\alpha(1-\alpha)\tilde{\delta}^2.\end{aligned}$$

□

Lemma 16 (Lemma 5.4 [44])**.** *Let $\boldsymbol{X}, \boldsymbol{Y} \in \mathbb{S}_{++}^{N\times N}$ be such that for all $i \in [N]$ $|\boldsymbol{X}_{i,i}| \leq \beta'$ and*

$|\boldsymbol{Y}_{i,i}| \leq \beta^{'}$ *Then for any* $\boldsymbol{L} \in \mathcal{L} = \{\boldsymbol{L} \in \mathbb{S}_{+}^{N\times N} : \|\mathrm{vec}(\boldsymbol{L})\|_1 \leq g\}$ *there exists that*

$$|\boldsymbol{X}_{i,j} - \boldsymbol{Y}_{i,j}| \geq \frac{|\boldsymbol{L} \bullet (\boldsymbol{X} - \boldsymbol{Y})|}{4\beta' g}(\boldsymbol{X}_{i,i} + \boldsymbol{Y}_{i,i} + \boldsymbol{X}_{j,j} + \boldsymbol{Y}_{j,j}). \tag{5.33}$$

Proposition 7 (Main proposition in main part)**.** *The generalised log-determinant regularizer* $R(X) = -\ln\det(\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\Gamma} + \epsilon\boldsymbol{E})$ *is* s*-strongly convex with respect to* $\mathcal{L}$ *for* $\mathcal{K}$ *with* $s = 1/(1152\sqrt{e}(\beta + \rho\epsilon)^2 g^2)$*. Here* $\boldsymbol{E}$ *is identity matrix.*

Proof. Firstly we know that $\boldsymbol{\Gamma}\boldsymbol{X}\boldsymbol{\Gamma} + \epsilon\boldsymbol{E} = \boldsymbol{\Gamma}(\boldsymbol{X} + \boldsymbol{\Gamma}^{-1}\epsilon\boldsymbol{E}\boldsymbol{\Gamma}^{-1})\boldsymbol{\Gamma}$.

Applying the Lemma 16 to $\boldsymbol{X} + \boldsymbol{\Gamma}^{-1}\epsilon\boldsymbol{E}\boldsymbol{\Gamma}^{-1}$ and $\boldsymbol{Y} + \boldsymbol{\Gamma}^{-1}\epsilon\boldsymbol{E}\boldsymbol{\Gamma}^{-1}$ for $\boldsymbol{X}, \boldsymbol{Y} \in \mathcal{K}$ where $\max_{i,j}|(\boldsymbol{X} + \boldsymbol{\Gamma}^{-1}\epsilon\boldsymbol{E}\boldsymbol{\Gamma}^{-1})_{i,j}| \leq \max_{i,j}|\boldsymbol{X}_{i,j}| + \epsilon\rho$, we have that $\beta^{'} = \beta + \epsilon\rho$, where $\rho = \max_{i,j}|(\boldsymbol{\Gamma}^{-1}\boldsymbol{\Gamma}^{-1})_{i,j}|$. According to Lemma 15 and Definition 5 we have this proposition.

□

Now we give the proof of the Main Theorem as follows:

Proof of Theorem 12. Due to Lemma 9 (in main part) we obtain that

$$\mathrm{Regret}_{\mathrm{OSDP}}(T, \mathcal{K}, \mathcal{L}, \boldsymbol{W}^*) \leq \frac{H_0}{\eta} + \frac{\eta}{s}T. \tag{5.34}$$

Due to the main proposition in main part we know that $s = 1/(1152(\beta + \rho\epsilon)^2\sqrt{e}g^2)$.

Thus we need only to show $H_0 \leq \frac{\tau}{\epsilon}$. Given $\boldsymbol{W}_0$ and $\boldsymbol{W}_1$ is the minimizer and maximizer of R respectively, then we obtain that

$$\begin{aligned}
&\max_{\boldsymbol{W},\boldsymbol{W}'\in\mathcal{K}}(R(\boldsymbol{W}) - R(\boldsymbol{W}^{'})) = R(\boldsymbol{W}_1) - R(\boldsymbol{W}_0)\\
&= -\ln\det(\boldsymbol{\Gamma}\boldsymbol{W}_1\boldsymbol{\Gamma} + \epsilon\boldsymbol{E}) + \ln\det(\boldsymbol{\Gamma}\boldsymbol{W}_0\boldsymbol{\Gamma} + \epsilon\boldsymbol{E})\\
&= \sum_{i=1}^{N}\ln\frac{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_0\boldsymbol{\Gamma}) + \epsilon}{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_1\boldsymbol{\Gamma}) + \epsilon}\\
&= \sum_{i=1}^{N}\ln\left(\frac{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_0\boldsymbol{\Gamma})}{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_1\boldsymbol{\Gamma}) + \epsilon} + \frac{\epsilon}{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_1\boldsymbol{\Gamma}) + \epsilon}\right)\\
&\leq \sum_{i=1}^{N}\ln\left(\frac{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_0\boldsymbol{\Gamma})}{\epsilon} + 1\right)\\
&\leq \sum_{i=1}^{N}\frac{\lambda_i(\boldsymbol{\Gamma}\boldsymbol{W}_0\boldsymbol{\Gamma})}{\epsilon} = \frac{\mathrm{Tr}(\boldsymbol{\Gamma}\boldsymbol{W}_0\boldsymbol{\Gamma})}{\epsilon} \leq \frac{\tau}{\epsilon}.
\end{aligned} \tag{5.35}$$

Plugging s, we obtain that

$$\text{Regret}_{\text{OSDP}}(T, \mathcal{K}, \mathcal{L}, \boldsymbol{W}^*) = O\left(g^2(\beta + \rho\epsilon)^2 T\eta + \frac{\tau}{\epsilon\eta}\right). \tag{5.36}$$

□

Lemma 17 (Lemma A.1 [44]). *Let P and Q be probability distributions over $\mathbb{R}^N$ and $\phi_P(u)$ and $\phi_Q(u)$ be their characteristic functions, respectively. Then*

$$\max_{u\in\mathbb{R}^N} |\phi_P(u) - \phi_Q(u)| \leq \int_x |P(x) - Q(x)|dx, \tag{5.37}$$

the right hand side is the total variation distance between any distribution Q and P.

Lemma 18 (Lemma A.2 [15]). *Let P and Q be probability distributions over $\mathbb{R}^N$ with total variation distance δ. Then*

$$H(\alpha P + (1-\alpha)Q) \leq \alpha H(P) + (1-\alpha)H(Q) - \alpha(1-\alpha)\delta^2, \tag{5.38}$$

where $H(P) = \mathbb{E}_{x\sim P}[\ln P(x)]$.

Lemma 19 (Lemma A.3 [44]). *For any probability distribution P over $\mathbb{R}^N$ with zero mean and covariance matrix $\boldsymbol{\Sigma}$, its entropy is bounded by the log-determinant of covariance matrix. That is*

$$-H(P) \leq \frac{1}{2}\ln(\det(\Sigma)(2\pi e)^N). \tag{5.39}$$

Lemma 20 (Lemma A.4 [44]).

$$e^{\frac{-x}{2}} - e^{-\frac{1-x}{2}} \geq \frac{e^{-1/4}}{2}(1-2x), \tag{5.40}$$

for $0 \leq x \leq 1/2$.

5.8.2 Definition of biclustered structure and ideal quasi dimension

As in [31], we define the class of (k, l)-biclustered structure matrices as follows:

Definition 6. *For $m \geq k$ and $n \geq l$, the class of (k, l)-binary biclustered matrices is defined as*

$$\mathbb{B}_{k,l}^{m\times n} = \{\boldsymbol{U} \in \{-1, +1\}^{m\times n} : \boldsymbol{r} \in [k]^m, \boldsymbol{c} \in [l]^n, \boldsymbol{V} \in \{1, -1\}^{k\times l}, \boldsymbol{U}_{i,j} = \boldsymbol{V}_{r_i,c_j}, i \in [m], j \in [n]\}.$$

Denote $\mathcal{B}^{m,d} = \{\boldsymbol{R} \subset \{0,1\}^{m\times d} : \|\boldsymbol{R}_i\|_2 = 1, i \in [m], \mathrm{rank}(\boldsymbol{R}) = d\}$, for any matrix $\boldsymbol{U} \in \mathbb{B}_{k,l}^{m,n}$ we can decompose $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ for some $\boldsymbol{U}^* \in \{-1,+1\}^{k\times l}, \boldsymbol{R} \in \mathcal{B}^{m,k}$ and $\boldsymbol{C} \in \mathcal{B}^{n,l}$.

Theorem 16 ([31]). *If $\boldsymbol{U} \in \mathbb{B}_{k,l}^{m\times n}$ define $\mathcal{D}^o_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ as*

$$\mathcal{D}^o_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) = 2\mathrm{Tr}(\boldsymbol{R}^\top\boldsymbol{M}\boldsymbol{R})\alpha_{\boldsymbol{M}} + 2\mathrm{Tr}(\boldsymbol{C}^\top\boldsymbol{N}\boldsymbol{C})\alpha_{\boldsymbol{N}} + 2k + 2l, \tag{5.41}$$

where $\boldsymbol{M}, \boldsymbol{N}$ are PD-Laplacian, as the minimum over all decompositions of $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ for some $\boldsymbol{U}^* \in \{-1,+1\}^{k\times l}$, $\boldsymbol{R} \in \mathcal{B}^{m,k}$ and $\boldsymbol{C} \in \mathcal{B}^{n,l}$. Thus, for $\boldsymbol{U} \in \mathbb{B}_{k,l}^{m\times n}$,

$$\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \le \mathcal{D}^o_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}), \tag{5.42}$$

if $\|\boldsymbol{U}\|_{\max} \le \frac{1}{\gamma}$.

Moreover, we define the max-norm of a matrix $\boldsymbol{U} \in \mathbb{R}^{m\times n}$ as follows:

$$\|\boldsymbol{U}\|_{\max} = \min_{\boldsymbol{P}\boldsymbol{Q}^\top=\boldsymbol{U}} \left\{ \max_{1\le i\le m} \|\boldsymbol{P}_i\| \max_{1\le j\le n} \|\boldsymbol{Q}_j\| \right\}. \tag{5.43}$$

Furthermore we define the quasi-dimension of a matrix $\boldsymbol{U}$ with $\boldsymbol{M} \in \mathbb{S}_{++}^{m\times m}$ and $\boldsymbol{N} \in \mathbb{S}_{++}^{n\times n}$ at margin γ as

$$\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) = \min_{\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top=\gamma\boldsymbol{U}} \alpha_{\boldsymbol{M}}\mathrm{Tr}(\bar{\boldsymbol{P}}^\top\boldsymbol{M}\bar{\boldsymbol{Q}}) + \alpha_{\boldsymbol{N}}\mathrm{Tr}(\bar{\boldsymbol{Q}}^\top\boldsymbol{N}\bar{\boldsymbol{Q}}). \tag{5.44}$$

See section 4.1 from [31], if $\boldsymbol{U}$ is a (k,l)-biclustered structured matrix, they show an example where $\mathcal{D}^o_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \in O(k+l)$ with ideal side information. When exactly that there exists a sequence that $y_t = (\bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top)_{i_t,j_t} = \boldsymbol{U}_{i_t,j_t}$ where $(\bar{\boldsymbol{P}},\bar{\boldsymbol{Q}}) = \arg\min_{\boldsymbol{P},\boldsymbol{Q}} \mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$, and $\boldsymbol{U}$ satisfies the assumptions in [31], then we have that $\widehat{\mathcal{D}} \in O(k+l)$ with same side information.

5.8.3 Online similarity prediction with side information

In this section, we show the application of our reduction method and generalised log-determinant regularizer to online similarity prediction with side information.

Let $G = (V,E)$ be an undirected and connected graph with $n = |V|$ vertices and $m = |E|$ edges. Assign vertices to K classes with a vector $\boldsymbol{y} = \{y_1, \cdots, y_n\}$ where $y_i \in \{1, \cdots, K\}$. For a matrix $\boldsymbol{L}$, we denote $\boldsymbol{L}^+$ as pseudo-inverse matrix of $\boldsymbol{L}$. The online similarity prediction is defined as follows: On each round t, for a given pair of vertices (i_t, j_t) algorithm needs to

predict whether they are in the same class denoted as $\hat{y}_{i_t,j_t}$. If they are in the same class then $y_{i_t,j_t} = 1$, $y_{i_t,j_t} = -1$, otherwise. Our target is to give a bound of the prediction mistakes $M = \sum_{t=1}^{T} \mathbb{I}_{\hat{y}_{i_t,j_t} \neq y_{i_t,j_t}}$.

Definition 7. *The set of cut-edges in $(G, \boldsymbol{y})$ is denoted as $\Phi^G(\boldsymbol{y}) = \{(i,j) \in E : y_i \neq y_j\}$ we abbreviate it to Φ^G and the cut-size is given as $|\Phi^G(\boldsymbol{y})|$. The set of cut-edges with respect to class label k is denoted as $\Phi_k^G(\boldsymbol{y}) = \{(i,j) \in E : k \in \{y_i, y_j\}, y_i \neq y_j\}$. Note that $\sum_{s=1}^{k} |\Phi_s^G(\boldsymbol{y})| = 2|\Phi^G(\boldsymbol{y})|$. Given $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ such that $\boldsymbol{A}_{ij} = \boldsymbol{A}_{ji} = 1$ if $(i,j) \in E(G)$ and $\boldsymbol{A}_{ij} = 0$, otherwise. $\boldsymbol{D}$ is denoted as diagonal matrix with $\boldsymbol{D}_{ii}$ is the degree of vertex i. We define the Laplacian as $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$.*

Definition 8. *If G is identified with a resistive network such that each edge is a unit resistor, then the effective resistance $R_{i,j}^G$ between pair $(i,j) \in V^2$ can be defined as $R_{i,j}^G = (e_i - e_j)\boldsymbol{L}^+(e_i - e_j)$, where e_i is the i-th vector in the canonical basis of $\mathbb{R}^n$.*

[22] gave a mistake bound in the following proposition:

Proposition 8. *Let $(G, \boldsymbol{y})$ be a labeled graph. If we run the Matrix Winnow with G as input graph, we have the following mistake bound*

$$M^W = O\left(|\Phi^G| \max_{(i,j)\in V^2} R_{i,j}^G \ln n\right) \tag{5.45}$$

In our new reduction, we define the comparator matrix $\boldsymbol{U} \in \{1,-1\}^{n\times n}$ where if vertices i, j are in the same class then $\boldsymbol{U}_{ij} = 1$, and $\boldsymbol{U}_{ij} = -1$, otherwise. Firstly, we denote that $\mathbf{1}$ is a K-dimensional vector that all entries are 1. Due to [22, 31], we see that $\boldsymbol{U}$ is a (K,K)-biclustered $n \times n$ matrix where $\boldsymbol{U}^* = 2\boldsymbol{I}_K - \mathbf{1}\mathbf{1}^\top$, and there exists $\boldsymbol{R} \in \mathcal{B}^{n,k}$ such that $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{R}^\top$. Define the side information matrices $\boldsymbol{M} = \boldsymbol{N} \in \mathbb{R}^{n\times n}$ as PD-Laplcian $\tilde{\boldsymbol{L}}$, where $\boldsymbol{L}$ is the Laplacian matrix based on the graph G.

Thus we have

$$\boldsymbol{\Gamma} = \begin{bmatrix} \sqrt{\alpha_{\tilde{\boldsymbol{L}}}\widetilde{\boldsymbol{L}}} & 0 \\ 0 & \sqrt{\alpha_{\tilde{\boldsymbol{L}}}\widetilde{\boldsymbol{L}}} \end{bmatrix}, \tag{5.46}$$

where $\alpha_{\tilde{\boldsymbol{L}}} = \max_i(\tilde{\boldsymbol{L}}^{-1})_{ii}$.

According to [31], we further obtain that $\frac{1}{\gamma} \in O(1)$, more concisely we can set that $\frac{1}{\gamma} = 3$. Meanwhile given sparse matrix $\boldsymbol{Z}$ in the following equation

$$\boldsymbol{Z}\langle i,j\rangle = \frac{1}{2}(\boldsymbol{e}_i\boldsymbol{e}_{n+j}^\top + \boldsymbol{e}_{n+j}\boldsymbol{e}_i^\top). \tag{5.47}$$

Thus we give the following proposition for our reduction from a graph based online similarity prediction to a generalised OSDP problem $(\mathcal{K}, \mathcal{L})$ with bounded $\mathbf{\Gamma}$-trace norm.

Proposition 9. *Given an online similarity prediction problem with graph $(G, \boldsymbol{y})$, then we can reduce this problem to a generalised OSDP problem $(\mathcal{K}, \mathcal{L})$ with bounded $\mathbf{\Gamma}$-trace norm such that*

$$\mathcal{K} = \left\{ \boldsymbol{X} \in \mathbb{S}_{++}^{n\times n} : |\boldsymbol{X}_{ii}| \leq 1, \mathrm{Tr}(\mathbf{\Gamma}\boldsymbol{X}\mathbf{\Gamma}) \leq \widehat{\mathcal{D}} \right\}$$
$$\mathcal{L} = \{ c\boldsymbol{Z}\langle i, j\rangle : c \in \{-1/\gamma, 1/\gamma\}, i \in [n], j \in [n] \},$$

where $\mathbf{\Gamma}$ is defined as above, and $\widehat{\mathcal{D}}$ is arbitrary. In particular, we have that

$$M = \sum_{t=1}^{T} \mathbb{I}_{\hat{y}_{i_t,j_t} \neq y_{i_t,j_t}} \leq \mathrm{Regret}_{\mathrm{OSDP}}(M, \mathcal{K}, \mathcal{L}) \tag{5.48}$$

According to [31], there exists $\bar{\boldsymbol{P}}, \bar{\boldsymbol{Q}}$ such that $\boldsymbol{U}^* = \bar{\boldsymbol{P}}\bar{\boldsymbol{Q}}^\top$, it implies that the hinge loss $\mathrm{hloss}(\mathcal{S}, \gamma) = 0$.

Remark 6. *According to Theorem 3 and section 4.2 in [31] if $\boldsymbol{U}$ obtains the (K, K)-biclustered structure, i.e., there exists $\boldsymbol{U}^*$, such that $\boldsymbol{U}^* = 2\boldsymbol{I}_K - \boldsymbol{1}\boldsymbol{1}^\top$, and there exists $\boldsymbol{R} \in \mathcal{B}^{n,k}$ such that $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{R}^\top$, due to our Theorem 14 in main part, we have that*

$$M \leq O\left(\mathrm{Tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R})\alpha_{\boldsymbol{L}}\right), \tag{5.49}$$

where $\boldsymbol{L}$ is Laplacian of the corresponding graph G.

Remark 7. *According to [31], we have that*

$$\mathrm{Tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R}) \leq 2 \sum_{(i,j)\in E} \|\boldsymbol{R}_i - \boldsymbol{R}_j\|^2,$$

where $\sum_{(i,j)\in E} \|\boldsymbol{R}_i - \boldsymbol{R}_j\|^2$ counts only when there is an edge between different classes. Due to the definition of $|\Phi^G|$, we have that $\sum_{(i,j)\in E} \|R_i - R_j\|^2 = |\Phi^G|$.

Simultaneously, $\alpha_{\boldsymbol{L}} = \max_{i\in[n]} \boldsymbol{L}_{ii}^+$ so we obtain that $\alpha_{\boldsymbol{L}} \geq \boldsymbol{e}_i^\top \boldsymbol{L}^+ \boldsymbol{e}_i, \forall i \in [n]$. It implies that $4\alpha_{\boldsymbol{L}} \geq \max_{(i,j)\in V^2} R_{i,j}^G$. Thus we have that our new mistake bound improves the previous bound a logarithmic factor and recovers the previous bound up to a constant factor.

Chapter 6

Conclusion

In this thesis, we consider some other classes of online decision making problems beyond online convex optimization framework. In the OCO framework, the target is to minimise the cumulative loss and confront the adversarial environment. Contrarily, in this thesis, we firstly consider another form of loss, from the cumulative loss in the OCO model to the global loss in online load balancing problems. Next, we release the adversary of the environment, that the loss sequence is structural or there is some assistant information (side information) to the algorithm during the learning process. We attempt to explore these new problems of the online decision making field by reducing these problems to the online linear optimization (OLO) framework.

In Chapter 3, we consider the online load balancing problem. This problem can be seen as a variant of standard expert advice by replacing the cumulative loss with global loss. The global cost function is firstly introduced by Even-Dar et al. [19], and in our thesis, on the one hand, by involving the combined norm and Blackwell approaching game, we give a general reduction for the global cost function with the general norm. In other words, we can reduce the online load balancing problem to two specified online linear optimization problems. On the other hand, for the infinity norm, we propose an efficient algorithm by constructing two efficient reduction process with second order cone programming and linear programming. Our proposed algorithm is efficient and achieves the best known regret bound [46] $O(\sqrt{T \ln N})$.

In Chapter 4, we explore the expert advice problem with noisy low rank loss. Prediction with expert advice is a famous online learning model [13]. Recently, Hazan et al. [29] restrict that the loss sequence obtains the d-rank structure and achieves a regret bound $O(d\sqrt{T})$, when the kernel is unknown. In our thesis, we release this assumption that the loss sequence is corrupted by the noise vector. By approximating the d-rank kernel from the receiving loss vectors during the learning process, we design an algorithm that achieves the regret bound

as $O(\sqrt{T}(d + d^{4/3}(N\epsilon)^{1/3}))$, where ϵ is the L_2-norm bound of noise vector. Our reduction to the OLO model is similar to the reductions of [29], but robust to the noise. Furthermore, in the experiment, our algorithm performs even better than the Hedge algorithm and Hazan's algorithm.

In Chapter 5, we extend the OLO model to the matrix version, as online semi-definite programming. Unlike the previous setting [28, 44], in our thesis, we extend the constraint to the decision set from the trace norm to the $\mathbf{\Gamma}$-trace norm. The trace norm is an extreme case when the $\mathbf{\Gamma}$ is the identity matrix. If we directly utilised the previous algorithm [44], the regret bound is with respect to the size of the matrix N. By utilising the generalised log-determinant regularizer, we can achieve a regret bound irrelevant to the size of matrices, i.e., obtain a tighter regret bound. Moreover, we can reduce the online binary matrix problem with side information [31] to our extended OSDP problem by mistake-driven technique, and we can improve the mistake bound from the previous work with a logarithmic factor. If the underlying binary obtains some latent structure, our mistake bound is optimal, which can recover the lower bound up a constant factor.

Conclusively, reducing the non-OLO problem to the OLO model is an effective method to explore the non-standard OCO games. With the refined reductions, the complex structure in the learning problem can be well represented in the corresponding OLO games. As we stated in the thesis, from the global cost function/global loss to the cooperative environment feedback, the learning model can be reduced to an appropriate OLO game, by selecting the appropriate decision set and loss space. Meanwhile, the reduced OLO game is not always standard. It also requires research on the OLO model itself to ensure that the reduced problem can be effectively solved.

Bibliography

[1] Jacob Abernethy, Peter L Bartlett, and Elad Hazan. Blackwell approachability and no-regret learning are equivalent. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 27–46, 2011.

[2] Jacob D Abernethy. Can we learn to gamble efficiently? In *COLT*, pages 318–319. Citeseer, 2010.

[3] Susanne Albers. Online scheduling. In *Introduction to scheduling*, pages 71–98. CRC Press, 2009.

[4] Yossi Azar. On-line load balancing. In Amos Fiat and Gerhard J Woeginger, editors, *Online Algorithms: The State of the Art*, pages 178–195. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998. ISBN 978-3-540-68311-7. doi: 10.1007/BFb0029569. URL `https://doi.org/10.1007/BFb0029569`.

[5] Yossi Azar, Bala Kalyanasundaram, Serge Plotkin, Kirk R Pruhs, and Orli Waarts. Online load balancing of temporary tasks. In *Workshop on algorithms and data structures*, pages 119–130. Springer, 1993.

[6] Maria-Florina F Balcan and Hongyang Zhang. Noise-tolerant life-long matrix completion via adaptive sampling. In *Advances in Neural Information Processing Systems*, pages 2955–2963, 2016.

[7] Keith Ball et al. An elementary introduction to modern convex geometry. *Flavors of geometry*, 31:1–58, 1997.

[8] Siddharth Barman, Aditya Gopalan, and Aadirupa Saha. Online learning for structured loss spaces. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[9] Melanie Beckerleg and Andrew Thompson. A divide-and-conquer algorithm for binary matrix completion. *Linear Algebra and its Applications*, 601:113–133, 2020.

[10] Aditya Bhaskara, Ashok Cutkosky, Ravi Kumar, and Manish Purohit. Online learning with imperfect hints. In *International Conference on Machine Learning*, pages 822–831. PMLR, 2020.

[11] Aditya Bhaskara, Ashok Cutkosky, Ravi Kumar, and Manish Purohit. Logarithmic regret from sublinear hints. *Advances in Neural Information Processing Systems*, 34, 2021.

[12] David Blackwell et al. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6(1):1–8, 1956.

[13] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, learning, and games*. Cambridge university press, 2006.

[14] Nicolo Cesa-Bianchi and Ohad Shamir. Efficient online learning via randomized rounding. In *Advances in Neural Information Processing Systems*, pages 343–351, 2011.

[15] Paul Christiano. Online local learning via semidefinite programming. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 468–474. ACM, 2014.

[16] Pilu Crescenzi, Giorgio Gambosi, Gaia Nicosia, Paolo Penna, and Walter Unger. On-line load balancing made simple: Greedy strikes back. *Journal of Discrete Algorithms*, 5(1): 162–175, 2007.

[17] Ofer Dekel, Nika Haghtalab, Patrick Jaillet, et al. Online learning with a hint. In *Advances in Neural Information Processing Systems*, pages 5299–5308, 2017.

[18] Mamadou Diop, Sebastian Miron, Anthony Larue, and David Brie. Binary matrix factorization applied to netflix dataset analysis. *IFAC-PapersOnLine*, 52(24):13–17, 2019.

[19] Eyal Even-Dar, Robert Kleinberg, Shie Mannor, and Yishay Mansour. Online learning for global cost functions. In *COLT*, 2009.

[20] Rina Foygel and Nathan Srebro. Concentration-based guarantees for low-rank matrix reconstruction. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 315–340, 2011.

[21] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[22] Claudio Gentile, Mark Herbster, and Stephen Pasteris. Online similarity prediction of networked data from known and unknown graphs. In *Conference on Learning Theory*, pages 662–695, 2013.

[23] Anupriya Gogna and Angshul Majumdar. Balancing accuracy and diversity in recommendations using matrix completion framework. *Knowledge-Based Systems*, 125:83–95, 2017.

[24] Andrew Goldberg, Ben Recht, Junming Xu, Robert Nowak, and Jerry Zhu. Transduction with matrix completion: Three birds with one stone. In *Advances in neural information processing systems*, pages 757–765, 2010.

[25] Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

[26] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

[27] Elad Hazan. Introduction to Online Convex Optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016. URL `http://ocobool.cs.prinston.edu/`.

[28] Elad Hazan, S Kale, and S Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. *SIAM Journal on Computing*, 2016.

[29] Elad Hazan, Tomer Koren, Roi Livni, and Yishay Mansour. Online learning with low rank experts. In *Conference on Learning Theory*, pages 1096–1114, 2016.

[30] Mark Herbster, Stephen Pasteris, and Massimiliano Pontil. Mistake bounds for binary matrix completion. In *Advances in Neural Information Processing Systems*, pages 3954–3962, 2016.

[31] Mark Herbster, Stephen Pasteris, and Lisa Tse. Online matrix completion with side information. *Advances in Neural Information Processing Systems*, 33, 2020.

[32] Dirk Hoeven, Tim Erven, and Wojciech Kotłowski. The many faces of exponential weights in online learning. In *Conference On Learning Theory*, pages 2067–2092, 2018.

[33] Jyrki Kivinen and Manfred K Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *information and computation*, 132(1):1–63, 1997.

[34] Vladimir Koltchinskii, Karim Lounici, Alexandre B Tsybakov, et al. Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion. *The Annals of Statistics*, 39(5):2302–2329, 2011.

[35] Tomer Koren and Roi Livni. Affine-invariant online optimization and the low-rank experts problem. *Advances in Neural Information Processing Systems*, 30:4747–4755, 2017.

[36] Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.

[37] Joon Kwon. Refined approachability algorithms and application to regret minimization with global costs. *Journal of Machine Learning Research*, 22(200):1–38, 2021.

[38] Yaxiong Liu, Kohei Hatano, and Eiji Takimoto. Improved algorithms for online load balancing. In *International Conference on Current Trends in Theory and Practice of Informatics*, pages 203–217. Springer, 2021.

[39] Yaxiong Liu, Xuanke Jiang, Kohei Hatano, and Eiji Takimoto. Expert advice problem with noisy low rank loss. In *Asian Conference on Machine Learning*, pages 1097–1112. PMLR, 2021.

[40] Yaxiong Liu, Ken-ichiro Moridomi, Kohei Hatano, and Eiji Takimoto. An online semi-definite programming with a generalised log-determinant regularizer and its applications. In *Asian Conference on Machine Learning*, pages 1113–1128. PMLR, 2021.

[41] Miguel Sousa Lobo, Lieven Vandenberghe, Stephen Boyd, and Hervé Lebret. Applications of second-order cone programming. *Linear algebra and its applications*, 284(1-3): 193–228, 1998.

[42] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. Adaptive computation and machine learning series. MIT Press, 2012. ISBN 9780262018258.

[43] Marco Molinaro. Online and random-order load balancing simultaneously. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1638–1650. Society for Industrial and Applied Mathematics, 2017.

[44] Ken-ichiro Moridomi, Kohei Hatano, and Eiji Takimoto. Online linear optimization with the log-determinant regularizer. *IEICE Transactions on Information and Systems*, 101(6): 1511–1520, 2018.

[45] Francesco Orabona, Koby Crammer, and Nicolo Cesa-Bianchi. A generalized online mirror descent with applications to classification and regression. *Machine Learning*, 99(3): 411–435, 2015.

[46] Alexander Rakhlin, Karthik Sridharan, and Ambuj Tewari. Online learning: Beyond regret. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 559–594, 2011.

[47] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In *Recommender systems handbook*, pages 1–35. Springer, 2011.

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

[49] Ohad Shamir and Shai Shalev-Shwartz. Collaborative filtering with the trace norm: Learning, bounding, and transducing. In *Proceedings of the 24th Annual Conference on Learning Theory*, pages 661–678, 2011.

[50] Nahum Shimkin. An online convex optimization approach to blackwell's approachability. *The Journal of Machine Learning Research*, 17(1):4434–4456, 2016.

[51] Jing Wang, Jie Shen, Ping Li, and Huan Xu. Online matrix completion for signed link prediction. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pages 475–484, 2017.

[52] Xiao Zhang, Lingxiao Wang, Yaodong Yu, and Quanquan Gu. A primal-dual analysis of global optimality in nonconvex low-rank matrix recovery. In *International conference on machine learning*, pages 5862–5871. PMLR, 2018.

[53] Xiao Zhang, Lingxiao Wang Wang, and Quanquan Gu. A unified framework for non-convex low-rank plus sparse matrix recovery. In *International Conference on Artificial Intelligence and Statistics*, 2018.

[54] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th international conference on machine learning (icml-03)*, pages 928–936, 2003.