Robust and Efficient Prediction with Structural Constraints

森富, 賢一郎

https://doi.org/10.15017/1931929

出版情報：Kyushu University, 2017, 博士（情報科学）, 課程博士
バージョン：
権利関係：

Robust and Efficient Prediction with Structural Constraints

Ken-ichiro Moridomi

February, 2018

Abstract

Many real-world problems can be formulated as problems of predicting and learning objects with some structural constraints. In this thesis we handle two standard models of machine learning problem formulations. One is the online prediction model and the other is the statistical learning model. Online prediction is a theoretical model of repeated processes of making decisions, receiving feedbacks and incurring the loss defined by feedbacks and decisions. The goal of the algorithm is to perform nearly as well as the best decision in hindsight, where the best decision is determined by the feedback sequence. On the other hand, in the statistical learning model, the algorithm is given a set of labeled instances which are generated according to a probability distribution and then predicts a function called the hypothesis which maps an instance to a label. The goal of the algorithm is to minimize the probability that the hypothesis makes a wrong label.

We consider various formulations of machine learning problems with some structural constraints. The motivation to introduce constraints is that many problems can be naturally and/or well formulated as optimization problems whose feasible solutions should satisfy some structural constraints, where by structural constraints we mean combinatorial constraints or algebraic constraints.

For example, a network routing problem is naturally formulated as an online prediction problem where decisions made by the algorithm are restricted to paths or spanning trees of a graph. Another example is a recommendation by ranking, where the decisions are restricted to permutations over the set of items under consideration.

As for the algebraic constraints, we consider the matrix completion problem under the both models of online prediction and statistical learning. The most common application of the matrix completion is the collaborative filtering, which can be viewed as the problem of matrix completion of the preference matrix. One of the most successful method is to put a low rank or a low matrix norm constraint on the space of matrices for the algorithm to choose from [70, 72].

In this thesis we develop the methods of designing and analyzing algorithms for predicting objects with some structural constraints, so that they are robust and efficient. For the robustness, we require algorithms to perform well against the worst-case environment. For the efficiency, we require algorithms to run in polynomial time.

There are four main results in this thesis. First, we consider an online prediction task of matrices. The reduction technique from some online prediction tasks of matrices to online prediction of symmetric positive semi-definite matrices (shortly online SDP problems) [38], and a framework of designing an algorithm [36]were known. The algorithm based framework of [36] makes prediction as minimize the cumulative loss and a convex function called the regularizer simultaneously. We develop a new technique of analyzing algorithms with introducing new notion of the strong convexity of the regularizer. We suggest an algorithm using the log-determinant as the regularizer for a online SDP problem, and analyze it with our new technique. We improve the performance measure of the algorithm and is optimal in some online prediction tasks. Next, we handle the binary matrix completion problem in online and statistical learning model. We introduce the new notion for measuring the difficulty of the data which the algorithm receives, and reduce the online binary matrix completion problem to the online SDP problem. And then apply our algorithm proposed in previous chapter. We improve the upper bound of mistakes that the algorithm will make and has the optimal mistake bound We can apply this algorithm to the statistical learning setting. In the statistical learning setting, our algorithm competes with the Support Vector Machine with the best feature mapping without knowing the feature map. Thirdly, we propose new classes of matrices for the matrix completion problem. The new class of matrices defined by matrix factorization with norm constraints. We give generalization error bounds for these matrix classes. We also show some experimental results. Finally, we develop an algorithm for the Metrical Task System problem (shortly MTS problem) of combinatorial objects. In combinatorial setting, the number of the decision grows exponentially of some domain and the naïve implementation of the algorithms for the MTS problem leads exponential time algorithms. We reduce the bottleneck of Marking algorithm [16] to a sampling problem over combinatorial objects. Combining sampling oracles, our algorithm is the first efficient time algorithm for this problem under the uniform metric. We derive its performance measure and give a proof of the optimality of our algorithm in some specific case of the combinatorial objects.

Acknowledgements

I would like to show my gratitude for everyone's support while studying. First of all I am extremely grateful for Professor Eiji Takimoto. He is my supervisor and the committee chair of this thesis. His deep understanding and strictly theoretical thinking of mathematics, machine learning and computational theory made me grow up a lot. His ideas based on theoretical point of view always helps me and became breakthroughs of my research. I also would like to thank Professor Jun'ichi Takeuchi and Associate Professor Shuji Kijima, who gave me many valuable comments as committee members of this thesis.

I greatly appreciate to Professor Kohei Hatano. He gave me many valuable advice not only for my research but also for my career as a researcher. He also supported me in the programming field. I also appreciate to Professor Masayuki Takeda, Professor Hideo Bannai and Professor Shunsuke Inenaga for their advice in weekly seminars. Their advice greatly improve my presentation skill. Professor Takeda also gave me many valuable advice for my job hunting.

The results in this thesis were published in proceedings of ACML 2014 [59], proceedings of ALT 2016 [60]. The journal version of ACML 2014 with some refinement of analysis will be published in The IEICE Transactions on Information and Systems. Its arXiv e-print is available in [58]. I appreciate all editors, committees, anonymous referees, and publishers.

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

Last, but not least, I really thank my family for their support.

Contents

Abstract **i**

Acknowledgements **iii**

1 Introduction **1**

1.1 Our contributions 2

1.2 Organization 6

2 Preliminaries **7**

2.1 Notations 7

2.2 Statistical learning model 8

2.3 Online learning model 9

2.4 Well-known formulas 11

3 Online linear optimization with the log-determinant regularizer **12**

3.1 Introduction 12

3.2 Problem setting 15

3.3 FTRL and its regret bounds by standard derivations 16

3.4 Online matrix prediction and reduction to online SDP 23

3.5 Main results 26

3.6 Conclusion 36

4 Optimal mistake bounds for binary matrix completion the log-determinant regularization **37**

4.1 Introduction 37

4.2 Related work 38

4.3 Preliminaries 39

4.4 Reduction of the binary matrix prediction 40
4.5 Our algorithm and analysis 44
4.6 Connection to the batch setting 47
4.7 Implementation details 50
4.8 Conclusion 52

5 Tighter generalization error bounds for matrix completion 53
5.1 Introduction 53
5.2 Related work 55
5.3 Problem setting 56
5.4 Generalization bounds for our hypothesis classes 60
5.5 A matching lower bound on the Rademacher complexity 64
5.6 Experimental results 67
5.7 Conclusion 69
5.8 Implementation details 70
5.9 Yet another proof of Lemma 16 70

6 A Combinatorial Metrical Task System Problem under the Uniform Metric 72
6.1 Introduction 72
6.2 Preliminaries 74
6.3 The Marking algorithm 75
6.4 The Weighted Marking algorithm 77
6.5 Conclusion and future work 85

7 Conclusion 86

Chapter 1

Introduction

Machine learning is used in many fields, and there are various problem formulations of machine learning problems. In particular, many real-world problems can be formulated as the problems of predicting and learning objects with some structural constraints. In this thesis we handle two standard models of machine learning problem formulations. One is the online prediction model and the other is the statistical learning model.

Online prediction is a theoretical model of repeated processes of making decisions, receiving feedbacks and incurring the loss defined by feedbacks and decisions. Each repeat is called the round. This type of problem has been extensively studied in the machine learning community for a couple of decades [27, 44, 23]. The goal of the algorithm is to perform nearly as well as the best decision in hindsight, where the best decision is determined by the feedback sequence. In particular, we consider two measures to evaluate the algorithm performance relative to the best decision. One is the *regret* which is defined as the difference of the cumulative loss of the algorithm and that of the best decision, and the other is the *competitive ratio* which is defined as the ratio of them.

On the other hand, in the statistical learning model, the algorithm is given a set of labeled instances (where the label of each instance represents its correct classification) which are generated according to a probability distribution and then predicts a function called the hypothesis which maps an instance to a label. The goal of the algorithm is to minimize the *generalization error*, the probability that the hypothesis makes a wrong label. This model is called the Probably Approximately Correct learning (PAC learning, for short) framework [75]. One of the major issues of the PAC learning framework is to give theories for analyzing the generalization error and has been extensively studied until now [78, 5].

We consider various formulations of machine learning problems with some structural con-

straints. The motivation to introduce constraints is that many problems can be naturally and/or well formulated as optimization problems whose feasible solutions should satisfy some structural constraints, where by structural constraints we mean combinatorial constraints or algebraic constraints.

For example, a network routing problem is naturally formulated as an online prediction problem where decisions made by the algorithm are restricted to paths or spanning trees of a graph. Another example is a recommendation by ranking, where the decisions are restricted to permutations over the set of items under consideration. In these cases, the algorithm is required to choose a combinatorial object as the prediction.

As for the algebraic constraints, we consider a particular problem, the matrix completion, under the both models of online prediction and statistical learning. Intuitively, the matrix completion problem is to predict the whole entries of a matrix from partial entries given. The most common application of the matrix completion is the collaborative filtering, where the task is to predict the preferences of m users over n items from partial information, which can be viewed as the problem of matrix completion of the $m \times n$ preference matrix. One of the most successful method is to put a low rank or a low matrix norm constraint on the space of matrices for the algorithm to choose from [70, 72].

Our aim in this thesis is to develop the methods of designing and analyzing algorithms for predicting objects with some structural constraints, so that they are *robust* and *efficient*. For the robustness, we require algorithms to perform well against the worst-case environment. For the efficiency, we require algorithms to run in polynomial time.

1.1 Our contributions

In this section, we briefly describe the problems we consider in this thesis and show related work and our contribution.

1.1.1 Online semi-definite programming problem

Modeling with matrices is more natural than modeling with vectors for some applications such as ranking and recommendation tasks [20, 39, 42]. Hazan et.al. [38] show that some online prediction problem of matrices related to ranking and recommendation can be reduced to an online prediction problem of *symmetric positive semi-definite matrices* under linear loss func-

tions. Thus, we can focus on the online prediction problems for symmetric positive semi-definite matrices, which we call the online semi-definite programming (online SDP) problems. In the online SDP problem, the algorithm repeats the following round: first chooses a symmetric positive semi-definite matrix from an bounded set, then receives a symmetric matrix from an environment and finally incurs the loss defined as the inner product of chosen matrix and received matrix. The goal of the algorithm is to minimize the regret, the difference between the cumulative loss of the algorithm and that of the best fixed matrix in hindsight in the bounded set.

On the online prediction tasks, there is a standard framework called Follow the Regularized Leader (FTRL) for designing and analyzing algorithms [36, 65, 68, 37], where we need to choose as a parameter an appropriate function called regularizer.

Hazan et.al.p̃ropose an algorithm for the online SDP problems which employ the von Neumann entropy as the regularizer, and shows that the $O(\sqrt{T\beta\tau \ln N})$ of the upper bound of the regret where β, τ are bounding parameter of the algorithm's predictions, N is the size of the matrices and T is the number of rounds. However, their algorithm turns out to be sub-optimal [38].

Another possible choice of the regularizer is the log-determinant. Christiano uses the log-determinant as the regularizer and considers a very specific sub-class of online SDP problems and succeeds to improve the regret bound for a particular application problem [21]. But the problems he examines do not cover the whole class of online prediction tasks which admit the reduction to online SDP problems.

In this thesis, we propose an algorithm for the online SDP problems which uses the log-determinant regularization. We extend the analysis of Christiano in [21] and develop a new technique of deriving regret bounds by exploiting the new notion of the strong convexity of the regularizer. The analysis in [21] is not explicitly stated as in a general form and focused on a very specific case. We improve the regret bound for online SDP problem when feedback matrices are sparse, and show the $O(\sqrt{T\beta\tau})$ of the upper bound of the regret which is optimal for some application problems proposed in [38].

1.1.2 Online binary matrix completion problem

Learning preferences of users over a set of items is an important task in recommendation systems. In particular, the collaborative filtering approach is known to be quite effective and popular [66, 52, 80]. The approach is formulated as the *matrix completion problem*, where, an

underlying matrix represents user's preferences of items, only some entries of the matrix are given and the goal is to predict rest of preferences. Under the natural assumption that captures the low rank property of the matrix, a lot of work have been done in various settings, say, the statistical i.i.d. setting [71, 31, 69, 56] and the online learning setting [20, 38, 59, 42]

The online binary matrix completion problem posed by Herbseter et al. [39] is defined in the following way. The problem is defined as a series of round. For each round, the algorithm receives a pair of a user and an item, then predicts that the user likes given item or not and finally incurs the loss if the prediction is wrong. The goal of the algorithm is to minimize the total loss (i.e. the number of errors).

Herbster et al. [39] considered an online version of the binary matrix completion problem and give an algorithm. Their analysis introduce the margin complexity of the matrix, an different aspects of the complexity of the matrix from the matrix rank. Their algorithm enjoys the mistake bounds of $O(\frac{(m+n)\log(m+n)}{\gamma^2})$ for the problem, where the m,n are the numbers of users and items respectively and γ is the margin complexity. However, their mistake bound is not optimal yet.

For the online binary matrix completion problem, we propose an FTRL-based algorithm with the log-determinant regularizer and prove mistake bounds of the algorithm under a natural assumption of [39]. We develop a sharper reduction technique from the online binary matrix completion to the online SDP problem [38] and apply of a modified notion of the strong convexity mentioned in previous subsection for the log-determinant [59, 58].

The mistake bounds of our algorithm $O(\frac{m+n}{\gamma^2})$ are optimal,

We also apply our online algorithm in the statistical learning model by a standard online to batch conversion framework (see, e.g., [57]), and derive a generalization error bound. In this setting, we also can drop the logarithmic factor from the result of [39]. Our generalization error bound is similar to the known margin-based bound obtained by the best kernel in hindsight. Additionally, we introduce the standard notion of the margin loss [57] to measure the difficulty of the given sample in the both settings. This is a tighter measure than the margin error.

1.1.3 Matrix factorization under norm constraints

We studied the matrix completion problem in statistical machine learning setting, that is, learning a user-item rating matrix from given sample. The sample consists partial entries of the matrix. Different from previous result, in this problem, the algorithm requires to predict user's

ratings in the real value (not binary). A common assumption in the previous work is that the true matrix $\mathrm{X} \in \mathbb{R}^{m \times n}$ can be well approximated by a matrix of low rank matrix $\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^{\mathsf{T}}$ (or low trace norm, as a convex relaxation of the rank constraint). Generalization ability of algorithms (such as the empirical risk minimization) using low rank or low trace norm matrices is intensively studied in the literature (see, e.g., [70, 71, 72, 31]).

Recently, further additional constraints on the class of hypothesis matrices turns out to be effective in practice. In particular, a major approach is to impose the constraints that U and V are non-negative with bounded L_2 norm [51, 48, 33, 79, 54, 28]. Such a decomposition is called the non-negative matrix factorization (NMF, for short). A typical scheme of the NMF approach is the empirical risk minimization with the norm regularization [48, 33, 28]. Despite the empirical success of the NMF approach, no theoretical justification has been given.

In this thesis, we consider different but closely related classes of hypothesis matrices and give generalization bounds of these classes. Our bounds are of $O(\sqrt{(nK + m \log K)/T})$, where T is the sample size and K is the rank of $\hat{\mathrm{X}}$. These bounds improve the previously known bound $\tilde{O}(\sqrt{(n+m)K/T})$ that is derived for the class where only the low-rank constraint is imposed [70]. Therefore, our results would give theoretical evidence for the empirical success of the NMF. However, our new bounds hold even when U and V have negative values in some components. This result suggests that our analysis may not yet fully capture the property of the non-negativity constraint, or the empirical success of the NMF may not rely on the non-negativity very much but mostly on the regularization.

Our technical contributions are twofold. The first one is that we develop a new technique for bounding the Rademacher complexity to derive generalization bounds. The second one is that we prove a matching lower bound of the Rademacher complexity of the first hypothesis class. This means that our generalization bounds are tightest among those derived from the Rademacher complexity argument. There are few results in the literature on deriving lower bounds of the Rademacher complexity.

1.1.4 Combinatorial metrical task system problems

The metrical task system (MTS) problem is a scheme of the online learning model, it repeats the round (making decision and receive feedbacks) but it impose cost for changing the algorithm's decision. The MTS problem is defined as a repeated game between the player and the adversary. Consider a fixed set of states, a metric over the state set and a initial state. For each round,

the adversary reveals a (processing) cost function, then the algorithm chooses a state from the state set and finally the algorithm incurs the loss defined as the sum of the processing cost and the distance from previous state to current state. The goal of the algorithm is minimizing the cumulative (processing and moving) cost. The performance of the algorithm is measured by the competitive ratio, that is, the ratio of the cumulative cost of the algorithm to the cumulative cost of the best fixed sequence of states in hindsight. In the setting that the state set consists of n states, there are many existing works on the MTS [16, 40, 8, 29, 9] problem. In particular, for the uniform metric (which is defined as 1 if states are different and otherwise 0), the MTS problem is well studied [16, 8, 40, 1].

When we consider the situation where the state set is a combinatorial set from $\{0, 1\}^d$, the computational issue arises. In general, for typical combinatorial sets, the size could be exponential in the dimension size d as well and straightforward implementations of known algorithms for the MTS problem take exponential time as well since time complexity of these algorithms is proportional to the size $n = O(2^d)$.

In this thesis, for the combinatorial MTS problem under the uniform metric, we propose a modification of the Marking algorithm [16], which employs an exponential weighting scheme. We prove that our algorithm retains $O(\log n)$ competitive ratio for the standard MTS problem with n states. Combining with efficient sampling techniques w.r.t. exponential weights on combinatorial objects [73, 19, 41, 2], our algorithm works efficiently for various classes of combinatorial sets. We also give a lower bound of the competitive ratio in the case of that the state set is k-set. This result shows that our algorithm is optimal in this case.

1.2 Organization

The rest of this thesis is organized as follows: In Chapter 3 we consider an online prediction problem of symmetric positive semi-definite matrices and propose a novel method for analyzing online algorithms by introducing a new notion of strong convexity. In Chapter 4, we apply the log-determinant regularization technique in the binary matrix completion problem, and show the mistake bound. We also apply our online algorithm to the statistical learning setting and derive a generalization error bound. In Chapter 5, we consider the matrix completion problem and give a generalization error bound of the class defined by the matrix factorization with norm constraints. In Chapter 6, we propose the first efficient algorithm for metrical task system problems over combinatorial objects under the uniform metric.

Chapter 2

Preliminaries

In this chapter, we define some notations to be used throughout this thesis and introduce three learning models, the statistical learning model, the online learning model.

2.1 Notations

In this thesis, matrices are denoted by roman capital letters. Let $\mathbb{R}^{m\times n}$, $\mathbb{S}^{N\times N}$, $\mathbb{S}_+^{N\times N}$ be the set of $m \times n$ matrices, the set of $N \times N$ symmetric matrices, and the set of $N \times N$ symmetric positive semi-definite matrices, respectively.

For a matrix X, let σ_i be its i-th largest singular value. We write the trace of a matrix X as $\mathrm{Tr}(\mathrm{X})$ and the determinant as $\det(\mathrm{X})$. We write its trace norm as $\|\mathrm{X}\|_{\mathrm{Tr}} = \sum_i \sigma_i$, spectral norm as $\|\mathrm{X}\|_{\mathrm{Sp}} = \max_i \sigma_i$, and Frobenius norm as $\|\mathrm{X}\|_{\mathrm{Fr}} = \sqrt{\sum_i \sigma_i^2}$. For the matrix X, we write its largest eigenvalue and smallest eigenvalue by $\lambda_{\max}(\mathrm{X})$ and $\lambda_{\min}(\mathrm{X})$, respectively.

We write the identity matrix as E. The notion $\mathrm{X} \succeq 0$ means X is positive semi-definite. For any positive integer m, we write $[m] = \{1, 2, \ldots, m\}$. We define the vectorization of a matrix $\mathrm{X} \in \mathbb{R}^{m\times n}$ as

$$\mathrm{vec}(\mathrm{X}) = (X_{*,1}^\mathsf{T}, X_{*,2}^\mathsf{T}, \ldots, X_{m,*}^\mathsf{T})^\mathsf{T},$$

where $X_{*,i}$ is the i-th column of X. For $m\times n$ matrices X and L, $\mathrm{X}\bullet\mathrm{L} = \sum_{i=1}^m \sum_{j=1}^n X_{i,j}L_{i,j} = \mathrm{vec}(\mathrm{X})^\mathsf{T}\mathrm{vec}(\mathrm{L})$ is the Frobenius inner product.

For a vector $\boldsymbol{x} \in \mathbb{R}^N$, $\mathrm{diag}(\boldsymbol{x})$ denote the $N \times N$ diagonal matrix X such that $X_{i,i} = x_i$. For $m \times n$ matrices X and L, $\mathrm{X} \bullet \mathrm{L} = \sum_{i,j}^{m,n} X_{i,j}L_{i,j} = \mathrm{vec}(\mathrm{X})^\mathsf{T}\mathrm{vec}(\mathrm{L})$ is the Frobenius inner product. For a matrix X, we write its i-th row as $\mathrm{X}_{i,*}$ and j-th column as $\mathrm{X}_{*,j}$.

For a differentiable function $R : \mathbb{R}^{m\times n} \to \mathbb{R}$, its gradient $\nabla R(\mathrm{X})$ is the $m\times n$ matrix whose

(i,j)-th component is $\frac{\partial R(\mathrm{X})}{\partial X_{i,j}}$, and its Hessian [55, 30] $\nabla^2 R(\mathrm{X})$ is the $mn \times mn$ matrix defined by

$$(\nabla^2 R(\mathrm{X}))_{(j-1)m+i,(l-1)m+k} = \frac{\partial^2 R(\mathrm{X})}{\partial X_{k,l} \partial X_{i,j}}.$$

We denote the Kronecker product of two matrices $\mathrm{A} \in \mathbb{R}^{M_1 \times N_1}$ and $\mathrm{B} \in \mathbb{R}^{M_2 \times N_2}$ as $\mathrm{A} \otimes \mathrm{B} \in \mathbb{R}^{M_1 M_2 \times N_1 N_2}$, which is defined as $(\mathrm{A} \otimes \mathrm{B})_{M_2(i-1)+j,N_2(k-1)+l} = A_{i,k} B_{j,l}$. We use the notation $\mathrm{A} \boxtimes \mathrm{B} \in \mathbb{R}^{M_1 M_2 \times N_1 N_2}$ as the box product of A and B, which is introduced by [30] and defined as $(\mathrm{A} \otimes \mathrm{B})_{M_2(i-1)+j,N_1(k-1)+l} = A_{i,l} B_{j,k}$. These products have following properties [30]:

$$(\mathrm{A} \otimes \mathrm{B})\mathrm{vec}(\mathrm{X}) = \mathrm{vec}(\mathrm{B}\mathrm{X}\mathrm{A}^\mathsf{T}), \tag{2.1}$$

$$(\mathrm{A} \boxtimes \mathrm{B})\mathrm{vec}(\mathrm{X}) = \mathrm{vec}(\mathrm{B}\mathrm{X}^\mathsf{T}\mathrm{A}^\mathsf{T}). \tag{2.2}$$

For a differentiable convex function R, the Bregman divergence with respect to R is defined as

$$B_R(\mathrm{X}, \mathrm{Y}) = R(\mathrm{X}) - R(\mathrm{Y}) - \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}).$$

2.2 Statistical learning model

The statistical learning model is one of the most important model in machine learning. In this model, the algorithm given a set of randomly generated data called the sample set, and then required to produce a function called as the hypothesis which maps a data to the label. The performance of the algorithm is measured by the generalization error of the hypothesis that the algorithm produced. Intuitively, the generalization error measures how well the hypothesis predicts labels for unseen data generated according to same distribution which generates the sample set.

2.2.1 Problem setting

Let $\mathcal{X}$ and $\mathcal{Y}$ be the input space and the label space, respectively. Let a sequence

$$\mathcal{S} = ((\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_T, y_T))$$

be a sample set where each $(x_t, y_t) \in \mathcal{X} \times \mathcal{Y}$ for $t \in [T]$ are generated independently according to some unknown distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$. We call T as the size of $\mathcal{S}$. The algorithm is given a sample $\mathcal{S}$ and then required to produce a function $h \in \mathcal{H}$ maps $\mathcal{X}$ to $\mathcal{Y}$. We call h

the hypothesis and $\mathcal{H}$ the hypothesis set. The goal of the algorithm is to produce $h \in \mathcal{H}$ that minimizes the generalization error as small as possible.

Let $\ell : \mathcal{Y} \times \mathcal{Y}$ be a loss function, then the generalization error is defined as follows:

Definition 1 (Generalization error)**.** *For a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ and distribution $\mathcal{D}$ The generalization error of h is*

$$L(h) = \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\ell(h(\boldsymbol{x}), y)].$$

If $\ell(x, y) = \mathbf{1}_{[x\neq y]}$ then $L(h) = \Pr_{(\boldsymbol{x},y)\sim\mathcal{D}}[h(\boldsymbol{x}) \neq y]$. We cannot compute the generalization error due to unknown distribution $\mathcal{D}$, but one can estimate it by the empirical error:

Definition 2 (Empirical error)**.** *For a hypothesis $h : \mathcal{X} \to \mathcal{Y}$ and a sample $\mathcal{S}$ of size T, the empirical error of h is*

$$\hat{L}(h) = \frac{1}{T}\sum_{t=1}^{T} \ell(h(\boldsymbol{x}), y).$$

Thus $\mathbb{E}_{\mathcal{S}\sim\mathcal{D}^T}[\hat{L}(h)] = L(h)$.

2.3 Online learning model

The online learning model consists a series of repeated games between the algorithm and an (adversarial) environment. We study two types under this model, the online convex optimization and the metrical task system.

2.3.1 Online convex optimization

Online convex optimization is a repeated game between an algorithm and an adversarial environment.

Problem setting

In the online convex optimization, the following protocol proceeds; For each round $t = 1, \ldots, T$, the algorithm (i) predicts $\boldsymbol{x}_t \in \mathcal{K}$, (ii) receives a convex function $\ell_t : \mathcal{K} \to \mathbb{R}$ and (iii) suffers the loss $\ell_t(\boldsymbol{x}_t)$. The goal of the algorithm is to minimize the cumulative loss $\sum_{t=1}^{T} \ell_t(\boldsymbol{x}_t)$. In this thesis, in some cases, the loss function is the linear function $\ell_t(\boldsymbol{x}_t) = \boldsymbol{\ell}_t \cdot \boldsymbol{x}_t$. We regard the algorithm receives $\boldsymbol{\ell}_t$ instead of ℓ_t and this problem was called the online linear optimization.

The regret is most standard measure of the performance of the algorithm.

Definition 3. *Let $\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_T \in \mathcal{K}$ be the predictions of the algorithm. The regret of the algorithm is*

$$Reg(T, \mathcal{K}, \mathcal{L}, \boldsymbol{x}^\star) = \sum_{t=1}^{T} \ell_t(\boldsymbol{x}_t) - \sum_{t=1}^{T} \ell_t(\boldsymbol{x}^\star),$$

where $\boldsymbol{x}^\star \in \mathcal{K}$ is a competitor.

Usually, the best offline solution $\boldsymbol{x}^\star = \arg\min_{\boldsymbol{x}\in\mathcal{K}} \sum_{t=1}^{T} \ell_t(\boldsymbol{x})$ was used as the competitor.

2.3.2 Metrical task system

The metrical task system takes into account a cost of changing decisions.

Problem setting

The metrical task system is defined by a set $\mathcal{C}$ and a metric function $\delta : \mathcal{C} \times \mathcal{C} \to \mathbb{R}$. In the metrical task system, first the adversary chooses a task sequence $\sigma = (\ell_1, \ell_2, \ldots, \ell_T)$, where each $\ell_t : \mathcal{C} \to \mathbb{R}$ is called a loss function. For a given initial state $\boldsymbol{c}_0 \in \mathcal{C}$, in each round $t = 1, \ldots, T$, the algorithm (i) receives $\ell_t : \mathcal{K} \to \mathbb{R}$ and (ii) chooses $\boldsymbol{c}_t \in \mathcal{C}$, (iii) suffers the cost $\ell_t(\boldsymbol{x}_t) + \delta(\boldsymbol{c}_{t-1}, \boldsymbol{c}_t)$. The first term $\ell_t(\boldsymbol{c}_t)$ of the cost is called the processing cost at round t, and the second term $\delta(\boldsymbol{c}_t, \boldsymbol{c}_{t-1})$ is called the moving cost at round t.

For a task sequence σ, the cumulative cost of an algorithm A is defined as

$$\mathrm{cost}_A(\sigma) = \sum_{t=1}^{T} (\ell_t(\boldsymbol{c}_t) + \delta(\boldsymbol{c}_t, \boldsymbol{c}_{t-1})),$$

and the cumulative cost of the best offline solution is defined as

$$\mathrm{cost}_{\mathrm{OPT}}(\sigma) = \min_{(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, \ldots, \boldsymbol{c}_T^*) \in C^T} \sum_{t=1}^{T} (\ell_t(\boldsymbol{c}_t^*) + \delta(\boldsymbol{c}_t^*, \boldsymbol{c}_{t-1}^*)).$$

We measure the performance of algorithm A by its competitive ratio, which is defined as

$$\mathrm{CR}(\sigma) = \frac{\mathbb{E}[\mathrm{cost}_A(\sigma)]}{\mathrm{cost}_{\mathrm{OPT}}(\sigma)},$$

where the expectation is with respect to the internal randomness of A. The goal of the algorithm is to minimize the worst case competitive ratio $\max_\sigma \mathrm{CR}(\sigma)$.

2.4 Well-known formulas

In this section we show well-known mathematical theorems we use in this thesis.

Definition 4 (Martingale difference)**.** *A sequence of random variables $V_1, V_2, \ldots$ is a martingale difference sequence with respect to $X_1, X_2, \ldots$ if for all $t > 0$, V_t is a function of $X_1, \ldots, X_t$ and $\mathbb{E}[V_{t+1}|X_1, \ldots, X_t] = 0$.*

Theorem 1 (Azuma's inequality)**.** *Let $V_1, V_2, \ldots$ be a martingale difference sequence with respect to the random variables $X_1, X_2, \ldots$. Assume that for all $t > 0$, there is a constant c_t and random variable Z_t which is a function of $X_1, \ldots, X_{t-1}$ such that $Z_t \leq V_t \leq Z_t + c_t$. Then, for all $\epsilon > 0$ and T, following inequalities hold:*

$$\begin{aligned} \Pr[\sum_{t=1}^{T} V_t \geq \epsilon] &\leq e^{-2\epsilon^2/\sum_{t=1}^{T} c_t^2}, \\ \Pr[\sum_{t=1}^{T} V_t \leq -\epsilon] &\leq e^{-2\epsilon^2/\sum_{t=1}^{T} c_t^2}. \end{aligned}$$

Theorem 2 (McDirmid's inequality)**.** *Let $X_1, \ldots, X_T \in \mathcal{X}^T$ be a set of $T \geq 1$ independent random variables and assume that $\exists c_1, \ldots, c_T > 0$ such that $f : \mathcal{X}^T \to \mathbb{R}$ satisfies*

$$|f(x_1, \ldots, x_t, \ldots, x_T) - f(x_1, \ldots, x_t', \ldots, x_T)| \leq c_t$$

for all $t \in [T]$ and any points $x_1, \ldots, x_T, x_t'$.

Then the following inequality holds:

$$\begin{aligned} \Pr[f(X_1, \ldots, X_T) - \mathbb{E}[f(X_1, \ldots, X_T)] \geq \epsilon] &\leq e^{-2\epsilon^2/\sum_{t=1}^{T} c_t^2}, \\ \Pr[f(X_1, \ldots, X_T) - \mathbb{E}[f(X_1, \ldots, X_T)] \leq -\epsilon] &\leq e^{-2\epsilon^2/\sum_{t=1}^{T} c_t^2}. \end{aligned}$$

Chapter 3

Online linear optimization with the log-determinant regularizer

3.1 Introduction

Online prediction is a theoretical model of repeated processes of making decisions and receiving feedbacks, and has been extensively studied in the machine learning community for a couple of decades [27, 44, 23]. Typically, decisions are formulated as vectors in a fixed set called the decision space and feedbacks as functions that define the losses for all decision vectors. Recently, much attention has been paid to a more general setting where decisions are formulated as matrices, since it is more natural for some applications such as ranking and recommendation tasks [20, 39, 42].

Take the online collaborative filtering as an example. The problem is formulated as in the following protocol: Assume we have a fixed set of n users and a fixed set of m items. For each round $t = 1, 2, \ldots, T$, the following happens. (i) The algorithm receives from the environment a user-item pair (i_t, j_t), (ii) the algorithm predicts how much user i_t likes item j_t and chooses a number x_t that represents the degree of preference, (iii) the environment returns the true evaluation value y_t of the user i_t for the item j_t, and then (iv) the algorithm suffers loss defined by the prediction value x_t and the true value y_t, say, $(x_t - y_t)^2$. Note that, (iii) and (iv) in the protocol above can be generalized in the following way: (iii) the environment returns a loss function ℓ_t, say $\ell_t(x) = (x - y_t)^2$, and (iv) the algorithm suffers loss $\ell_t(x_t)$. The goal of the algorithm is to minimize the cumulative loss, or more formally, to minimize the *regret*, which is the most standard measure in online prediction. The regret is the difference between the cumulative loss of the algorithm and that of the best fixed prediction policy in

some policy class. Note that the best policy is determined in hindsight, i.e., it depends on the whole feedback sequence. Now we claim that the problem above can be regarded as a matrix prediction problem: the algorithm chooses (before observing the pair (i_t, j_t)) the prediction values for all pairs as an $m \times n$ matrix, although only the (i_t, j_t)-th entry is used as the prediction. In this perspective, the policy class is formulated as a restricted set of matrices, say, the set of matrices of bounded trace norm, which is commonly used in collaborative filtering [71, 56, 69, 47]. Moreover, we can assume without loss of generality that the prediction matrices are also chosen from the policy class. So, the policy class is often called the decision space.

In most application problems including the online collaborative filtering, the matrices to be predicted are not square, which makes the analysis difficult. Hazan et al. [38] show that any online matrix prediction problem formulated as in the protocol above can be reduced to an online prediction problem where the decision space consists of symmetric positive semi-definite matrices under linear loss functions. A notable property of the reduction is that the loss functions of the reduced problem are the inner product with sparse loss matrices, where only at most 4 entries are non-zero. Thus, we can focus on the online prediction problems for symmetric positive semi-definite matrices, which we call the online semi-definite programming (online SDP) problems. In particular we are interested in the case where the problems are obtained by the reduction, which we call the online *sparse* SDP problems. Thanks to the symmetry and positive semi-definiteness of the decision matrices and the sparseness of the loss matrices, the problem becomes feasible and Hazan et al. propose an algorithm for the online sparse SDP problems, by which they give regret bounds for various application problems including the online max-cut, online gambling, and the online collaborative filtering [38]. Unfortunately, however, all these bounds turn out to be sub-optimal.

In this chapter, we propose an algorithm for the online sparse SDP problems by which we achieve optimal regret bounds for those application problems.

To this end, we employ a standard framework called Follow the Regularized Leader (FTRL) for designing and analyzing algorithms [36, 65, 68, 37], where we need to choose as a parameter an appropriate regularization function (or regularizer) to obtain a good regret bound. Hazan et al. use the von Neumann entropy (or sometimes called the matrix negative entropy) as the regularizer to obtain the results mentioned above [38], which is a generalization of Tsuda et al. [74]. Another possible choice is the log-determinant regularizer, whose Bregman divergence is so called the LogDet divergence. There are many applications of the LogDet divergence such as metric learning [26], learning of low-rank kernels [50], clustering of multivariate Gaussians

[25], and Gaussian graphical models [67]. However, the log-determinant regularizer is less popular in online prediction and it is unclear how to derive general and non-trivial regret bounds when using the FTRL with the log-determinant regularizer, as posed as an open problem in [74]. Indeed, Davis et al. apply the FTRL with the log-determinant regularizer for square loss and give a cumulative loss bound [26], but it contains a data-dependent parameter and the regret bound is still unclear. Christiano considers a very specific sub-class of online sparse SDP problems and succeeds to improve the regret bound for a particular application problem, the online max-cut problem [21]. But the problems he examines do not cover the whole class of online sparse SDP problems and hence his algorithm cannot be applied to the online collaborative filtering, for instance.

We improve regret bounds for online sparse SDP problems by analyzing the FTRL with the log-determinant regularizer. In particular, our contributions are summarized as follows.

1. We give a non-trivial regret bound of the FTRL with the log-determinant regularizer for a general class of online SDP problems. Although the bound seems to be somewhat loose, it gives a tight bound when the matrices are diagonal (which corresponds to the vector predictions).

2. We extend the analysis of Christiano in [21] and develop a new technique of deriving regret bounds by exploiting the property of strong convexity of the regularizer with respect to the loss matrices. The analysis in [21] is not explicitly stated as in a general form and focused on a very specific case where the loss matrices are block-wise sparse.

3. We improve the regret bound for the online sparse SDP problems, by which we give optimal regret bounds for the application problems, namely, the online max-cut, online gambling, and the online collaborative filtering.

4. We apply the results to the case where the decision space consists of vectors, which can be reduced to online matrix prediction problems where the decision space consists of diagonal matrices. In this case, the general regret bound mentioned in 1 also gives tight regret bound.

3.2 Problem setting

We first describe the problem setting: the online semi-definite programming problem (online SDP problem, for short).

3.2.1 Online SDP problem

We consider an online linear optimization problem over symmetric semi-definite matrices, which we call the online SDP problem. The problem is specified by a pair $(\mathcal{K}, \mathcal{L})$, where $\mathcal{K} \subseteq \mathbb{S}_+^{N \times N}$ is a convex set of symmetric positive semi-definite matrices and $\mathcal{L} \subseteq \mathbb{S}^{N \times N}$ is a set of symmetric matrices. The set $\mathcal{K}$ is called the decision space and $\mathcal{L}$ the loss space. The online SDP problem $(\mathcal{K}, \mathcal{L})$ is a repeated game between the algorithm and the adversary (i.e., an environment that may behave adversarially), which is described as the following protocol.

In each round $t = 1, 2, \ldots, T$, the algorithm

1. chooses a matrix $\mathrm{X}_t \in \mathcal{K}$,
2. receives a loss matrix $\mathrm{L}_t \in \mathcal{L}$ from the adversary, and
3. suffers the loss $\mathrm{X}_t \bullet \mathrm{L}_t$.

The goal of the algorithm is to minimize the regret $Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star)$, defined as

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star) = \sum_{t=1}^{T} \mathrm{L}_t \bullet \mathrm{X}_t - \sum_{t=1}^{T} \mathrm{L}_t \bullet \mathrm{X}^\star.$$

where $\mathrm{X}^\star \in \mathcal{K}$ is some competitor matrix.

If the competitor is the best matrix in the decision set $\mathcal{K}$ that minimizes the cumulative loss i.e. $\mathrm{X}^\star = \arg\min_{\mathrm{X} \in \mathcal{K}} \sum_{t=1}^{T} \mathrm{L}_t \bullet \mathrm{X}$, the matrix $\mathrm{X}^\star$ is called the best offline matrix. In this case we abbreviate the argument $\mathrm{X}^\star$ and write $Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) = Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star)$.

3.2.2 Online linear optimization over vectors

The online SDP problem is a generalization of the online linear optimization problem over vectors, which is a more standard problem setting in the literature. For the "vector" case, the problem is described as the following protocol:

In each round $t = 1, \cdots, T$, the algorithm

1. chooses $\boldsymbol{x}_t \in \mathcal{K} \subset \mathbb{R}^N_+$,
2. receives $\boldsymbol{\ell}_t \in \mathcal{L} \subset \mathbb{R}^N$ from the adversary, and
3. suffers the loss $\boldsymbol{x}_t^\mathsf{T} \boldsymbol{\ell}_t$.

It is easy to see that the problem is equivalent to the online SDP problem $(\mathcal{K}'\mathcal{L}')$ where $\mathcal{K}' = \{\mathrm{diag}(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{K}\}$ and $\mathcal{L}' = \{\mathrm{diag}(\boldsymbol{\ell}) \mid \boldsymbol{\ell} \in \mathcal{L}\}$. So, all the results for the online SDP problem can be applied to the online linear optimization over vectors.

3.3 FTRL and its regret bounds by standard derivations

Follow the Regularized Leader (FTRL) is a standard framework for designing algorithms for a wide class of online optimizations (see, e.g., [68]). To employ the FTRL, we need to specify a convex function $R : \mathcal{K} \to \mathbb{R}$ called the regularization function, or simply the regularizer. For the online SDP problem $(\mathcal{K}, \mathcal{L})$, the FTRL with regularizer R suggests to choose a matrix $\mathrm{X}_t \in \mathcal{K}$ as the decision at each round t according to

$$\mathrm{X}_{t+1} = \arg\min_{\mathrm{X} \in \mathcal{K}} \Big(R(\mathrm{X}) + \eta \sum_{s=1}^{t} \mathrm{L}_s \bullet \mathrm{X} \Big), \tag{3.1}$$

where $\eta > 0$ is a constant called the learning rate. Note that we adopt $\mathrm{X}_1 = \arg\min_{\mathrm{X} \in \mathcal{K}} R(\mathrm{X})$, naturally induced by this rule. Throughout the thesis, we assume for simplicity that all the regularizers R are differentiable.

The FTRL has two closely related or equivalent formulation, which has the different form of from (3.1), and called as the Online Mirror Descent (OMD for short). Which is

$$\tilde{\mathrm{X}}_{t+1} = (\nabla R)^{-1}(\nabla R(\tilde{\mathrm{X}}_t) - \eta \mathrm{L}_t) \qquad \text{and} \qquad \mathrm{X}_{t+1} = \arg\min_{\mathrm{X} \in \mathcal{K}} B_R(\mathrm{X}, \tilde{\mathrm{X}}_{t+1}), \tag{3.2}$$

$$\tilde{\mathrm{X}}_{t+1} = (\nabla R)^{-1}(\nabla R(\mathrm{X}_t) - \eta \mathrm{L}_t) \qquad \text{and} \qquad \mathrm{X}_{t+1} = \arg\min_{\mathrm{X} \in \mathcal{K}} B_R(\mathrm{X}, \tilde{\mathrm{X}}_{t+1}), \tag{3.3}$$

where B_R is the Bregman divergence with respect to R. The algorithm defined by (3.3) is called the lazy version and (3.3) is called the agile version [37]. We can say that if the loss function is linear then the lazy version of OMD is equivalent to the FTRL in (3.1).

Proposition 3 ([37]). *Let B_R be the Bregman divergence with respect to the convex function R. For any online SDP problem $(\mathcal{K}, \mathcal{L})$, Let $\tilde{\mathrm{X}}_{t+1} = (\nabla R)^{-1}(\nabla R(\tilde{\mathrm{X}}_t) - \eta \mathrm{L}_t)$ where $(\nabla R)^{-1}$ is the*

inverse function of ∇R*. Then*

$$\arg\min_{\mathrm{X}\in\mathcal{K}} R(\mathrm{X}) + \eta\sum_{s=1}^{t}\mathrm{L}_t \bullet \mathrm{X} = \arg\min_{\mathrm{X}\in\mathcal{K}} B_R(\mathrm{X}, \tilde{\mathrm{X}}_{t+1}). \tag{3.4}$$

Proof. By the update rule of $\tilde{\mathrm{X}}_{t+1}$, we have

$$\nabla R(\tilde{\mathrm{X}}_{t+1}) = \nabla R(\tilde{\mathrm{X}}_t) - \eta\mathrm{L}_t = \nabla R\left((\nabla R)^{-1}(\nabla R(\tilde{\mathrm{X}}_{t-1}) - \eta\mathrm{L}_{t-1})\right) - \eta\mathrm{L}_t = -\eta\sum_{s=1}^{t}\mathrm{L}_s.$$

Then

$$\begin{aligned}
\arg\min_{\mathrm{X}\in\mathcal{K}} B_R(\mathrm{X}, \tilde{\mathrm{X}}_{t+1}) &= \arg\min_{\mathrm{X}\in\mathcal{K}} R(\mathrm{X}) - R(\tilde{\mathrm{X}}_{t+1}) - \nabla R(\tilde{\mathrm{X}}_{t+1}) \bullet (\mathrm{X} - \tilde{\mathrm{X}}_{t+1}) \\
&= \arg\min_{\mathrm{X}\in\mathcal{K}} R(\mathrm{X}) - \nabla R(\tilde{\mathrm{X}}_{t+1}) \bullet \mathrm{X} \\
&= \arg\min_{\mathrm{X}\in\mathcal{K}} R(\mathrm{X}) + \eta\sum_{s=1}^{t}\mathrm{L}_s \bullet \mathrm{X}.
\end{aligned}$$

□

It is well known that the agile version of OMD has the same bound of the regret as the FTRL or the lazy OMD.

There is a well-known derivation of the regret bound when R is strongly convex with respect to some norm and the loss is bounded by its dual norm. The strongly convexity is defined as follows;

Definition 5. *A function* $R : \mathcal{K} \to \mathbb{R}$ *is* s*-strongly convex with respect to the norm* $\|\cdot\|$ *if it satisfies the following condition; for any* $\mathrm{X}, \mathrm{Y} \in \mathcal{K}$*,*

$$R(\mathrm{X}) \geq R(\mathrm{Y}) + \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) + \frac{s}{2}\|\mathrm{X} - \mathrm{Y}\|^2,$$

A sufficient condition of the strong convexity is, for any $\mathrm{X} \in \mathcal{K}$ and $\mathrm{W} \in \mathbb{S}^{N\times N}$,

$$\mathrm{vec}(\mathrm{W})^{\mathsf{T}}\nabla^2 R(\mathrm{X})\mathrm{vec}(\mathrm{W}) \geq s\|\mathrm{W}\|^2.$$

The next lemma is a standard derivation method of regret bounds.

Lemma 1 (See, e.g., Theorem 2.11 of [68])**.** *Assume that for some real numbers* $s, g > 0$ *and a norm* $\|\cdot\|$ *the following holds.*

1. *R is s-strongly convex with respect to the norm $\|\cdot\|$.*

2. *Any loss matrix $\mathrm{L} \in \mathcal{L}$ satisfies $\|\mathrm{L}\|_* \leq g$, where $\|\cdot\|_*$ is the dual norm of $\|\cdot\|$.*

Then, the FTRL with regularizer R and $\mathrm{X}_1 = \arg\min_{\mathrm{X}\in\mathcal{K}} R(\mathrm{X})$ achieves

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) \leq \frac{H_0}{\eta} + \eta\frac{g^2}{s}T.$$

where $H_0 = \max_{\mathrm{X},\mathrm{Y}\in\mathcal{K}}(R(\mathrm{X}) - R(\mathrm{Y}))$ Additionally, with the learning rate $\eta = \sqrt{H_0 s/(g^2 T)}$, we have $Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) \leq 2g\sqrt{TH_0/s}$.

The proof of this lemma consists of some series of other lemmas. First we show an intermediate bound of the regret known as the FTL-BTL (Follow-The-Leader-Be-The-Leader) Lemma.

Lemma 2 (Lemma 2.3 of [68])**.** *Let $\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_T \in \mathcal{K}$ be matrices generated by a FTRL-based algorithm with the regularizer R. Then for any $\mathrm{X}^\star \in \mathcal{K}$,*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star) \leq \frac{R(\mathrm{X}^\star) - R(\mathrm{X}_1)}{\eta} + \sum_{t=1}^{T} \mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1}). \tag{3.5}$$

Proof. For simplicity of the notation, let we write $f_t(\mathrm{X}) = \eta\mathrm{L}_t \bullet \mathrm{X}$ for $t = 1, \ldots, T$ and $f_0(\mathrm{X}) = R(\mathrm{X})$. Rearranging above inequality and subtracting $R(\mathrm{X}_1) + \eta\sum_{t=1}^{T} \mathrm{L}_t \bullet \mathrm{X}_t$ from both sides, we have

$$\sum_{t=0}^{T} f_t(\mathrm{X}_{t+1}) \leq \sum_{t=0}^{T} f_t(\mathrm{X}^\star).$$

Actually we prove this inequality by induction. For the case $T = 0$, we have $f_0(\mathrm{X}_1) = R(\mathrm{X}_1) \leq R(\mathrm{X}^\star) = f_0(\mathrm{X}^\star)$ by the definition of X_1. For the case $T \geq 1$, we assume that the inequality $\sum_{t=0}^{T-1} f_t(\mathrm{X}_{t+1}) \leq \sum_{t=0}^{T-1} f_t(\mathrm{X}^\star)$ holds. Adding $f_T(\mathrm{X}_{T+1})$ to both sides, we have

$$\sum_{t=0}^{T} f_t(\mathrm{X}_{t+1}) \leq f_T(\mathrm{X}_{T+1}) + \sum_{t=0}^{T-1} f_t(\mathrm{X}^\star).$$

This inequality holds for any $\mathrm{X}^\star$, so we set $\mathrm{X}^\star = \mathrm{X}_{T+1}$ then we get

$$\sum_{t=0}^{T} f_t(\mathrm{X}_{t+1}) \leq \sum_{t=0}^{T} f_t(\mathrm{X}^\star) = \min_{\mathrm{X}\in\mathcal{K}} \sum_{t=0}^{T} f_t(\mathrm{X}).$$

Thus we get the intermediate bound. □

Next we bound $\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1})$ in terms of s, the parameter of the strong convexity. To bound this term, we need the following technical lemma.

Lemma 3 (Lemma 2.8 on [68])**.** *Let $\mathcal{K}$ be a nonempty convex set. Let R be a differentiable and s-strongly convex function w.r.t. the norm $\|\cdot\|$ over $\mathcal{K}$. Let $\mathrm{X}^\star = \arg\min_{\mathrm{X}\in\mathcal{K}} R(\mathrm{X})$. Then for all $\mathrm{X} \in \mathcal{K}$,*

$$R(\mathrm{X}) - R(\mathrm{X}^\star) \geq \frac{s}{2}\|\mathrm{X}^\star - \mathrm{X}\|^2.$$

Proof. In the $\mathrm{X}^\star$ is interior of $\mathcal{K}$, then $\nabla R(\mathrm{X}^\star) = 0$ and by the definition of the strong convexity leads the lemma. In the boundary case, one can say $\forall \mathrm{X}^\star \in \mathcal{K}, \nabla R(\mathrm{X}) \bullet (\mathrm{X}^\star - \mathrm{X}) \geq 0$. □

Now we are ready to give a proof of 1.

Proof of Lemma 1. Let $F_t(\mathrm{X}) = R(\mathrm{X}) + \eta \sum_{s=1}^{t} \mathrm{L}_s \bullet \mathrm{X}$. The FTRL update rule can be written as $\mathrm{X}_{t+1} = \arg\min_{\mathrm{X}\in\mathcal{K}} F_t(\mathrm{X})$. Adding an affine function does not change the convexity thus F_t is also s-strongly convex w.r.t. $\|\cdot\|$. Using Lemma 3 to F_t and F_{t+1}, we get

$$\begin{aligned} F_t(\mathrm{X}_t) &\geq F_t(\mathrm{X}_{t+1}) + \frac{s}{2}\|\mathrm{X}_t - \mathrm{X}_{t+1}\|^2, \\ F_{t-1}(\mathrm{X}_{t+1}) &\geq F_{t-1}(\mathrm{X}_t) + \frac{s}{2}\|\mathrm{X}_{t+1} - \mathrm{X}_t\|^2. \end{aligned}$$

Summing these inequalities and rearranging, we have

$$s\|\mathrm{X}_t - \mathrm{X}_{t+1}\|^2 \leq \mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1}) \leq g\|\mathrm{X}_t - \mathrm{X}_{t+1}\|,$$

where the last inequality we use $\|\mathrm{L}_t\|_* \leq g$ and Cauchy-Schwartz inequality. Thus we have $\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1}) \leq g^2/s$ and complete the proof. □

In the subsequent subsections, we give regret bounds for the FTRL with popular regularizers. The first two are straightforward to derive from known results.

3.3.1 FTRL with the Frobenius norm regularization

The Frobenius norm regularization function is defined as $R(\mathrm{X}) = \frac{1}{2}\|\mathrm{X}\|_{\mathrm{Fr}}^2$, which is the matrix analogue of the L_2-norm for vectors. The FTRL with this regularizer yields the online gradient descent (OGD) algorithm [37]. Since R is 1-strongly convex with respect to $\|\cdot\|_{\mathrm{Fr}}$ and the dual of $\|\cdot\|_{\mathrm{Fr}}$ is $\|\cdot\|_{\mathrm{Fr}}$, Lemma 1 gives

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}_2, \mathcal{L}_2) \leq \rho\gamma_2\sqrt{2T}, \tag{3.6}$$

where $\mathcal{K}_2 = \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : \|\mathrm{X}\|_{\mathrm{Fr}} \leq \rho\}$ and $\mathcal{L}_2 = \{\mathrm{L} \in \mathbb{S}^{N\times N} : \|\mathrm{L}\|_{\mathrm{Fr}} \leq \gamma_2\}$.

3.3.2 FTRL with the entropic regularization

The entropic regularization function is defined as $R(\mathrm{X}) = \mathrm{Tr}(\mathrm{X}\log\mathrm{X} - \mathrm{X})$, which is the matrix analogue of the unnormalized entropy for vectors. Slightly modifying the proof in [38], we obtain the following regret bound for the FTRL with this regularizer.

Proposition 4. *For an online SDP problem $(\mathcal{K}_1, \mathcal{L}_\infty)$ such that $\mathcal{K}_1 = \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : \|\mathrm{X}\|_{\mathrm{Tr}} \leq \tau\}$ and $\mathcal{L}_\infty = \{\mathrm{L} \in \mathbb{S}^{N\times N} : \|\mathrm{L}\|_{\mathrm{Sp}} \leq \gamma_\infty\}$, the FTRL with the entropic regularizer $R(\mathrm{X}) = \mathrm{Tr}(\mathrm{X}\log\mathrm{X} - \mathrm{X})$ achieves*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}_1, \mathcal{L}_\infty) \leq 2\tau\gamma_\infty\sqrt{T\log N}. \tag{3.7}$$

Proof. We begin with the intermediate bound shown in Theorem 18 of [38];

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}_1, \mathcal{L}_\infty) \leq \eta\sum_{t=1}^{T}\mathrm{X}_t \bullet \mathrm{L}_t^2 + \frac{\tau\log N}{\eta}.$$

Thus our modification of the proof is bounding $\mathrm{X}_t \bullet \mathrm{L}_t^2$ using our constraints. Using Cauchy-Schwartz inequality and the fact that the spectral norm of a matrix is its maximal eigenvalue, we have

$$\mathrm{X}_t \bullet \mathrm{L}_t^2 \leq \|\mathrm{X}_t\|_{\mathrm{Tr}}\|\mathrm{L}_t^2\|_{\mathrm{Sp}} = \|\mathrm{X}_t\|_{\mathrm{Tr}}\lambda_1(\mathrm{L}_t^2) = \|\mathrm{X}_t\|_{\mathrm{Tr}}\lambda_1(\mathrm{L}_t)^2 \leq \tau\gamma_\infty^2,$$

where $\lambda_1(\mathrm{L})$ is the maximal eigenvalue of L. □

Note that, originally the algorithm proposed in [38] is the agile version of the OMD,

$$\mathrm{X}_{t+1} = \arg\min_{\mathrm{X}\in\mathcal{K}_1} B_R(\mathrm{X}, e^{\ln\mathrm{X}_t - \eta\mathrm{L}_t}).$$

where $B_R(\mathrm{X}, \mathrm{Y}) = \mathrm{Tr}(\mathrm{X}\ln\mathrm{X} - \mathrm{X}\ln\mathrm{Y} - \mathrm{X} + \mathrm{Y})$, the Bregman divergence with respect to R. But we can see that the regret bound of the FTRL with the regularizer R is same as the agile OMD and Theorem 18 of [38] still holds in this case. We have $\nabla R(\mathrm{X}) = \ln\mathrm{X}$, $(\nabla R)^{-1}(\mathrm{X}) = e^{\mathrm{X}}$ and $\tilde{\mathrm{X}}_{t+1} = e^{\ln(\tilde{\mathrm{X}}_t) - \eta\mathrm{L}_t}$. Additionally, we can solve the projection step onto $\mathcal{K}_1$ as :

$$\arg\min_{\mathrm{X}\in\mathcal{K}_1} B_R(\mathrm{X}, \mathrm{Y}) = \begin{cases} \mathrm{Y}, & \mathrm{Y} \in \mathcal{K}_1. \\ \tau\mathrm{Y}/\mathrm{Tr}(\mathrm{Y}), & \mathrm{Y} \notin \mathcal{K}_1 \end{cases}$$

by using the Lagrange multiplier and KKT conditions. This implies that $\tilde{\mathrm{X}}_{t+1} = e^{\ln(\mathrm{X}_t) - \eta \mathrm{L}_t}$ so the agile OMD is equivalent to the lazy OMD.

3.3.3 FTRL with the log-determinant regularization

The log-determinant regularization function is defined as $R(\mathrm{X}) = -\ln\det(\mathrm{X} + \epsilon\mathrm{E})$ where ϵ is a positive constant. This is the matrix analogue of the Burg entropy $-\sum_{i=1}^{N} \ln x_i$ for vectors $\boldsymbol{x}$ whose Bregman divergence is the Itakura-Saito divergence. The constant ϵ stabilizes the regularizer to make the regret bound finite. Unfortunately, it is unclear what norm is appropriate for measuring the strong convexity of the log-determinant regularizer to obtain a tight regret bound. In the next theorem, we examine the spectral norm and give a (probably loose) regret bound for the online SDP problem $(\mathcal{K}_\infty, \mathcal{L}_1)$, where $\mathcal{K}_\infty = \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : \|\mathrm{X}\|_{\mathrm{Sp}} \leq \sigma\}$ and $\mathcal{L}_1 = \{\mathrm{L} \in \mathbb{S}^{N\times N} : \|\mathrm{L}\|_{\mathrm{Tr}} \leq \gamma_1\}$.

Theorem 5. *The FTRL with the log-determinant regularizer with $\epsilon = \sigma$ achieves*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}_\infty, \mathcal{L}_1) \leq 4\sigma\gamma_1\sqrt{TN\ln 2}. \tag{3.8}$$

To show this theorem, we need the following technical lemma. This lemma is a composition of the chain rule and standard derivation of the log-determinant appeared in many papers (e.g. [67]).

Lemma 4 ([30])**.** *The Hessian of $R(\mathrm{X}) = -\ln\det(\mathrm{X} + \epsilon\mathrm{E})$ for $\mathrm{X} \in \mathbb{S}_+^{N\times N}$ is $\nabla^2 R(\mathrm{X}) = (\mathrm{X} + \epsilon\mathrm{E})^{-1} \boxtimes (\mathrm{X} + \epsilon\mathrm{E})^{-1}$ where $\boxtimes$ denotes the box product.*

Proof. By using Equation (R21) of [30], we have the first-order derivative,

$$\begin{aligned}
\mathrm{vec}(\nabla \ln\det(\mathrm{X} + \epsilon\mathrm{E})^{\mathsf{T}}) &= (\nabla(\mathrm{X} + \epsilon\mathrm{E}))^{\mathsf{T}}\mathrm{vec}((\mathrm{X} + \epsilon\mathrm{E})^{-1}) \\
&= \mathrm{E}^{\mathsf{T}}\mathrm{vec}((\mathrm{X} + \epsilon\mathrm{E})^{-1}) = \mathrm{vec}((\mathrm{X} + \epsilon\mathrm{E})^{-1}).
\end{aligned}$$

Thus we get $\nabla \ln\det(\mathrm{X} + \epsilon\mathrm{E}) = (\mathrm{X} + \epsilon\mathrm{E})^{-T}$.

Regarding $\mathrm{F}(\mathrm{X}) = \mathrm{X}^{-1}$ and $\mathrm{G}(\mathrm{X}) = (\mathrm{X} + \epsilon\mathrm{E})$ and using Equation (R18) with (R12) of [30], we get

$$\begin{aligned}
\nabla(\mathrm{X} + \epsilon\mathrm{E})^{-T} &= \nabla\mathrm{F}(\mathrm{G}(\mathrm{X})) = (\nabla_{\mathrm{Y}}\mathrm{F}(\mathrm{Y})|_{\mathrm{Y}=\mathrm{G}(\mathrm{X})})\nabla\mathrm{G}(\mathrm{X}) \\
&= -\mathrm{Y}^{-\mathsf{T}} \boxtimes \mathrm{Y}^{-1}|_{\mathrm{Y}=\mathrm{X}+\epsilon\mathrm{E}} \\
&= -(\mathrm{X} + \epsilon\mathrm{E})^{-\mathsf{T}} \boxtimes (\mathrm{X} + \epsilon\mathrm{E})^{-1} = -(\mathrm{X} + \epsilon\mathrm{E})^{-1} \otimes (\mathrm{X} + \epsilon\mathrm{E})^{-1}.
\end{aligned}$$

Note that we use $\mathrm{X} \in \mathbb{S}_+^{N \times N}$ in the last equation. Thus we get the first statement. □

Proof of Theorem 5. Below we show that R is $(1/(4\sigma^2))$-strongly convex with respect to $\|\cdot\|_{\mathrm{Sp}}$ and $R(\mathrm{X}) - R(\mathrm{X}') \leq N \ln 2$ for any $\mathrm{X}, \mathrm{X}' \in \mathcal{K}$. Since $\|\cdot\|_{\mathrm{Tr}}$ is the dual norm of $\|\cdot\|_{\mathrm{Sp}}$ and it is clear that $\|L\|_{\mathrm{Tr}} \leq \gamma_1$ for any $\mathrm{L} \in \mathcal{L}_1$, the theorem follows from Lemma 1.

The strong convexity of the log-determinant can be verified by showing positive definiteness of the Hessian of R. By Lemma 4, the Hessian of $R(\mathrm{X}) = -\ln\det(\mathrm{X} + \epsilon\mathrm{E})$ is $\nabla^2 R(\mathrm{X}) = (\mathrm{X} + \epsilon\mathrm{E})^{-1} \boxtimes (\mathrm{X} + \epsilon\mathrm{E})^{-1}$ [30]. Now we will convert the box product to the Kronecker product. By using (2.1), (2.2) and the symmetricity of matrices we have

$$\begin{aligned}\mathrm{vec}(\mathrm{W})^\mathsf{T}(\mathrm{X} + \epsilon\mathrm{E})^{-1} \boxtimes (\mathrm{X} + \epsilon\mathrm{E})^{-1}\mathrm{vec}(\mathrm{W}) &= \mathrm{vec}(\mathrm{W})^\mathsf{T}\mathrm{vec}((\mathrm{X} + \epsilon\mathrm{E})^{-1}\mathrm{W}(\mathrm{X} + \epsilon\mathrm{E})^{-1}) \\ &= \mathrm{vec}(\mathrm{W})^\mathsf{T}((\mathrm{X} + \epsilon\mathrm{E})^{-1} \otimes (\mathrm{X} + \epsilon\mathrm{E})^{-1})\mathrm{vec}(\mathrm{W})\end{aligned}$$

Since an eigenvalue of $\mathrm{A} \otimes \mathrm{B}$ is the product of some eigenvalues of A and B (see, e.g., [64]) and an eigenvalue of A^{-1} is the reciprocal of an eigenvalue of A, the minimum eigenvalue of $(\mathrm{X} + \epsilon\mathrm{E})^{-1} \otimes (\mathrm{X} + \epsilon\mathrm{E})^{-1}$ is $(\|\mathrm{X}\|_{\mathrm{Sp}} + \epsilon)^{-2}$. This implies that $\min_{\mathrm{W} \in \mathbb{S}^{N \times N}} \mathrm{vec}(\mathrm{W})^\mathsf{T} \nabla^2 R(\mathrm{X}) \mathrm{vec}(\mathrm{W})^\mathsf{T} \geq (\sigma + \epsilon)^{-2} \|\mathrm{W}\|_{\mathrm{Fr}}^2$. In other words, for any $\mathrm{W} \in \mathbb{S}^{N \times N}$,

$$\mathrm{vec}(\mathrm{W})^\mathsf{T}(\nabla^2 R(\mathrm{X}) - (\sigma + \epsilon)^{-2}\mathrm{E})\mathrm{vec}(\mathrm{W}) \geq 0.$$

Rearranging this inequality and using the fact that $\mathrm{vec}(\mathrm{W})^\mathsf{T}\mathrm{vec}(\mathrm{W}) = \|\mathrm{W}\|_{\mathrm{Fr}}^2 \geq \|\mathrm{W}\|_{\mathrm{Sp}}^2$, we get $\mathrm{vec}(\mathrm{W})^\mathsf{T} \nabla^2 R(\mathrm{X}) \mathrm{vec}(\mathrm{W}) \geq (\sigma + \epsilon)^{-2} \|\mathrm{W}\|_{\mathrm{Sp}}^2$. This implies that R is $(1/(4\sigma^2))$-strongly convex with respect to $\|\cdot\|_{\mathrm{Sp}}$.

Next we give upper and lower bounds of R. Note that $\det(\mathrm{X} + \epsilon\mathrm{E})$ is the product of all eigenvalues of $\mathrm{X} + \epsilon\mathrm{E}$. Since, all the eigenvalues are positive and the maximum of them is bounded by $\sigma + \epsilon$, we have $\epsilon^N \leq \det(\mathrm{X} + \epsilon\mathrm{E}) \leq (\sigma + \epsilon)^N = (2\epsilon)^N$. So, $H_0 = \max_{\mathrm{X}, \mathrm{Y} \in \mathcal{K}}(R(\mathrm{X}) - R(\mathrm{Y})) \leq N \ln 2$. □

Note that this result is not very impressive, because $\mathcal{K}_\infty \subseteq \mathcal{K}_2$ with $\rho = \sqrt{N}\sigma$ and $\mathcal{L}_1 \subseteq \mathcal{L}_2$ with $\gamma_2 = \gamma_1$, and hence the FTRL with the Frobenius norm regularizer has a slightly better regret bound for $(\mathcal{K}_\infty, \mathcal{L}_1)$.

In the following sections, we consider a special class of online SDP problems $(\mathcal{K}, \mathcal{L})$ where $\mathcal{K}$ and $\mathcal{L}$ are further restricted by some complicated way. For such problems, it is unlikely to derive tight regret bounds from Lemma 1.

3.4 Online matrix prediction and reduction to online SDP

Before going to our main contribution, we give a more natural setting to describe various applications, which is called the online matrix prediction (OMP) problem. Then we briefly review the result of Hazan et al., saying that OMP problems are reduced to online SDP problems $(\mathcal{K}, \mathcal{L})$ of special form [38]. In particular, the loss matrices in $\mathcal{L}$ obtained by the reduction are sparse. This result motivates us to improve regret bounds for online sparse SDP problems.

An OMP problem is specified by a pair $(\mathcal{W}, G)$, where $\mathcal{W} \subseteq [-1,1]^{m \times n}$ is a convex set of matrices of size $m \times n$ and $G > 0$ is a positive real number. Note that we do not require $m = n$ or $W^\mathsf{T} = W$. The OMP problem $(\mathcal{W}, G)$ is described as the following protocol: In each round $t = 1, 2, \ldots, T$, the algorithm

1. receives a pair $(i_t, j_t) \in [m] \times [n]$ from the adversary,
2. chooses $\mathrm{W}_t \in \mathcal{W}$ and output $W_{t,(i_t,j_t)}$,
3. receives G-Lipschitz convex loss function $\ell_t : [-1,1] \to \mathbb{R}$ from the adversary, and
4. suffers the loss $\ell_t(W_{t,(i_t,j_t)})$.

The goal is to minimize the following regret:

$$Reg_{\mathrm{OMP}}(T, \mathcal{W}) = \sum_{t=1}^{T} \ell_t(W_{t,(i_t,j_t)}) - \min_{\mathrm{U} \in \mathcal{W}} \sum_{t=1}^{T} \ell_t(U_{i_t,j_t}).$$

The online max-cut, the online gambling and the online collaborative filtering problems are instances of the OMP problems.

Online max-cut: On each round, the algorithm receives a pair of nodes $(i, j) \in [n] \times [n]$ and should decide whether there is an edge between the nodes. Formally, the algorithm chooses $\hat{y}_t \in [-1, 1]$, which is interpreted as a randomized prediction in $\{-1, 1\}$: predicts 1 with probability $(1 + \hat{y}_t)/2$ and -1 with the remaining probability. The adversary then gives the true outcome $y_t \in \{-1, 1\}$ indicating whether (i_t, j_t) is actually joined by an edge or not. The loss suffered by the algorithm is $\ell_t(\hat{y}_t) = |\hat{y}_t - y_t|/2$, which is interpreted as the probability that the prediction is incorrect. Note that ℓ_t is $(1/2)$-Lipschitz. The decision space $\mathcal{W}$ of this problem is the convex hull of the set $\mathcal{C}$ of matrices that represent cuts, that is, $\mathcal{C} = \{\mathrm{C}^A \in \{-1,1\}^{n \times n} : A \subseteq [n]\}$, where $C^A_{i,j} = 1$ if $((i \in A)$ and $(j \notin A))$ or $((i \notin A)$ and $(j \in A))$, and

$C^A_{i,j} = -1$ otherwise. Note that the best offline matrix $\mathrm{C}^A = \arg\min_{\mathrm{C}^A \in \mathcal{C}} \sum_t \ell_t(U_{i_t,j_t})$ in $\mathcal{C}$ is the matrix corresponding to the max-cut A in the weighted graph whose edge weight are given by $w_{ij} = \sum_{t:(i_t,j_t)=(i,j)} y_t$ for every (i,j) [38]. This is the reason why the problem is called online max-cut.

Online gambling: On each round, the algorithm receives a pair of teams $(i,j) \in [n] \times [n]$, and should decide whether i is going to beat j or not in the upcoming game. The decision space is the convex hull of all permutation matrices $\mathrm{W}^P \in \{0,1\}^{n \times n}$, where W^P is the matrix corresponding to permutation P over $[n]$ that satisfies $W^P_{i,j} = 1$ if i appears before j in the permutation P and $W^P_{i,j} = 0$ otherwise.

Online collaborative filtering: We described this problem in Introduction. We consider $\mathcal{W} = \{\mathrm{W} \in [-1,1]^{m \times n} : \|\mathrm{W}\|_{\mathrm{Tr}} \leq \tau\}$ for some constant $\tau > 0$, which is a typical choice for the decision space in the literature.

The next proposition shows how the OMP problem $(\mathcal{W}, G)$ is reduced to the online SDP problem $(\mathcal{K}, \mathcal{L})$. We need to define the notion of (β,τ)-decomposability of $\mathcal{W}$ before stating the proposition.

For a matrix $\mathrm{W} \in \mathcal{W}$, let $\mathrm{sym}(\mathrm{W}) = \begin{bmatrix} 0 & \mathrm{W} \\ \mathrm{W}^\mathsf{T} & 0 \end{bmatrix}$ if $\mathcal{W}$ is not symmetric (some $\mathrm{W} \in \mathcal{W}$ is not symmetric) and $\mathrm{sym}(\mathrm{W}) = \mathrm{W}$ otherwise. Let $q = m$ if $\mathcal{W}$ is not symmetric and $q = 0$ otherwise. Let p be the order of $\mathrm{sym}(\mathrm{W})$, that is $p = q + n$. Note that any symmetric matrix can be represented by the difference of two symmetric and positive semi-definite matrices. For real numbers $\beta > 0$ and $\tau > 0$, the decision space $\mathcal{W}$ is (β,τ)-decomposable if for any $\mathrm{W} \in \mathcal{W}$, there exists $\mathrm{P}, \mathrm{Q} \in \mathbb{S}^{p \times p}_+$ such that $\mathrm{sym}(\mathrm{W}) = \mathrm{P} - \mathrm{Q}$, $\mathrm{Tr}(\mathrm{P} + \mathrm{Q}) \leq \tau$ and $P_{i,i} \leq \beta$, $Q_{i,i} \leq \beta$ for every $i \in [p]$. Note that if $\mathcal{W}$ is (β,τ)-decomposable then its convex hull is also (β,τ)-decomposable due to the linearity of the trace and the summation of matrices. According to [38], the best offline solution of of the OMP U is chosen from an appropriate set of matrices, not a convex hull $\mathcal{W}$. But the decomposability of $\mathcal{W}$ also ensures that the regret bound still holds even if we choose U from a convex hull $\mathcal{W}$ as in our problem formulation.

Proposition 6 (Hazan et al. [38])**.** *Let $(\mathcal{W}, G)$ be the OMP problem where $\mathcal{W} \subseteq [-1,1]^{m \times n}$ is (β,τ)-decomposable. Then, the OMP problem $(\mathcal{W}, G)$ can be reduced to the online SDP*

problem $(\mathcal{K}, \mathcal{L})$*, where* $N = 2p$ *and*

$$\begin{aligned}\mathcal{K} &= \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : \|\mathrm{X}\|_{\mathrm{Tr}} \le \tau, \forall i \in [N], X_{i,i} \le \beta, \\ &\qquad \forall (i,j) \in [m] \times [n], X_{i,j+q} - X_{p+i,p+j+q} \in [-1,1]\}, \\ \mathcal{L} &= \{\mathrm{L} \in \mathbb{S}^{N\times N} : \forall (i,j) \in [N] \times [N], L_{i,j} \le G, |\{(i,j) : L_{i,j} \neq 0\}| \le 4, L^2 \textit{ is diagonal}\}.\end{aligned}$$

Moreover, the regret of the OMP problem is bounded by that of the reduced online SDP problem

$$Reg_{\mathrm{OMP}}(T, \mathcal{W}) \le \frac{1}{2} Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}).$$

We omit the condition

$$\forall (i,j) \in [m] \times [n] X_{i,j+q} - X_{p+i,p+j+q} \in [-1,1]$$

for $q = m$ if $\mathcal{W}$ is not symmetric and $q = 0$ otherwise are from original definition of $\mathcal{K}$ in [38]. This is because these conditions does not affect the derivation of the regret bound we give in this thesis.

Note that the loss space $\mathcal{L}$ obtained by the reduction is very sparse: each loss matrix has only 4 non-zero entries. Thus, we can say that for every $\mathrm{L} \in \mathcal{L}$, $\|\mathrm{L}\|_{\mathrm{Fr}} \le 2G$ and $\|\mathrm{vec}(\mathrm{L})\|_1 \le 4G$.

Hazan et al. also give a regret bound of the FTRL with the entropic regularizer when applied to the online SDP problem $(\mathcal{K}, \mathcal{L})$ for $\mathcal{K}$ obtained by the reduction above with a larger loss space $\mathcal{L}$ (thus applicable to the online OMP problems).

Theorem 7 (Hazan et al. [38])**.** *For the online SDP problem* $(\mathcal{K}, \mathcal{L})$ *where*

$$\begin{aligned}\mathcal{K} &= \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : \|\mathrm{X}\|_{\mathrm{Tr}} \le \tau, \forall i \in [N], X_{i,i} \le \beta\}, \\ \mathcal{L} &= \{\mathrm{L} \in \mathbb{S}^{N\times N} : \mathrm{Tr}(\mathrm{L}^2) \le \gamma, \mathrm{L}^2 \textit{ is diagonal}\},\end{aligned}$$

the agile OMD with the Bregman divergence with respect to the negaitve entropy $R(\mathrm{X}) = \mathrm{Tr}(\mathrm{X} \ln \mathrm{X} - \mathrm{X})$ *achieves*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) \le 2\sqrt{\beta\tau\gamma T \ln N}.$$

Combining Proposition 6 and Theorem 7, we can easily get regret bounds for OMP problems.

Corollary 8. *For the OMP problem $(\mathcal{W}, G)$, where $\mathcal{W} \subseteq [-1,1]^{m\times n}$ is (β,τ)-decomposable, there exists an algorithm that achieves*

$$Reg_{\mathrm{OMP}}(T,\mathcal{K},\mathcal{L}) = O(G\sqrt{\beta\tau T\ln(m+n)}).$$

Hazan et al. apply the bound to the three applications, for which the decision classes $\mathcal{W}$ are all (β,τ)-decomposable for some β and τ [38]. More specifically, the results are summarized as shown below.

Online max-cut: The problem is $(1,n)$-decomposable and thus the algorithm has a regret bound of $O(G\sqrt{nT\ln n})$.

Online gambling: The problem is $(O(\ln n), O(n\ln n))$-decomposable and thus the algorithm has a regret bound of $O(G\sqrt{nT(\ln n)^3})$.

Online collaborative filtering: The problem is $(\sqrt{m+n}, 2\tau)$-decomposable and thus the algorithm has a regret bound of $O(G\sqrt{\tau T\sqrt{m+n}\ln(m+n)})$, which is $O(G\sqrt{\tau T\sqrt{n}\ln n})$ if we assume without loss of generality that $n \geq m$.

Christiano provides another technique of reduction from a special type of OMP problems to a special type of online SDP problems, and apply the FTRL with the log-determinant regularizer [21]. He then improves the regret bound for the online max-cut problem to $O(G\sqrt{nT})$, which matches a lower bound up to a constant factor. However, the regret bound for online gambling is much worse ($O(Gn^2\sqrt{T})$) and his reduction cannot be applied to online collaborative filtering. It is worth noted that the loss matrices obtained by his reduction are not just sparse but *block-wise sparse*, by which we mean non-zero entries forming at most two block matrices, and seemingly his regret analysis depends on this fact.

3.5 Main results

Motivated by the sparse online SDP problem reduced from an OMP problem, we consider a specific problem $(\tilde{\mathcal{K}}, \tilde{\mathcal{L}})$, where

$$\begin{aligned}\tilde{\mathcal{K}} &= \{\mathrm{X}\in\mathbb{S}_+^{N\times N} : \|\mathrm{X}\|_{\mathrm{Tr}} \leq \tau, \forall i\in[N], \mathrm{X}_{i,i}\leq\beta\},\\ \tilde{\mathcal{L}} &= \{\mathrm{L}\in\mathbb{S}^{N\times N} : \|\mathrm{vec}(\mathrm{L})\|_1 \leq g_1\},\end{aligned}$$

and give a regret bound of the FTRL with the log-determinant regularizer. Note that $\tilde{\mathcal{K}}$ is the same as the one obtained by the reduction and $\tilde{\mathcal{L}}$ is much larger if $g_1 = 4G$. By Proposition 6 the regret bound immediately yields a regret bound for the OMP problem $(\mathcal{W}, G)$ for a (β, τ)-decomposable decision space $\mathcal{W}$, which turns out to be tighter than the one using the entropic regularizer shown in Theorem 7.

Our analysis partly follows that of [21] with some generalizations. In particular, we figure out a general method for deriving regret bounds by using a new notion of strong convexity of regularizers, which is implicitly used in [21]. First we state the general theory.

3.5.1 A general theory

We begin with Lemma 2, FTL-BTL lemma. Thanks to this lemma, all we need to do is to bound $H_0 = \max_{\mathrm{X},\mathrm{Y}\in\mathcal{K}}(R(\mathrm{X}) - R(\mathrm{Y}))$ and $\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1})$.

Now we define the new notion of strong convexity. Intuitively, this is an integration of the strong convexity of regularizers with respect to a norm and the Lipschitzness of loss functions with respect to the norm.

Definition 6. *For a decision space $\mathcal{K}$ and a real number $s > 0$, a regularizer $R : \mathcal{K} \to \mathbb{R}$ is said to be s-strongly convex with respect to a loss space $\mathcal{L}$ if for any $\alpha \in [0, 1]$, any $\mathrm{X}, \mathrm{Y} \in \mathcal{K}$, and any $\mathrm{L} \in \mathcal{L}$,*

$$R(\alpha\mathrm{X} + (1-\alpha)\mathrm{Y}) \le \alpha R(\mathrm{X}) + (1-\alpha)R(\mathrm{Y}) - \frac{s}{2}\alpha(1-\alpha)|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})|^2. \tag{3.9}$$

This condition has an equivalent form as same as the strong convexity with respect to the norm.

Proposition 9 (Cf. [62])**.** *The following condition is equivalent to* (3.9) *: For any $\mathrm{X}, \mathrm{Y} \in \mathcal{K}$ and $\mathrm{L} \in \mathcal{L}$,*

$$R(\mathrm{X}) \ge R(\mathrm{Y}) + \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) + \frac{s}{2}|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})|^2. \tag{3.10}$$

Proof. We prove this by showing that the convexity of $\tilde{R}(\mathrm{X}) = R(\mathrm{X}) - s(\mathrm{L} \bullet \mathrm{X})^2/2$ and using the first-order condition of the convexity. Note that the derivative of $\tilde{R}$ is $\nabla\tilde{R}(\mathrm{X}) = \nabla R(\mathrm{X}) + s(\mathrm{L} \bullet \mathrm{X})\mathrm{L}$.

First we begin with (3.10) then derive the first order condition of $\tilde{R}$. By (3.10), we have

$$R(\mathrm{X}) \ge R(\mathrm{Y}) + \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) + \frac{s}{2}((\mathrm{L} \bullet \mathrm{X})^2 - 2(\mathrm{L} \bullet \mathrm{X})(\mathrm{L} \bullet \mathrm{Y}) + (\mathrm{L} \bullet \mathrm{Y})^2).$$

Thus, we get

$$\begin{aligned}\tilde{R}(\mathrm{X}) &\geq \tilde{R}(\mathrm{Y}) + \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) + s((\mathrm{L} \bullet \mathrm{Y})^2 - (\mathrm{L} \bullet \mathrm{X})(\mathrm{L} \bullet \mathrm{Y})) \\ &= \tilde{R}(\mathrm{Y}) + (\nabla R(\mathrm{Y}) - s(\mathrm{L} \bullet \mathrm{X})\mathrm{L}) \bullet (\mathrm{X} - \mathrm{Y}),\end{aligned}$$

the first order condition of the convexity of $\tilde{R}$.

Next we start with the convexity of $\tilde{R}$ and derive (3.9). Let $\mathrm{Z} = \alpha\mathrm{X} + (1-\alpha)\mathrm{Y}$ for any $\alpha \in [0,1]$. Since $\tilde{R}$ is convex we have

$$\tilde{R}(\mathrm{Z}) \leq \alpha\tilde{R}(\mathrm{Y}) + (1-\alpha)\tilde{R}(\mathrm{Y}).$$

Thus we get,

$$R(\mathrm{Z}) \leq \alpha R(\mathrm{X}) + (1-\alpha)R(\mathrm{Y}) - \frac{s}{2}\left(\alpha(\mathrm{L} \bullet \mathrm{X})^2 + (1-\alpha)(\mathrm{L} \bullet \mathrm{Y})^2 - (\mathrm{L} \bullet \mathrm{Z})^2\right),$$

and $\mathrm{Z} = \alpha\mathrm{X} + (1-\alpha)\mathrm{Y}$ leads

$$\begin{aligned}&\alpha(\mathrm{L} \bullet \mathrm{X})^2 + (1-\alpha)(\mathrm{L} \bullet \mathrm{Y})^2 - (\mathrm{L} \bullet \mathrm{Z})^2 \\ &= \alpha(1-\alpha)(\mathrm{L} \bullet \mathrm{X})^2 + \alpha(1-\alpha)(\mathrm{L} \bullet \mathrm{Y})^2 - 2\alpha(1-\alpha)(\mathrm{L} \bullet \mathrm{X})(\mathrm{L} \bullet \mathrm{Y}) \\ &= \alpha(1-\alpha)(\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y}))^2.\end{aligned}$$

This completes the proof. □

Note that the condition (3.10) has the same form as the condition of s-strong convexity given in Lemma 1 except $\|\mathrm{X} - \mathrm{Y}\|$ is replaced by $|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})|$.

The following lemma gives a bound of the term $\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1})$ in inequality (3.5) in terms of the strong convexity of the regularizer. The lemma is implicitly stated in [68] and hence is not essentially new. But we give a proof for completeness since it is not very straightforward to show.

Lemma 5 (Main lemma of this chapter)**.** *Let $R : \mathcal{K} \to \mathbb{R}$ be s-strongly convex with respect to $\mathcal{L}$ for $\mathcal{K}$. Then, the FTRL with the regularizer R and learing rate $\eta > 0$ applied to $(\mathcal{K}, \mathcal{L})$ achieves*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) \leq \frac{H_0}{\eta} + \frac{\eta T}{s}.$$

Proof. By Lemma 2, it suffices to show that

$$\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1}) \leq \frac{\eta}{s},$$

since the theorem follows by setting $\eta = \sqrt{sH_0/T}$. In what follows, we prove the inequality. First observe that any s-strongly convex function F with respect to $\mathcal{L}$ satisfies

$$F(\mathrm{X}) - F(\mathrm{Y}) \geq \frac{s}{2}|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})|^2 \tag{3.11}$$

for any $\mathrm{X} \in \mathcal{K}$ and any $\mathrm{L} \in \mathcal{L}$ for $\mathrm{Y} = \arg\min_{\mathrm{Z} \in \mathcal{K}} F(\mathrm{Z})$. To see this, we use (3.10) (with replacement of R by F) due to the strong convexity of F and $\nabla F(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) \geq 0$ (otherwise Y would not be the minimizer since we can make a small step in the direction $\mathrm{X} - \mathrm{Y}$ and decrease the value of F.) See the proof of Lemma 2.8 of [68] for more detail.

Recall that the update rule of the FTRL is $\mathrm{X}_{t+1} = \arg\min_{\mathrm{X} \in \mathcal{K}} F_t(\mathrm{X})$ where $F_t(\mathrm{X}) = \sum_{i=1}^{t} \eta \mathrm{L}_i \bullet \mathrm{X} + R(\mathrm{X})$. Note that F_t is s-strongly convex with respect to $\mathcal{L}$ due to the linearity of $\mathrm{L}_i \bullet \mathrm{X}$. Applying (3.11) to F_t and F_{t-1} with $\mathrm{L} = \mathrm{L}_t$, we get

$$\begin{aligned} F_t(\mathrm{X}_t) &\geq F_t(\mathrm{X}_{t+1}) + \frac{s}{2}|\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1})|^2, \\ F_{t-1}(\mathrm{X}_{t+1}) &\geq F_{t-1}(\mathrm{X}_t) + \frac{s}{2}|\mathrm{L}_t \bullet (\mathrm{X}_{t+1} - \mathrm{X}_t)|^2. \end{aligned}$$

Summing up these two inequalities we get

$$\eta \mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1}) \geq s|\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1})|^2.$$

Dividing both side by $\mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}_{t+1})$ we get the desired result. □

Note that Lemma 5 gives a more general method of deriving regret bounds than the standard one given by Lemma 1. To see this, assume that the two conditions of Lemma 1 hold. Then, Cauchy-Schwarz inequality says that $|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})| \leq \|\mathrm{L}\|_* \|\mathrm{X} - \mathrm{Y}\| \leq g\|\mathrm{X} - \mathrm{Y}\|$ for every $\mathrm{L} \in \mathcal{L}$ and $\mathrm{X}, \mathrm{Y} \in \mathcal{K}$, where the second inequality is from the second condition. Thus, the first condition implies the condition of Lemma 5 with s replaced by s/g^2 as

$$\begin{aligned} R(\mathrm{X}) &\geq R(\mathrm{Y}) + \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) + \frac{s}{2}\|\mathrm{X} - \mathrm{Y}\|^2 \\ &\geq R(\mathrm{Y}) + \nabla R(\mathrm{Y}) \bullet (\mathrm{X} - \mathrm{Y}) + \frac{s}{2g^2}|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})^2|. \end{aligned}$$

Another advantage of using Lemma 5 is that we can avoid looking for appropriate norms to

obtain good regret bounds.

3.5.2 Strong convexity of the log-determinant regularizer

Now we prove the strong convexity of the log-determinant for our problem $(\tilde{\mathcal{K}}, \tilde{\mathcal{L}})$ defined in the beginning of this section. The following lemma provides a sufficient condition that turns out to be useful.

Lemma 6 (Christiano [21])**.** *Let* $\mathrm{X}, \mathrm{Y} \in \mathbb{S}_+^{N \times N}$ *be such that*

$$\exists (i,j) \in [N] \times [N], |X_{i,j} - Y_{i,j}| \geq \delta (X_{i,i} + X_{j,j} + Y_{i,i} + Y_{j,j}).$$

Then the following inequality holds:

$$-\ln\det(\alpha \mathrm{X} + (1-\alpha)\mathrm{Y}) \quad \leq \quad -\alpha \ln\det(\mathrm{X}) - (1-\alpha)\ln\det(\mathrm{Y}) - \frac{\alpha(1-\alpha)}{2}\frac{\delta^2}{72\sqrt{e}}.$$

Note that the original proof by Christiano only gives the order of the lower bound of the last term of $\Omega(\delta^2)$. So we give the proof with a constant factor. Now we give the proof of this lemma. To complete the proof, we need to show a series of definitions and technical lemmas.

First, we need to define some functions related to probability distributions. For a continuous probability distribution P, we abuse the notation P as its probability density function. The negative entropy function over the set of probability distributions P over $\mathbb{R}^N$ is defined as $H(P) = \mathbb{E}_{\boldsymbol{x} \sim P}[\ln P(\boldsymbol{x})]$. The total variation distance between probability distributions P and Q over $\mathbb{R}^N$ is defined as $\frac{1}{2}\int_{\boldsymbol{x}} |P(\boldsymbol{x}) - Q(\boldsymbol{x})| d\boldsymbol{x}$. The characteristic function of a probability distribution P over $\mathbb{R}^N$ is defined as $\phi(\boldsymbol{u}) = \mathbb{E}_{\boldsymbol{x} \sim P}[e^{i\boldsymbol{u}^\mathsf{T}\boldsymbol{x}}]$, where i is the imaginary unit.

The following lemma shows that the difference of the characteristic functions gives a lower bound of the total variation distance.

Lemma 7. *Let* P *and* Q *be probability distribution over* $\mathbb{R}^N$ *and* $\phi_P(\boldsymbol{u})$*,* $\phi_Q(\boldsymbol{u})$ *be their characteristic functions, respectively. Then,*

$$\max_{\boldsymbol{u} \in \mathbb{R}^N} |\phi_P(\boldsymbol{u}) - \phi_Q(\boldsymbol{u})| \leq \int_{\boldsymbol{x}} |P(\boldsymbol{x}) - Q(\boldsymbol{x})| d\boldsymbol{x}.$$

Proof.

$$\begin{aligned}\max_{\boldsymbol{u}\in\mathbb{R}^N}|\phi_P(\boldsymbol{u})-\phi_Q(\boldsymbol{u})| &= \max_{\boldsymbol{u}\in\mathbb{R}^N}\Big|\int_{\boldsymbol{x}} e^{i\boldsymbol{u}^T\boldsymbol{x}}P(\boldsymbol{x})d\boldsymbol{x}-\int_{\boldsymbol{x}} e^{i\boldsymbol{u}^T\boldsymbol{x}}Q(\boldsymbol{x})d\boldsymbol{x}\Big| \\ &\le \max_{\boldsymbol{u}\in\mathbb{R}^N}\int_{\boldsymbol{x}}|e^{i\boldsymbol{u}^T\boldsymbol{x}}||P(\boldsymbol{x})-Q(\boldsymbol{x})|d\boldsymbol{x} \\ &\le \int_{\boldsymbol{x}}|P(\boldsymbol{x})-Q(\boldsymbol{x})|d\boldsymbol{x}\end{aligned}$$

where we use the fact that $|e^{i\boldsymbol{u}^T\boldsymbol{x}}|=1$ for every $\boldsymbol{u}\in\mathbb{R}^N$. □

The negative entropy function is strongly convex with respect to the total variation distance.

In [21], the proof of the following lemma was given for only discrete entropies and the differential entropies are regarded as the limit of the discrete entropies, but this assertion is incorrect [22]. We fix the problem by considering the limit of the "difference" of discrete entropies as given in our proof.

Lemma 8 (Christiano [21])**.** *Let P and Q be continuous probability distributions over $\mathbb{R}^N$ with total variation distance δ. Then,*

$$H(\alpha P+(1-\alpha)Q)\le\alpha H(P)+(1-\alpha)H(Q)-\alpha(1-\alpha)\delta^2.$$

Proof. First we fix a discretization interval Δ. As in Sec 8.3 of [22], for a continuous distribution P we consider its discretization. Let we divide $\mathbb{R}^N$ be "tiles" with width Δ, namely $S_{\boldsymbol{j}}=\{\boldsymbol{x}\in\mathbb{R}^N:\forall i\in[N],x_i\in[j_i\Delta,(j_i+1)\Delta]\}$ where $\boldsymbol{j}\in\mathbb{N}^N$. By the mean-value theorem, there exists $\boldsymbol{x}_{\boldsymbol{j}}\in S_{\boldsymbol{j}}$ such that $P(\boldsymbol{x}_{\boldsymbol{j}})\Delta^N=\int_{S_{\boldsymbol{j}}}P(\boldsymbol{x})d\boldsymbol{x}$. Then we define the discretized distribution P^Δ over $\mathbb{N}^N$ as following:

$$P_{\boldsymbol{j}}=\int_{S_{\boldsymbol{j}}}P(\boldsymbol{x})d\boldsymbol{x}=P(\boldsymbol{x}_{\boldsymbol{j}})\Delta^N.$$

Thus we can define the discrete entropy $H(P^\Delta)$ and we have

$$H(P^\Delta)=\sum_{\boldsymbol{j}}P_{\boldsymbol{j}}\ln P_{\boldsymbol{j}}=\sum_{\boldsymbol{j}}(P(\boldsymbol{x}_{\boldsymbol{j}})\Delta^N)\ln(P(\boldsymbol{x}_{\boldsymbol{j}})\Delta^N)=\sum_{\boldsymbol{j}}\Delta^N P(\boldsymbol{x}_{\boldsymbol{j}})\ln P(\boldsymbol{x}_{\boldsymbol{j}})+N\ln\Delta$$

Thus for two continuous distributions P and Q, $\lim_{\Delta\to0}\big(H(P^\Delta)-H(Q^\Delta)\big)=H(P)-H(Q)$.

Next we consider the total variation distance $\delta^\Delta = \frac{1}{2}\sum_{\boldsymbol{j}} |P_{\boldsymbol{j}} - Q_{\boldsymbol{j}}|$ then we get

$$2\delta^\Delta = \sum_{\boldsymbol{j}} |P_{\boldsymbol{j}} - Q_{\boldsymbol{j}}| = \sum_{\boldsymbol{j}} |(P(\boldsymbol{x}_{\boldsymbol{j}})\Delta^N) - (Q(\boldsymbol{x}_{\boldsymbol{j}})\Delta^N)| = \sum_{\boldsymbol{j}} \Delta^N |P(\boldsymbol{x}_{\boldsymbol{j}}) - Q(\boldsymbol{x}_{\boldsymbol{j}})|$$

thus $\lim_{\Delta\to 0} \delta^\Delta = \delta$. Using these equalities, we can prove this lemma. □

The following lemma connects the entropy and the log-determinant.

Lemma 9 (Cover and Thomas [22])**.** *For any probability distribution P over $\mathbb{R}^N$ with 0 mean and covariance matrix Σ, its entropy is bounded by the log-determinant of covariance matrix. That is,*

$$H(P) \geq \frac{-1}{2}\ln(\det(\Sigma)(2\pi e)^N),$$

where the equality holds if and only if P is a Gaussian.

We need the following technical lemma.

Lemma 10. $e^{-\frac{x}{2}} - e^{-\frac{1-x}{2}} \geq \frac{e^{-1/4}}{2}(1-2x)$ *for* $0 \leq x \leq 1/2$

Proof. Since the function $f(x) = e^{-x/2} - e^{-(1-x)/2}$ is convex on $0 \leq x \leq 1/2$, its tangent at $x = 1/2$ always gives a lower bound of $f(x)$. Hence we get $f(x) \geq f'(1/2)(x - 1/2) + f(1/2) = e^{-1/4}(1-2x)/2$. □

The following lemma provides us a relation between covariance matrices and the total variation distance.

Lemma 11 (Christiano [21])**.** *Let $\mathcal{G}_1$ and $\mathcal{G}_2$ are zero-mean Gaussian distributions with covariance matrix Σ and Θ, respectively. If there exists $(i,j) \in [N] \times [N]$ such that*

$$|\Sigma_{i,j} - \Theta_{i,j}| \geq \delta(\Sigma_{i,i} + \Theta_{i,i} + \Sigma_{j,j} + \Theta_{j,j}),$$

then the total variation distance between $\mathcal{G}_1$ and $\mathcal{G}_2$ is at least $\frac{1}{12e^{1/4}}\delta$.

The original proof by Christiano gives an asymptotic bound of the form of $\Omega(\delta)$. Now we give the proof with a constant factor.

Proof. By Lemma 7, it is sufficient to derive a lower bound of the maximum of difference between characteristic functions. In this case, the characteristic functions of $\mathcal{G}_1$ and $\mathcal{G}_2$ are $\phi_1(\boldsymbol{u}) = e^{-\frac{1}{2}\boldsymbol{u}^T\Sigma\boldsymbol{u}}$ and $\phi_2(\boldsymbol{u}) = e^{-\frac{1}{2}\boldsymbol{u}^T\Theta\boldsymbol{u}}$, respectively.

Let $\alpha_1 = \boldsymbol{v}^T\Sigma\boldsymbol{v}, \alpha_2 = \boldsymbol{v}^T\Theta\boldsymbol{v}, \boldsymbol{u} = \frac{\boldsymbol{v}}{\sqrt{\alpha_1+\alpha_2}}$. Then,

$$\max_{\boldsymbol{u}\in\mathbb{R}^N} |\phi_1(\boldsymbol{u}) - \phi_2(\boldsymbol{u})| \geq \max_{\boldsymbol{v}\in\mathbb{R}^N} \left| e^{\frac{-\alpha_1}{2(\alpha_1+\alpha_2)}} - e^{\frac{-\alpha_2}{2(\alpha_1+\alpha_2)}} \right| \geq \max_{\boldsymbol{v}\in\mathbb{R}^N} \left| \frac{1}{2e^{1/4}} \frac{\alpha_1-\alpha_2}{\alpha_1+\alpha_2} \right|.$$

Note that we use Lemma 10 in the last inequality.

By the assumption, we have for some (i, j) that

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

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

On the other hand,

$$\begin{aligned} (\boldsymbol{e}_i + \boldsymbol{e}_j)^\mathsf{T}(\Sigma + \Theta)(\boldsymbol{e}_i + \boldsymbol{e}_j) &= (\Sigma + \Theta)_{i,i} + (\Sigma + \Theta)_{j,j} + 2(\Sigma + \Theta)_{i,j} \\ &\leq 2(\Sigma + \Theta)_{i,i} + 2(\Sigma + \Theta)_{j,j}. \end{aligned}$$

In the last inequality we use $\Sigma+\Theta \in \mathbb{S}_+^{N\times N}$ and the fact that $X_{i,j} \leq \frac{1}{2}(X_{i,i}+X_{j,j})$ for symmetric semi-definite matrix X. So,

$$\forall \boldsymbol{v} \in \{\boldsymbol{e}_i, \boldsymbol{e}_j, \boldsymbol{e}_i + \boldsymbol{e}_j\}, \boldsymbol{v}^\mathsf{T}(\Sigma + \Theta)\boldsymbol{v} \leq 2(\Sigma + \Theta)_{i,i} + 2(\Sigma + \Theta)_{j,j}$$

and thus we have

$$\max_{\boldsymbol{u}\in\mathbb{R}^N} |\phi_1(\boldsymbol{u}) - \phi_2(\boldsymbol{u})| \geq \max_{\boldsymbol{v}\in\{\boldsymbol{e}_i,\boldsymbol{e}_j,\boldsymbol{e}_i+\boldsymbol{e}_j\}} \left| \frac{1}{2e^{1/4}} \frac{\boldsymbol{v}^\mathsf{T}(\Sigma - \Theta)\boldsymbol{v}}{\boldsymbol{v}^\mathsf{T}(\Sigma + \Theta)\boldsymbol{v}} \right| \geq \frac{\delta}{6e^{1/4}}.$$

□

Now we are ready to give a proof of Lemma 6.

Proof. Let $\mathcal{G}_1, \mathcal{G}_2$ are zero-mean Gaussian distributions with covariance matrix $\Sigma = \mathrm{X}, \Theta = \mathrm{Y}$, respectively. By the assumption and Lemma 11, total variation distance between $\mathcal{G}_1$ and $\mathcal{G}_2$ is at least $\frac{\delta}{12e^{1/4}}$. For simplicity of notation, let $\tilde{\delta} = \frac{\delta}{12e^{1/4}}$. Consider the entropy of the following probability distribution of $\boldsymbol{v}$; with probability α, $\boldsymbol{v} \sim \mathcal{G}_1$, with remaining probability $1 - \alpha$,

$\boldsymbol{v} \sim \mathcal{G}_2$. Its covariance matrix is $\alpha\Sigma + (1-\alpha)\Theta$. By Lemma 8 and 9,

$$\begin{aligned} -\ln\det(\alpha\Sigma + (1-\alpha)\Theta) &\leq 2H(\alpha\mathcal{G}_1 + (1-\alpha)\mathcal{G}_2) + \ln(2\pi e)^N \\ &\leq 2\alpha H(\mathcal{G}_1) + 2(1-\alpha)H(\mathcal{G}_2) + \ln(2\pi e)^N - \alpha(1-\alpha)\tilde{\delta}^2 \\ &= -\alpha\ln\det(\Sigma) - (1-\alpha)\ln\det(\Theta) - \alpha(1-\alpha)\tilde{\delta}^2. \end{aligned}$$

□

The following lemma shows that the sufficient condition actually holds for our problem $(\tilde{\mathcal{K}}, \tilde{\mathcal{L}})$ for $\delta = O(|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})|)$, which establishes the strong convexity of the log-determinant regularizer. The lemma is a slight generalization of [21] in that loss matrices are not necessarily block-wise sparse.

Lemma 12. *Let* $\mathrm{X}, \mathrm{Y} \in \mathbb{S}^{N\times N}$ *be such that* $X_{i,i} \leq \beta'$ *and* $Y_{i,i} \leq \beta'$ *for every* $i \in [N]$. *Then, for any* $\mathrm{L} \in \tilde{\mathcal{L}}$, *there exists* $(i,j) \in [N] \times [N]$ *such that*

$$|X_{i,j} - Y_{i,j}| \geq \frac{|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})|}{4g_1\beta'}(X_{i,i} + X_{j,j} + Y_{i,i} + Y_{j,j}).$$

Proof. By Cauchy-Schwarz inequality,

$$|\mathrm{L} \bullet (\mathrm{X} - \mathrm{Y})| \leq \|\mathrm{vec}(\mathrm{L})\|_1 \|\mathrm{vec}(\mathrm{X} - \mathrm{Y})\|_\infty \leq g_1 \max_{i,j} |X_{i,j} - Y_{i,j}|.$$

Thus the lemma follows since $X_{i,i} + X_{j,j} + Y_{i,i} + Y_{j,j} \leq 4\beta'$. □

Applying Lemma 12 to $\mathrm{X} + \epsilon\mathrm{E}$ and $\mathrm{Y} + \epsilon\mathrm{E}$ for $\mathrm{X}, \mathrm{Y} \in \tilde{\mathcal{K}}$ and $\beta' = \beta + \epsilon$, and then applying Lemma 6, we immediately get the following proposition.

Proposition 10. *The log-determinant regularizer* $R(\mathrm{X}) = -\ln\det(\mathrm{X}+\epsilon\mathrm{E})$ *is* s*-strongly convex with respect to* $\tilde{\mathcal{L}}$ *for* $\tilde{\mathcal{K}}$ *with* $s = 1/(1152\sqrt{e}g_1^2(\beta+\epsilon)^2)$.

Combining this proposition with Lemma 5, we can derive a regret bound.

Theorem 11 (Main theorem of Chapter 3)**.** *For the online SDP problem* $(\tilde{\mathcal{K}}, \tilde{\mathcal{L}})$, *the FTRL with the log-determinant regularizer* $R(\mathrm{X}) = -\ln\det(\mathrm{X} + \epsilon\mathrm{E})$ *achieves*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) \leq \frac{\tau}{\epsilon\eta} + \eta \times 1152\sqrt{e}g_1^2(\beta+\epsilon)T$$

Especially, for appropriate choices of η and ϵ,

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}) \leq 175 g_1 \sqrt{\beta \tau T}.$$

Proof. By Proposition 10 we know that R is s-strongly convex for $s = 1/(1152\sqrt{e} g_1^2 (\beta + \epsilon)^2)$. It remains to give a bound on $H_0 = R(\mathrm{X}_0) - R(\mathrm{X}_1)$, where X_0 and X_1 be the maximizer and the minimizer of R in $\tilde{\mathcal{K}}$, respectively. Let $\lambda_i(\mathrm{X})$ be the i-th eigenvalue of X. Then,

$$\begin{aligned} R(\mathrm{X}_0) - R(\mathrm{X}_1) &= \sum_{i=1}^{N} \ln \frac{\lambda_i(\mathrm{X}_1) + \epsilon}{\lambda_i(\mathrm{X}_0) + \epsilon} \\ &\leq \sum_{i=1}^{N} \ln \left(\frac{\lambda_i(\mathrm{X}_1)}{\epsilon} + 1 \right) \leq \sum_{i=1}^{N} \frac{\lambda_i(\mathrm{X}_1)}{\epsilon} \leq \frac{\tau}{\epsilon}. \end{aligned}$$

Note that we use the inequality $\ln(x+1) \leq x$ for $-1 < x$. Applying Lemma 5 with $\epsilon = \beta$, we get the theorem. □

Since the OMP problem $(\mathcal{W}, G)$ for a (β, τ)-decomposable decision space $\mathcal{W}$ can be reduced to the online SDP problem $(\tilde{\mathcal{K}}, \tilde{\mathcal{L}})$ with $g_1 = 4G$, Proposition 6 implies the following regret bound for the OMP problem.

Corollary 12. *For the OMP problem $(\mathcal{W}, G)$ where $\mathcal{W} \subseteq [-1, 1]^{m \times n}$ is (β, τ)-decomposable, there exists an algorithm that achieves*

$$Reg_{\mathrm{OMP}}(T, \mathcal{W}) = O(G\sqrt{\beta \tau T}).$$

Note that the bound does not depend on the size (m or n) of matrices and improves by a factor of $O(\sqrt{m+n})$ from Corollary 8. Accordingly, we get $O(\sqrt{\ln n})$ improvements for the three application problems:

Online max-cut has a regret bound of $O(G\sqrt{nT})$.

Online gambling has a regret bound of $O(G \ln n \sqrt{nT})$.

Online collaborative filtering has a regret bound of $O(G\sqrt{\tau T \sqrt{n}})$ for $n \geq m$.

All these bounds match the lower bounds given in [38] up to constant factors.

3.5.3 The vector case

We can apply the results obtained above to the vector case by just restricting the decision and loss spaces to diagonal matrices. That is, our problem $(\tilde{\mathcal{K}}, \tilde{\mathcal{L}})$ is now rewritten as

$$\begin{aligned}\tilde{\mathcal{K}} &= \{\mathrm{diag}(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{R}^N_+, \|\boldsymbol{x}\|_\infty \leq \beta, \|\boldsymbol{x}\|_1 \leq \tau\}, \text{ and} \\ \tilde{\mathcal{L}} &= \{\mathrm{diag}(\boldsymbol{\ell}) : \boldsymbol{\ell} \in \mathbb{R}^N, \|\boldsymbol{\ell}\|_1 \leq g_1\},\end{aligned}$$

and the log-determinant is a variant of the Burg entropy $R(\mathrm{diag}(\boldsymbol{x})) = -\sum_{i=1}^N \ln(x_i + \epsilon)$. Applying Theorem 11 to the problem, we have $O(g_1\sqrt{\beta\tau T})$ regret bound.

Curiously, unlike the matrix case, we can also apply the standard technique, namely, Theorem 5 (with a slight modification), to get the same regret bound. To see this, observe that $\|\mathrm{diag}(\boldsymbol{x})\|_{\mathrm{Sp}} = \|\boldsymbol{x}\|_\infty \leq \beta$ for every $\mathrm{diag}(\boldsymbol{x}) \in \tilde{\mathcal{K}}$, and $\|\mathrm{diag}(\boldsymbol{\ell})\|_{\mathrm{Tr}} = \|\boldsymbol{\ell}\|_1 \leq g_1$ for every $\mathrm{diag}(\boldsymbol{\ell}) \in \tilde{\mathcal{L}}$. These imply that $\tilde{\mathcal{K}} \subseteq \mathcal{K}_\infty$ with $\sigma = \beta$ and $\tilde{\mathcal{L}} \subseteq \mathcal{L}_1$ with $\gamma_1 = g_1$. Moreover, as shown in the proof of Theorem 11, we have $\max_{\mathrm{X},\mathrm{X}' \in \tilde{\mathcal{K}}}(R(\mathrm{X}) - R(\mathrm{X}')) \leq \tau/\epsilon$. So, $N \ln 2$ in Theorem 5 can be replaced by τ/ϵ, and hence we get a regret bound of $4g_1\sqrt{\beta\tau T}$.

3.6 Conclusion

In this chapter, we consider the online symmetric positive semi-definite matrix prediction problem. We proposed an FTRL-based algorithm with the log-determinant regularization. We tighten and generalize existing analyses. As a result, we show that the log-determinant regularizer is effective when loss matrices are sparse. Reducing online collaborative filtering task to the online SDP problems with sparse loss matrices, our algorithms obtain optimal regret bounds.

Our future work includes (i) improving a constant factor in the regret bound, (ii) applying our method to other online prediction tasks with sparse loss settings including the "vector" case, (iii) developing a fast implementation of our algorithm.

Chapter 4

Optimal mistake bounds for binary matrix completion the log-determinant regularization

4.1 Introduction

Learning preferences over a set of items is an important task in recommendation systems. A typical formulation of learning preferences is the matrix completion problem, where the underlying matrix represents user's preferences of items, only some entries of the matrix are given and the goal is to predict rest of preferences. Under the natural assumption that the matrix has low rank or low trace-norm, a lot of work have been done in various settings, say, the statistical i.i.d. setting [71, 31, 69, 56] and the online learning setting [20, 38, 59, 42]

Recently, another direction of work is done using a different notion from the rank. Herbster et al. [39] considered an online version of the binary matrix completion problem and prove a mistake bound for the problem using the notion of the margin complexity of the matrix. Furthermore, the mistake bound is comparable to that of the perceptron with the best kernel without knowing it. Their mistake bound, however, is not optimal yet. More precisely, the upper and lower bounds have a logarithmic gap.

In this chapter, we close the gap by showing a new algorithm for the online binary matrix completion task with matching lower/upper mistake bounds.

The online binary matrix completion problem posed by Herbseter et al. [39] is defined in the following way. Let m and n are the numbers of users and items respectively. We consider a matrix $\mathrm{U} \in \{-1, 1\}^{m \times n}$ where each component $U_{i,j}$ denotes the user i likes or dislikes the

item j, for $i = 1, \ldots, m$ and $j = 1, \ldots, n$. Our task is to predict U in an online fashion. More precisely, we assume the following protocol; For each round $t = 1, \ldots, T$, the algorithm (i) receives $(i_t, j_t) \in [m] \times [n]$, (ii) predicts $\hat{y}_t \in \{-1, 1\}$, (iii) receives $y_t \in \{-1, 1\}$ and (iv) suffers the loss $\ell_t = \mathbf{1}_{y_t \neq \hat{y}_t}$. Thus, we assume that there is an underlying binary matrix $\mathrm{U} \in \{-1, 1\}^{n \times m}$. Let $\mathcal{M}$ be a set of rounds algorithm made mistake $\mathcal{M} = \{t \in [T] : \hat{y}_t \neq y_t\}$. The goal of the algorithm is to minimize the total loss (i.e. the number of errors) $|\mathcal{M}|$.

We adopt the assumption of [39] that there exists a matrix $\mathrm{P} \in \mathbb{R}^{m \times k}$ (matrix of m users) and $\mathrm{Q} \in \mathbb{R}^{n \times k}$ (matrix of n items), where k is possibly infinite, that for each user $i \in [m]$ and item $j \in [n]$, $U_{i,j} = \mathrm{sgn}(\mathrm{P}_{i,*}\mathrm{Q}_{j,*}^{\mathsf{T}})$. In other words, there exists a k-dimensional feature space in which each user $i \in [m]$ and item $j \in [n]$ have their own feature vectors $\mathrm{P}_{i,*}$ and $\mathrm{Q}_{j,*}$ and the weight vector $\mathrm{P}_{i,*}$ of each user i correctly classifies the feature vector $\mathrm{Q}_{j,*}$ of item j. This is a kind of realizability assumption. Apparently, the assumption looks strong, but we also assume that the algorithm knows neither P and Q nor the feature space itself. Therefore, the assumption is more generic than it looks.

For the online binary matrix completion problem, we propose an FTRL-based algorithm with the log-determinant regularizer and prove mistake bounds of the algorithm under the realizability assumption. As mentioned, the mistake bounds are optimal, up to the constant factor. The technical contributions of our results consist of (i) a sharper reduction technique from the online binary matrix completion to the online SDP problem [38] and (ii) an application of a modified notion of the strong convexity for the log-determinant [59, 58]. These techniques might be interesting in their own rights.

We also apply our online algorithm in the statistical (batch) learning setting using the online-to-batch conversion framework (see, e.g., [57]), and derive a generalization error bound. In this setting, we also can drop the logarithmic factor from the result of [39]. Our generalization error bound is similar to the known margin-based bound obtained by the best kernel in hindsight. Additionally, we introduce the standard notion of the margin loss [57] to measure the difficulty of the given sample in the both settings. This is a tighter measure than the margin error, the quantity [39] used.

4.2 Related work

As mentioned in the previous section, [39] propose an algorithm for the online binary matrix completion problem with a mistake bound $O((m+n)\log(m+n)/\gamma^2 + \mathrm{merr}(\mathcal{S}, \gamma))$, where $\mathcal{S}$ is

the sequence of (i_t, j_t, y_t) and $\mathrm{merr}(\mathcal{S}, \gamma)$ is a measure of the complexity of $\mathcal{S}$ called the *margin error*, which we formally define in next section. The key idea of [39] is, reducing the binary matrix completion problem to an online prediction task over positive semi-definite matrices, namely the online semi-definite programming task (online SDP, for short). Then Herbster et al. apply the Matrix Winnow algorithm [77]. This algorithm can be regarded as a variant of the matrix multiplicative weights algorithm [74].

For online prediction tasks, there is a standard framework called Follow the Regularized Leader (FTRL) for designing algorithms [65, 68, 37], for which we need to choose an appropriate regularization function (or regularizer) as a parameter to obtain a good regret bound. As mentioned in previous chapter, the matrix negative entropy $R(\mathrm{X}) = \mathrm{Tr}(\mathrm{X} \ln \mathrm{X} - \mathrm{X})$ is widely used as the regularizer for some online SDP tasks[74, 38].

On the other hand, the log-determinant $R(\mathrm{X}) = -\ln \det(\mathrm{X})$ is also used as the regularizer for positive semi-definite matrices [21, 59]. As shown in the previous chapter, for the online SDP problem, the key point of the analysis of FTRL-based algorithm is the strong convexity of the regularizer with respect to the loss space. We also know that the log-determinant regularization works well when the loss information is "sparse" [59].

In batch learning setting, [72] also considers the assumption that the rating matrix was generated from implicit classification of feature vectors of items by each user's hyperplane. They show generalization error bounds with the matrix trace-norm or the max-norm and the margin of the hyperplane. The matrix completion from 1-bit observations is also well studied [24, 11]. In their setting, the algorithm can observe binary information according to the uniform distribution over $[m] \times [n]$. The value of observed entry is probabilistically determined by the true rating matrix $\mathrm{M} \in \mathbb{R}^{m \times n}$. [24] show some generalization error bounds of learning M and probability distribution which determines binary observations from given sample with assumption that the trace norm of M is bounded. [11] show generalization error bounds for learning a low-rank matrix M. [18] also study the 1-bit matrix completion with bounding the max norm of M, and provide distribution-free generalization error bounds.

4.3 Preliminaries

In this section, we define some mathematical notations.

We write the sequence which the algorithm received as $\mathcal{S} = ((i_t, j_t, y_t) : t \in [T]) \in ([m] \times [n] \times \{-1, 1\})^T$.

For a binary matrix $\mathrm{U} \in \{-1,1\}^{m\times n}$ and a sequence $\mathcal{S}$, we define the consistent set $\mathrm{cons}(\mathcal{S}) = \{\mathrm{M} \in \mathbb{R}^{m\times n} : \forall(i,j,y), yM_{i,j} > 0\}$, the sign pattern set $\mathrm{SP}(\mathrm{U}) = \{\mathrm{M} \in \mathbb{R}^{m\times n} : \forall(i,j), U_{i,j}M_{i,j} > 0\}$.

The margin of U is defined as follows [53]:

$$\mathrm{m}(\mathrm{U}) = \sup_{\mathrm{PQ}^\intercal \in \mathrm{SP}(\mathrm{U})} \min_{i,j} \frac{|\mathrm{P}_{i,*}\mathrm{Q}_{j,*}^\intercal|}{\|\mathrm{P}_{i,*}\|_2 \|\mathrm{Q}_{j,*}\|_2}. \tag{4.1}$$

Note that this definition of $\mathrm{m}(\mathrm{U})$ is the inverse of the margin complexity originally used in [39]. We define the margin of a sequence $\mathcal{S}$ as $\mathrm{m}(\mathcal{S}) = \sup_{\mathrm{M}\in\mathrm{cons}(\mathcal{S})} \mathrm{m}(\mathrm{M})$.

The margin error is defined as :

$$\mathrm{merr}(\mathcal{S},\gamma) = \inf_{\mathrm{PQ}^\intercal \in \mathrm{cons}(\mathcal{S})} \left| \left\{ (i,j,y) \in \mathcal{S} : \frac{|\mathrm{P}_{i,*}\mathrm{Q}_{j,*}^\intercal|}{\|\mathrm{P}_{i,*}\|_2 \|\mathrm{Q}_{j,*}\|_2} < \gamma \right\} \right|. \tag{4.2}$$

Let $\phi_\gamma(x)$ be a function such that

$$\phi_\gamma(x) = \begin{cases} 0 & \gamma \le x \\ 1 - x/\gamma & 0 \le x \le \gamma \\ 1 & x \le 0. \end{cases}$$

Then we define the empirical margin loss as :

$$\mathrm{mloss}(\mathcal{S},\gamma) = \inf_{\mathrm{PQ}^\intercal \in \mathrm{cons}(\mathcal{S})} \sum_{(i,j,y)\in\mathcal{S}} \phi_\gamma \left(\frac{|\mathrm{P}_{i,*}\mathrm{Q}_{j,*}^\intercal|}{\|\mathrm{P}_{i,*}\|_2 \|\mathrm{Q}_{j,*}\|_2} \right) \tag{4.3}$$

Thus, if $\gamma \le \mathrm{m}(\mathcal{S})$ then $\mathrm{merr}(\mathcal{S},\gamma) = 0$ and $\mathrm{mloss}(\mathcal{S},\gamma) = 0$. This margin loss function ϕ_γ is standard choice in statistical learning tasks of hyperplanes [57] .

4.4 Reduction of the binary matrix prediction

In this section we show the reduction technique of the binary matrix prediction to an online SDP task which we consider in Chapter 3.

Let a decision space $\mathcal{K} \subseteq \mathbb{S}_+^{N\times N}$ be a convex set, let a loss space $\mathcal{L} \subseteq \mathbb{S}^{N\times N}$ and let $(\mathcal{K},\mathcal{L})$ be a online SDP task described in Chapter 3. The regret is defined as;

$$Reg_{\mathrm{SDP}}(T,\mathcal{K},\mathcal{L},\mathrm{X}^\star) = \sum_{t=1}^{T} \mathrm{L}_t \bullet \mathrm{X}_t - \sum_{t=1}^{T} \mathrm{L}_t \bullet \mathrm{X}^\star \tag{4.4}$$

where $\mathrm{X}^\star \in \mathcal{K}$ is an competitor matrix.

We adopt the Follow the Regularized Leader(FTRL) to design an algorithm for the online SDP. Let $R : \mathcal{K} \to \mathbb{R}$ be a convex function called the regularizer. A FTRL-based algorithm with the regularizer R and learning rate $\eta > 0$ makes prediction as;

$$\mathrm{X}_{t+1} = \arg\min_{\mathrm{X} \in \mathcal{K}} R(\mathrm{X}) + \eta \sum_{s=1}^{t} \mathrm{L}_t \bullet \mathrm{X}. \tag{4.5}$$

4.4.1 Details of the reduction

Key ideas of the reduction are positive semi-definite embedding of the decision set and convexification of the loss function by the hinge loss.

We describe how to embed a prediction $\hat{\mathrm{U}}$ into a positive semi-definite matrix. One can predict y_t without mistake over the sequence $\mathcal{S}$ if predictions are according to some $\tilde{\mathrm{U}} \in \mathrm{cons}(\mathcal{S})$. Let the positive semi-definite embedding of this $\tilde{\mathrm{U}}$ be $\tilde{\mathrm{X}}$ and we design the decision set of online SDP $\mathcal{K}$ to include $\tilde{\mathrm{X}}$. The regret bound ensures that algorithm's predictions $(\mathrm{X}_t)_{t\in[T]}$ are close to $\tilde{\mathrm{X}}$.

Now we design the decision set $\mathcal{K}$. Let $\mathrm{P} \in \mathbb{R}^{m\times k}, \mathrm{Q} \in \mathbb{R}^{n\times k}$ are matrices such that $\mathrm{PQ}^\mathsf{T} \in \mathrm{cons}(\mathcal{S})$,

$$\bar{\mathrm{P}} = \mathrm{diag}\Big(\frac{1}{\|\mathrm{P}_{1,*}\|_2}, \dots, \frac{1}{\|\mathrm{P}_{m,*}\|_2}\Big)\mathrm{P} \in \mathbb{R}^{m\times k} \text{ and } \bar{\mathrm{Q}} = \mathrm{diag}\Big(\frac{1}{\|\mathrm{Q}_{1,*}\|_2}, \dots, \frac{1}{\|\mathrm{Q}_{n,*}\|_2}\Big)\mathrm{Q} \in \mathbb{R}^{n\times k}.$$

Then we choose $\tilde{\mathrm{U}} = \theta\bar{\mathrm{P}}\bar{\mathrm{Q}}^\mathsf{T} \in \mathrm{cons}(\mathcal{S})$ and its entries are $\theta(\bar{\mathrm{P}}\bar{\mathrm{Q}}^\mathsf{T})_{i,j} = \theta\frac{\mathrm{P}_{i,*}\mathrm{Q}_{j,*}^\mathsf{T}}{\|\mathrm{P}_{i,*}\|_2\|\mathrm{Q}_{j,*}\|_2}$. We design its positive semi-definite embedding $\tilde{\mathrm{X}}$ as

$$\tilde{\mathrm{X}} = \theta \begin{bmatrix} \bar{\mathrm{P}} \\ \bar{\mathrm{Q}} \end{bmatrix} \begin{bmatrix} \bar{\mathrm{P}} \\ \bar{\mathrm{Q}} \end{bmatrix}^\mathsf{T} = \theta \begin{bmatrix} \bar{\mathrm{P}}\bar{\mathrm{P}}^\mathsf{T} & \bar{\mathrm{P}}\bar{\mathrm{Q}}^\mathsf{T} \\ (\bar{\mathrm{P}}\bar{\mathrm{Q}}^\mathsf{T})^\mathsf{T} & \bar{\mathrm{Q}}\bar{\mathrm{Q}}^\mathsf{T} \end{bmatrix}. \tag{4.6}$$

Thus the set $\mathcal{K} = \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : X_{i,i} \le \theta\}$ includes $\tilde{\mathrm{X}}$ and we employ this set as the decision set of the online SDP.

Next, we describe the design of loss matrices on the online SDP. To ease the analysis of the 0-1 loss $\mathbf{1}[\hat{y}_y \neq y_t]$, we convexify this by the hinge loss. Let the hinge loss function with margin ρ be $h(x) = \max(1 - \frac{x}{\rho}, 0)$. The total loss measured by h upper bounds the total number of mistake.

Let $\mathrm{Z}_t = \frac{1}{2}(\boldsymbol{e}_{i_t} + \boldsymbol{e}_{m+j_t})(\boldsymbol{e}_{i_t} + \boldsymbol{e}_{m+j_t})^\mathsf{T}$ then $\tilde{\mathrm{X}} \bullet \mathrm{Z}_t - \theta = \theta(\bar{\mathrm{P}}\bar{\mathrm{Q}}^\mathsf{T})_{i_t,j_t}$. We employ the

prediction of $\hat{y}_t = \mathrm{sgn}(\mathrm{Z}_t \bullet \mathrm{X}_t - \theta)$ at the round t where $\mathrm{X}_t \in \mathcal{K}$ is the prediction of the algorithm for the online SDP task and the loss is defined as $h(y_t(\mathrm{X}_t \bullet \mathrm{Z}_t - \theta))$.

Now we have the decision set $\mathcal{K}$ and loss functions for each t. The regret of this problem is

$$Reg_h(T, \mathcal{K}, \mathrm{X}^\star) = \sum_{t=1}^{T} (h(y_t(\mathrm{X}_t \bullet \mathrm{Z}_t - \theta)) - h(y_t(\mathrm{X}^\star \bullet \mathrm{Z}_t - \theta))),$$

where $\mathrm{X}^\star \in \mathcal{K}$ is a competitor matrix.

By the convexity of h, we can bound Reg_h by the sub-gradient of h, this is a standard linearization technique (see, e.g., [68]). Let L_t be the sub-gradient of h;

$$\mathrm{L}_t = \nabla_{\mathrm{X}} h(y_t(\mathrm{X} \bullet \mathrm{Z}_t - \theta)) = \begin{cases} -\frac{y_t}{\rho}\mathrm{Z}_t & y_t(\mathrm{X} \bullet \mathrm{Z}_t - \theta) \leq \rho \\ 0 & \text{otherwise.} \end{cases}$$

Then we have

$$Reg_h(T, \mathcal{K}, \mathrm{X}^\star) \leq \sum_{t=1}^{T} \mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}^\star) = \sum_{t \in \mathcal{M}_h} \mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}^\star)$$

where $\mathcal{M}_h = \{t \in [T] : \mathrm{L}_t \neq 0\}$. This quantity can be regarded as the regret of a online SDP task with loss matrices $(\mathrm{L}_t)_{t\in[T]}$, namely $Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star) = \sum_{t=1}^{T} \mathrm{L}_t \bullet (\mathrm{X}_t - \mathrm{X}^\star)$ where $\mathcal{L}$ is the set of all possible L_t.

Now we have $Reg_h \leq Reg_{\mathrm{SDP}}$. In general, we cannot bound the number of mistake $|\mathcal{M}|$ from this inequality due to the relation between $|\mathcal{M}_h|$ and $|\mathcal{M}|$ is unclear. But if $\mathcal{S}$ is consistent (i.e. $\exists \mathrm{U} \in \{-1, 1\}^{m\times n}$ such that $\forall t, U_{i_t,j_t} = y_t$) and enforce the algorithm to be mistake-driven, we can bound the number of mistakes the algorithm made.

Theorem 13. *Let matrices* P *and* Q *be chosen to realize the infimum of the equation* (4.3) *and* $\tilde{\mathrm{X}}$ *be an embedding of* U *defined in the equation* (4.6)*. Assume that an FTRL-based algorithm* $\mathcal{A}$ *for an online SDP task has a regret bound* $Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}})$*. Then running* $\mathcal{A}$ *with the sequence of loss matrices*

$$\mathrm{L}_t = \begin{cases} -\frac{y_t}{\rho}\mathrm{Z}_t & t \in \mathcal{M} \\ 0 & t \notin \mathcal{M} \end{cases}$$

satisfies the following inequality;

$$|\mathcal{M}| \leq Reg_{\mathrm{SDP}}(|\mathcal{M}|, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}}) + \mathrm{mloss}(\mathcal{S}, \rho/\theta).$$

Proof. By the definition of the FTRL update rule and $\mathrm{L}_t = 0$ if $t \notin \mathcal{M}$, we have

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}}) = Reg_{\mathrm{SDP}}(|\mathcal{M}|, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}}).$$

The regret on the online SDP Reg_{SDP} can be lower bounded as;

$$\begin{aligned} Reg_{\mathrm{SDP}}(|\mathcal{M}|, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}}) &= \sum_{t\in\mathcal{M}} (\frac{-y_t}{\rho}\mathrm{Z}_t \bullet \mathrm{X}_t - \frac{-y_t}{\rho}\mathrm{Z}_t \bullet \tilde{\mathrm{X}}) \\ &= \sum_{t\in\mathcal{M}} (\frac{-y_t}{\rho}(\mathrm{Z}_t \bullet \mathrm{X}_t - \theta) - \frac{-y_t}{\rho}(\mathrm{Z}_t \bullet \tilde{\mathrm{X}} - \theta)) \geq \sum_{t\in\mathcal{M}} \frac{y_t}{\rho}(\mathrm{Z}_t \bullet \tilde{\mathrm{X}} - \theta) \end{aligned}$$

where the last inequality we use if $t \in \mathcal{M}$ then $y_t \neq \mathrm{sgn}(\mathrm{Z}_t \bullet \mathrm{X}_t - \theta)$. For the simplicity of the notation, we write $S_t = \frac{\mathrm{P}_{i_t,*}\mathrm{Q}_{j_t,*}^{\mathsf{T}}}{\|\mathrm{P}_{i_t,*}\|_2\|\mathrm{Q}_{j_t,*}\|_2}$. By the definition of Z_t and $\tilde{\mathrm{X}}$ the Frobenius product $\mathrm{Z}_t \bullet \tilde{\mathrm{X}} - \theta = \theta S_t$. Let

$$\mathcal{M}_- = \{t \in \mathcal{M} : y_t S_t \leq \rho/\theta\} \text{ and } \mathcal{M}_+ = \{t \in \mathcal{M} : y_t S_t \geq \rho/\theta\}$$

then we have

$$Reg_{\mathrm{SDP}}(|\mathcal{M}|, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}}) \geq \sum_{t\in\mathcal{M}} \frac{y_t}{\rho}(\mathrm{Z}_t \bullet \tilde{\mathrm{X}} - \theta) = \sum_{t\in\mathcal{M}_-} \frac{\theta}{\rho} y_t S_t + \sum_{t\in\mathcal{M}_+} \frac{\theta}{\rho} y_t S_t.$$

The definition of $\mathcal{M}_+$ implies $y_t S_t \theta/\rho \geq 1$ for all $t \in \mathcal{M}_+$ and we have $\sum_{t\in\mathcal{M}_+} y_t S_t \theta/\rho \geq |\mathcal{M}_+|$. For $t \in \mathcal{M}_-$ we have $0 \leq y_t S_t \leq \rho/\theta$ thus

$$y_t S_t = \frac{\rho}{\theta}(1 - \phi_{\rho/\theta}(y_t S_t))$$

and

$$\sum_{t\in\mathcal{M}_-} \frac{\theta}{\rho} y_t S_t = |\mathcal{M}_-| - \sum_{t\in\mathcal{M}_-} \phi_{\rho/\theta}(y_t S_t) = |\mathcal{M}_-| - \sum_{t\in\mathcal{M}} \phi_{\rho/\theta}(y_t S_t) = |\mathcal{M}_-| - \mathrm{mloss}(\mathcal{S}, \rho/\theta),$$

where the second equality we use $\phi_{\rho/\theta}(y_t S_t) = 0$ for $t \in \mathcal{M}_+$.

Combining them, we get a lower bound of Reg_{SDP} as

$$|\mathcal{M}| \leq Reg_{\mathrm{SDP}}(|\mathcal{M}|, \mathcal{K}, \mathcal{L}, \tilde{\mathrm{X}}) + \mathrm{mloss}(\mathcal{S}, \rho/\theta).$$

□

By this theorem, we can fix one of θ or ρ at arbitrary value and then $\rho/\theta = \gamma$ be a new parameter.

4.5 Our algorithm and analysis

In this section, we describe our algorithm and derive its mistake bound. We adopt the FTRL-based algorithm with the log-determinant regularization $R(\mathrm{X}) = -\ln\det(\mathrm{X}+\epsilon\mathrm{E})$ to this online SDP problem. Of course $\mathrm{L}_t = 0$ if $t \notin \mathcal{M}$, hence we can avoid the optimization on the update rule and employ $\mathrm{X}_t = \mathrm{X}_{t+1}$ in these rounds.

The pseudo code of our algorithm is shown in Algorithm 1.

Algorithm 1 Binary matrix completion by Log-determinant regularization

Parameters $\theta > 0, \rho > 0, \epsilon > 0, \eta > 0$
Initialize $\mathrm{X}_1 = \theta\mathrm{E}$.
for $t = 1, 2, \ldots, T$ **do**
  Receive (i_t, j_t).
  Let $\mathrm{Z}_t = \frac{1}{2}(\boldsymbol{e}_{i_t} + \boldsymbol{e}_{m+j_t})(\boldsymbol{e}_{i_t} + \boldsymbol{e}_{m+j_t})^\mathsf{T}$.
  Predict $\hat{y}_t = \mathrm{sgn}(\mathrm{X}_t \bullet \mathrm{Z}_t - \theta)$ and receive y_t.
  if $y_t(\mathrm{W}_t \bullet \mathrm{X}_t - \theta) \leq 0$ **then**
    Let $\mathrm{L}_t = \frac{-y_t}{\rho}\mathrm{Z}_t$ and

$$\mathrm{X}_{t+1} = \arg\min_{\mathrm{X}\in\mathcal{K}} -\ln\det(\mathrm{X}+\epsilon\mathrm{E}) + \eta\sum_{s=1}^{t}\mathrm{X}\bullet\mathrm{L}_s.$$

  else
    Let $\mathrm{L}_t = 0$ and $\mathrm{X}_{t+1} = \mathrm{X}_t$.
  end if
end for

Now we derive the regret bound and the mistake bound. As shown in Theorem11, for any online SDP task $(\mathcal{K}, \mathcal{L})$ and competitor matrix $\mathrm{X}^\star$, the FTRL with the regularizer $R(\mathrm{X}) = -\ln\det(\mathrm{X}+\epsilon\mathrm{E})$ satisfies

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star) \leq O(g_1^2(\beta+\epsilon)^2 T)\eta + \frac{\tau}{\epsilon\eta}, \tag{4.7}$$

for the decision space $\mathcal{K} = \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : X_{i,i} \leq \beta, \mathrm{Tr}(\mathrm{X}) \leq \tau\}$ and the loss space $\mathcal{L} = \{\mathrm{L} \in \mathbb{S}^{N\times N} : \|\mathrm{vec}(\mathrm{L})\|_1 \leq g_1\}$.

In our situation, parameters are $N = m+n, \beta = \theta, \tau = N\theta$ and $g_1 = 2/\rho$. As mentioned

in previous section, we put $\rho/\theta = \gamma$ as the new parameter. Now we have

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star) \leq O(T/\gamma^2)\eta + (m+n)/\eta, \tag{4.8}$$

by setting $\epsilon = \theta$. Additionally setting $\eta = \sqrt{\frac{\gamma^2}{NT}}$ leads the following corollary.

Corollary 14. *For any* $\mathrm{X}^\star$*, the FTRL with the regularizer* $R(\mathrm{X}) = -\ln\det(\mathrm{X} + \theta\mathrm{E})$ *running with the loss sequence* $\mathrm{L}_t = \nabla_{\mathrm{X}} h(y_t(\mathrm{X} \bullet \mathrm{Z}_t - \theta))$ *satisfies the following regret bound.*

$$Reg_{\mathrm{SDP}}(T, \mathcal{K}, \mathcal{L}, \mathrm{X}^\star) = O(\frac{1}{\gamma}\sqrt{(m+n)T}). \tag{4.9}$$

Applying Theorem 13 to inequality (4.8), we have the mistake bound.

Theorem 15. *Assume that* $\mathrm{cons}(\mathcal{S}) \neq \emptyset$ *and* $\mathrm{X}^\star = \tilde{\mathrm{X}}$ *be the same as defined in Theorem 13. Then Algorithm* (1) *mistakes at most*

$$|\mathcal{M}| = O\left(\frac{m+n}{\gamma^2}\right) + 2\mathrm{mloss}(\mathcal{S}, \gamma). \tag{4.10}$$

Proof. By applying Theorem 13 to inequality (4.8) and rearranging, we have

$$(1 - \eta/\gamma^2)O(|\mathcal{M}|) \leq (m+n)/\eta + \mathrm{mloss}(\mathcal{S}, \gamma).$$

Setting $\eta = \gamma^2/2$ we get the theorem. □

Of course, optimizing γ is impossible due to the dependency of $\mathcal{S}$. Fortunately, as shown in Algorithm 2, we can use the standard "doubling trick" as same as [39] to avoid tuning γ.

Algorithm 2 Binary matrix completion with the doubling trick

1: Set $\kappa = \sqrt{2}$
2: **for** $t = 1, 2, \ldots,$ **do**
3: Run Algorithm 1 with parameter $\gamma = 1/\kappa$ until it has made $\lceil (m+n)\kappa^2 \rceil$ mistakes.
4: Set $\kappa \leftarrow \kappa \times \sqrt{2}$
5: **end for**

Corollary 16. *For any* $\gamma^\star \geq 0$*, Algorithm 2 satisfies*

$$|\mathcal{M}| = O\left(\frac{m+n}{(\gamma^\star)^2}\right) + 2\mathrm{mloss}(\mathcal{S}, \gamma^\star). \tag{4.11}$$

Proof. By slightly modifying the original proof on [39], replacing the definition of $A(\gamma)$ and $B(\gamma)$ in original proof, we get the proof in our case.

For any $\gamma > 0$, we define $A(\gamma) = (m+n)/\gamma^2$, $B(\gamma) = 2\mathrm{merr}(\mathcal{S}, \gamma)$ and $M(\gamma) = A(\gamma) + B(\gamma)$. Since $A(\cdot)$ is continuous, positive, monotonically decreasing and $B(\cdot)$ is monotonically non-decreasing, we can define η such that for all $\gamma \leq \eta$ we have $A(\gamma) \geq B(\gamma)$ and for all $\gamma \geq \eta$ we have $A(\gamma) \leq B(\gamma)$.

Let ϵ be such that $A(\eta+\epsilon) \geq A(\eta) - 1/2$. For $\gamma \leq \eta+\epsilon$ we have $M(\gamma) \geq A(\gamma) \geq A(\eta+\epsilon)$ and for $\gamma \geq \eta + \epsilon$ we have $M(\gamma) \geq B(\gamma) \geq B(\eta+\epsilon) \geq A(\eta+\epsilon)$. Either case we have $M(\gamma) \geq A(\eta+\epsilon) \geq A(\eta) - 1/2$.

Let z be the minimum integer power of $\sqrt{2}$ that is greater than or equal to $1/\eta$. Since $1/z \leq \eta$ we have $B(1/z) \leq B(\eta) \leq A(\eta) \leq A(1/z)$ so $M(1/z) = A(1/z) + B(1/z) \leq 2A(1/z) = 2(m+n)z^2$ and hence the doubling algorithm above terminates at some $\kappa \leq z$. The total number of mistakes M made by Algorithm 2 is bounded by

$$\begin{aligned}
&(2A(2^{-1/2}) + 1) + (2A(2^{-2/2}) + 1) + (2A(2^{-3/2}) + 1) + \ldots + (2A(1/z) + 1) \\
&\qquad \geq 3A(2^{-1/2}) + 3A(2^{-2/2}) + 3A(2^{-3/2}) + \ldots + 3A(1/z) \\
&\qquad \geq 3(m+n)(2 + 2^2 + \ldots + z^2) \\
&\qquad = 6(m+n)z^2 - 6 = 6A(1/z) - 6.
\end{aligned}$$

We also have $z \leq \sqrt{2}/\eta$ so $A(1/z) \leq 2A(\eta)$ and putting together we get $M \leq 12A(\eta) - 6$. □

Now we consider the lower bound. We can say that the upper bound of the mistake of our algorithm matches the lower bound shown in [39].

Definition 7 ([39]). *The class of (k,l)-biclusterd matrices is defined as*

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

Theorem 17 ([39]). *Given $l \in [n]$ and an online algorithm $\mathcal{A}$ that predicts entries of $\mathrm{U} \in \mathbb{B}_{m,l}^{m\times n}$, an adversary can construct a sequence $\mathcal{S}$ such that $\mathrm{m}(\mathcal{S}) \geq 1/\sqrt{l}$ and on this sequence $\mathcal{A}$ will make at least $l \times m$ mistake.*

We need the following lemma.

Lemma 13 ([39]). *If $\mathrm{U} \in \mathbb{B}_{k,l}^{m\times n}$ then $\mathrm{m}(\mathrm{U})^2 \geq 1/\min(k,l)$.*

One can take the "transpose" of the Theorem 17 i.e. an adversary also can construct a $\tilde{\mathcal{S}}$ s.t. $\mathrm{m}(\tilde{\mathcal{S}}) \geq 1/\sqrt{k}$ and $\mathcal{A}$ will make at least $k \times n$ mistake for any $k \in [m]$. Henceforth, we assume $m \geq n$ and $l \leq k$ without loss of generality.

Now we compare our result and Theorem 17.

Theorem 18. *For any* $\mathrm{U} \in \mathbb{B}^{m \times n}_{m,l}$ *and* $\mathcal{S}$ *consistent with* U*, Algorithm 1 with* $\gamma = 1/\sqrt{l}$ *makes mistake at most* $O(l \times m)$ *on the sequence* $\mathcal{S}$*.*

Proof. By the same argument on Section 4.4.1, for any $\mathrm{U} \in \mathbb{B}^{m \times n}_{m,l}$, we can embed U in $\mathrm{X}^\star \in \mathcal{K} = \{\mathrm{X} \in \mathbb{S}^{N \times N}_+ : X_{i,i} \leq 1\}$ (w.l.o.g we set $\theta = 1$ and $\gamma = \rho$) and Algorithm 1 runs over this decision set.

By Lemma 13, the margin of the sequence $\mathcal{S}$ is $\mathrm{m}(\mathcal{S}) \geq 1/\sqrt{l}$. So setting $\gamma = 1/\sqrt{l}$ leads $\mathrm{mloss}(\mathcal{S}, \gamma) = 0$. Combining Theorem 15, we have the mistake bound of $|\mathcal{M}| \leq O((m+n)l) = O(ml)$. □

Thus, our algorithm has optimal mistake bound up to a constant factor.

4.6 Connection to the batch setting

In this section, we derive a generalization error bound of our algorithm based on its mistake bound. In the batch learning, the algorithm is given a sample $\mathcal{S}$ all at once, and learns a matrix $\mathrm{X} \in \mathcal{W}$ from $\mathcal{S}$. Finally, algorithm predicts $\hat{y} \in \{-1, 1\}$ using the learned hypothesis X.

First, we describe our batch setting precisely. We fix a true matrix $\mathrm{U} \in \{-1, 1\}^{m \times n}$ to be learned and a probability distribution $\mathcal{D}$ over $[m] \times [n]$, which is unknown to the learner. We consider the problem under the standard PAC learning framework. Let $\mathcal{F}$ be a set of functions $[m] \times [n]$ maps to $\{-1, 1\}$, which is called the hypothesis class. The input to the learner is a sequence of triples

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

where each $(i_t, j_t) \in [m] \times [n]$ is independently chosen according to the distribution $\mathcal{D}$. When given the sample $\mathcal{S}$, the learner produce a hypothesis $f \in \mathcal{F}$, so that its generalization error is as small as possible (with high probability over the random choice of $\mathcal{S}$). The generalization error of f is defined as

$$L(f) = \mathbb{E}_{(i,j)\sim\mathcal{D}}[\ell(f(i,j); U_{i,j})] \tag{4.12}$$

for a fixed loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$. If the loss function ℓ is 0-1 loss $\ell(x, y) = \mathbf{1}[x \neq y]$, the generalization error is equivalent to the probability of the algorithm makes mistake $L(f) = \Pr_{(i,j)\sim\mathcal{D}}[f(i, j) \neq U_{i,j}]$.

In our situation, we restrict our hypothesis as $f(i, j) = \mathrm{sgn}(\mathrm{X} \bullet \mathrm{Z} - \theta)$ where $\mathrm{Z} = \frac{1}{2}(\boldsymbol{e}_i + \boldsymbol{e}_{m+j})(\boldsymbol{e}_i + \boldsymbol{e}_{m+j})^\mathsf{T}$ and $\mathrm{X} \in \mathcal{K} = \{\mathrm{X} \in \mathbb{S}_+^{N\times N} : X_{i,i} \leq \theta\}$. Thus, the hypothesis can be represented by a symmetric positive semi-definite matrix X. The loss function is the 0-1 loss $\ell(f(i, j); y) = \mathbf{1}_{[f(i,j)\neq y]}$ or the hinge loss $\ell(f(i, j); \mathrm{U}_{i,j}) = h(y(\mathrm{X} \bullet \mathrm{Z} - \theta))$.

Now we describe how to use Algorithm 1 in the batch setting and show its generalization error bound. Our application of Algorithm 1 in the batch setting is shown in Algorithm 3;

Algorithm 3 Binary matrix completion by Log-determinant regularization in the batch setting

1: Parameters $\theta > 0, \rho > 0, \epsilon > 0, \eta > 0$
2: Receive the sample set $\mathcal{S}$ of size T and sort $\mathcal{S}$ randomly.
3: Initialize $\mathrm{X}_1 = \theta\mathrm{E}$.
4: **for** $t = 1, 2, \ldots, T$ **do**
5: Run Algorithm 1 with input (i_t, j_t, y_t) as the t-th member of $\mathcal{S}$, and store X_t.
6: **end for**
7: Choose $\mathrm{X} \in \{\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_T\}$ uniformly random.
8: Predict $f(i, j) = \mathrm{sgn}(\mathrm{X} \bullet \mathrm{Z} - \theta)$ where $\mathrm{Z} = \frac{1}{2}(\boldsymbol{e}_i + \boldsymbol{e}_{m+j})(\boldsymbol{e}_i + \boldsymbol{e}_{m+j})^\mathsf{T}$.

To bound the generalization error, we use the following lemma.

Lemma 14 (Lemma 7.1 of [57])**.** *Let $\mathcal{S}$ be a sample drawn i.i.d. according to $\mathcal{D}$. ℓ is bounded by B. f_t be the hypothesis generated by an online algorithm by sequentially processing $\mathcal{S}$. Then for any $\delta > 0$, with probability at least $1 - \delta$, the following inequality holds;*

$$\frac{1}{T}\sum_{t=1}^{T} L(f_t) \leq \frac{1}{T}\sum_{t=1}^{T} \ell(f_t(i_t, j_t); y_t) + B\sqrt{\frac{2\log 1/\delta}{T}} \tag{4.13}$$

Proof. For any $t \in [T]$, let V_t be a random variable $V_t = L(f_t) - \ell(f_t(i_t, j_t); y_t) \in [-B, B]$. Then for any t, we have

$$\mathbb{E}[V_t|(i_t, j_t)_{t=1}^{t-1}] = L(f_t) - \mathbb{E}[\ell(f_t(i_t, j_t); y_t)|f_t] = L(f_t) - L(f_t) = 0.$$

Using Azuma's inequality, $\Pr[\sum_{t=1}^{T} V_t/T \geq \epsilon] \leq e^{-2T\epsilon^2/(2B)^2}$. Let $\delta = e^{-2T\epsilon^2/(2B)^2}$ then we get the lemma. □

Applying this lemma to Algorithm 3 with the fact that $\sum_{t=1}^{T} L(f_t)/T = \mathbb{E}[L(f)]$ by choosing X_t uniformly randomly, we have the following bound.

Theorem 19. *For any $\delta > 0$, with probability at least $1 - \delta$, Algorithm 3 satisfies the following two inequalities. If $\ell(f(i,j); U_{i,j}) = h(y(\mathrm{X} \bullet \mathrm{Z} - \theta))$ then for any $\mathrm{X}^\star \in \mathcal{K}$,*

$$\mathbb{E}[L(f)] \leq O\left(\sqrt{\frac{m+n}{\gamma^2 T}}\right) + \frac{1}{T}\sum_{t=1}^{T} h(y_t(\mathrm{X}^\star \bullet \mathrm{Z}_t - \theta))) + (1 + \frac{\theta}{\rho})\sqrt{\frac{2\log 1/\delta}{T}}. \tag{4.14}$$

If $\ell(f(i,j); U_{i,j}) = \mathbf{1}_{[} f(i,j) \neq y]$ and $\mathrm{cons}(\mathcal{S}) \neq \emptyset$ then for $\mathrm{X}^\star$ defined in Theorem 13,

$$\mathbb{E}[L(f)] \leq \frac{1}{T} \times O\left(\frac{m+n}{\gamma^2} + \mathrm{mloss}(\mathcal{S}, \gamma)\right) + \sqrt{\frac{2\log 1/\delta}{T}}. \tag{4.15}$$

In both cases, the expectation is by the randomness of choosing X_t of the algorithm.

Now we compare our generalization bound with the following known bound based on kernel method.

Theorem 20 (Corollary 5.1 of [57])**.** *Consider a classification task with given a sample $\mathcal{S} = ((x_1, y_1), \ldots, (x_T, y_T)) \in (\mathcal{X} \times \{-1, 1\})^T$. Let $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a positive definite symmetric kernel with $r = \sup_{x \in \mathcal{X}} K(x, x)$. Let $\mathbb{H}$ be a Hilbert space and $\Phi : \mathcal{X} \to \mathbb{H}$ be the feature mapping associated with K. Let $\mathcal{F} = \{x \mapsto \boldsymbol{w}^T \Phi(\cdot) : \|\boldsymbol{w}\|_{\mathbb{H}} \leq \Lambda\}$ for some $\Lambda \geq 0$. Fix $\gamma > 0$ then for any $\delta > 0$, the following inequality holds with probability at least $1 - \delta$ for any $f \in \mathcal{F}$:*

$$L(f) \leq \frac{1}{T}\sum_{t=1}^{T} \phi_\gamma(y_t f(x_t)) + 2\sqrt{\frac{r^2\Lambda^2}{\gamma^2 T}} + \sqrt{\frac{\log 1/\delta}{2T}} \tag{4.16}$$

In our case, the norm of the instance (i.e. $\bar{\mathrm{Q}}_{j,*}$) and the norm of the hyperplane (i.e. $\bar{\mathrm{P}}_{i,*}$) are equal to 1 due to the normalization. The corollary shows that excess risk term of a hyperplane with some kernel is bounded by $O(\sqrt{\frac{1}{\gamma^2 T}})$ in this case. Thus if we use such a kernel based methods in our situation, there are m individual learners to learn each classification tasks and this term grows $O(m)$ times in the whole execution of the algorithm.

On the other hand, in our bound, we obtain a similar sample complexity with the one obtained by minimizing the right hand of side of (4.16) over any possible feature mapping and linear hypotheses in the feature space. Of course, we have to learn m hyperplanes simultaneously, so our bound has $O(m)$ factor (w.l.o.g., we assume $m \geq n$). On the other hand, we can remove the square root by the separability assumption $\mathrm{cons}(\mathcal{S}) \neq \emptyset$.

4.7 Implementation details

Herbster et al. [39] uses $R(\mathrm{X}) = \mathrm{Tr}(\mathrm{X}\ln\mathrm{X} - \mathrm{X})$ as the regularizer. One of the advantage of this reguarizer function is, one can solve the FTRL update rule analytically.

On the other hand, our FTRL update rule

$$\arg\min_{\mathrm{X}} -\ln\det(\mathrm{X}+\epsilon\mathrm{E}) + \mathrm{L}\bullet\mathrm{X} \text{ s.t. } \forall i\in[N], X_{i,i}\leq\beta, \mathrm{X}\in\mathbb{S}_+^{N\times N} \tag{4.17}$$

do not admit any analytical solution due to $-\ln\det(\mathrm{X}+\epsilon\mathrm{E})$ cannot force X be positive semi-definite. This is one of shortcomings of our algorithm.

Of course, this optimization problem is an convex optimization so one can solve this in polynomial time by optimizing $O(N^2)$ variables but this is too slow for large N. In this appendix, we describe the implementation techniques with some relaxation to avoid time consuming projections onto $\mathbb{S}_+^{N\times N}$. For the sake of simplicity of the notation, we write $f(\mathrm{X}) = -\ln\det(\mathrm{X}+\epsilon\mathrm{E}) + \mathrm{L}\bullet\mathrm{X}$.

We use the proximal gradient descent algorithm (see, e.g., [63]) to solve (4.17) shown in the following pseudo code in Algorithm 4.

Algorithm 4 FTRL update with the log-determinant regularization

1: Parameters $\beta > 0, \mathrm{L}\in\mathbb{S}^{N\times N}$, step size ν_t
2: Initialize $\mathrm{X}_1 = \beta\mathrm{E}$.
3: **while** until converge **do**
4: $\quad\mathrm{V}_{t+1} = \mathrm{X}_{t-1} - \nu_t\nabla_{\mathrm{X}} f(\mathrm{X}_{t-1})$.
5: $\quad\mathrm{X}_{t+1} = \arg\min_{\mathrm{X}} \frac{1}{2}\|\mathrm{X}-\mathrm{V}_{t+1}\|_{\mathrm{Fr}}^2$ s.t. $\mathrm{X}\in\mathcal{K}$.
6: **end while**

Note that $\nabla_{\mathrm{X}} f(\mathrm{X}) = -(\mathrm{X}+\epsilon\mathrm{E})^{-1} + \mathrm{L}$. Unfortunately, still we cannot analytically solve the optimization problem

$$\arg\min_{\mathrm{X}} \frac{1}{2}\|\mathrm{X}-\mathrm{V}\|_{\mathrm{Fr}}^2 \text{ s.t. } \mathrm{X}\in\mathbb{S}_+^{N\times N}, X_{i,i}\leq\beta. \tag{4.18}$$

Now we relax the semi-definite constraint $\mathrm{X}\in\mathbb{S}_+^{N\times N}$ on this problem by using a log-barrier function. The relaxed problem is,

$$\arg\min_{\mathrm{X}} \frac{1}{2}\|\mathrm{X}-\mathrm{V}\|_{\mathrm{Fr}}^2 - \mu\ln\det\mathrm{X} \text{ s.t. } \mathrm{X}^\mathsf{T} = \mathrm{X}, X_{i,i}\leq\beta. \tag{4.19}$$

Thanks to this barrier function, we can reduce the number of variables $O(N^2)$ to $O(N)$. Note

that by the update rule on the proximal gradient descent, V is symmetric.

We introduce Lagrange multiplier $\boldsymbol{\alpha} \in \mathbb{R}^N_+$ and $\Gamma \in \mathbb{R}^{N\times N}$ then the Lagrangean function of Problem (4.19) is

$$Lag(\mathrm{X}\boldsymbol{\alpha}, \Gamma) = \frac{1}{2}\|\mathrm{X} - \mathrm{V}\|^2_{\mathrm{Fr}} - \mu \ln \det \mathrm{X} + \sum_{i=1}^{N} \alpha_i (X_{i,i} - \beta) + \mathrm{X} \bullet (\Gamma - \Gamma^{\mathsf{T}}). \tag{4.20}$$

Let a $N \times N$ diagonal matrix which has α_i as the (i,i)-th element by $\mathrm{diag}(\boldsymbol{\alpha}) = \mathrm{A}$. Setting $\nabla_{\mathrm{W}} Lag = 0$, we have

$$\nabla_{\mathrm{X}} Lag(\mathrm{X}, \boldsymbol{\alpha}, \Gamma) = \mathrm{X} - \mathrm{V} - \mu \mathrm{X}^{-1} + \mathrm{A} + \Gamma^{\mathsf{T}} - \Gamma = 0. \tag{4.21}$$

The symmetricity constraint forces

$$0 = \mathrm{X} - \mathrm{V} - \mu \mathrm{X}^{-1} + \mathrm{A} + \Gamma^{\mathsf{T}} - \Gamma = \mathrm{X} - \mathrm{V} - \mu \mathrm{X}^{-1} + \mathrm{A} + \Gamma - \Gamma^{\mathsf{T}}.$$

This leads $\Gamma^{\mathsf{T}} - \Gamma = 0$. We have a matrix quadratic equation $0 = \mathrm{X}^2 + (\mathrm{A} - \mathrm{V})\mathrm{X} - \mu \mathrm{E}$ and its solution is $\mathrm{X}(\boldsymbol{\alpha}) = \frac{1}{2}(\sqrt{(\mathrm{A} - \mathrm{V})^2 + 4\mu \mathrm{E}} - (\mathrm{A} - \mathrm{V})^2)$. Thus we can get a solution of Problem (4.19) by maximizing $Lag(\mathrm{X}(\boldsymbol{\alpha}), \boldsymbol{\alpha})$ (for short, we write this as $Lag(\boldsymbol{\alpha})$). Substitute $\mathrm{X}(\boldsymbol{\alpha})$ and rearranging we get

$$Lag(\boldsymbol{\alpha}) = \frac{1}{4}\mathrm{Tr}(\mathrm{QD}) - \frac{1}{4}\mathrm{Tr}(\mathrm{D}^2) + \mu \ln \det(\mathrm{Q} + \mathrm{D}) - \boldsymbol{\alpha}^{\mathsf{T}} \boldsymbol{\beta} + Const.$$

where $\mathrm{D} = \mathrm{A} - \mathrm{V}, \mathrm{Q} = \sqrt{\mathrm{D}^2 + 4\mu \mathrm{E}}$. Note that in this case D and Q are commute. Now the variable is only $\boldsymbol{\alpha}$.

To solve this maximization more easily, we derive the derivative of $Lag(\boldsymbol{\alpha})$. Let I_i be the $N \times N$ matrix with only (i,i)-th element is 1 and others are 0. Using the chain rules and derivative of matrix functions (see, e.g., [64, 55]), we get

$$\begin{aligned}
\frac{\partial}{\partial \alpha_i}\mathrm{Tr}(\mathrm{QD}) &= \mathrm{Tr}\left(\frac{\partial \mathrm{Q}}{\partial \alpha_i}\mathrm{D} + \mathrm{QI}_i\right), \\
\frac{\partial}{\partial \alpha_i}\mathrm{Tr}(\mathrm{D}^2) &= 2\mathrm{Tr}(\mathrm{I}_i \mathrm{D}), \\
\frac{\partial}{\partial \alpha_i} \ln \det(\mathrm{Q} + \mathrm{D}) &= \mathrm{Tr}\left((\mathrm{Q} + \mathrm{D})^{-1}(\frac{\partial \mathrm{Q}}{\partial \alpha_i} + \mathrm{I}_i)\right).
\end{aligned}$$

The derivative $\frac{\partial \mathrm{Q}}{\partial \alpha_i}$ can be calculated as,

$$\frac{\partial}{\partial \alpha_i}\mathrm{QQ} = \frac{\partial}{\partial \alpha_i}(\mathrm{D}^2 + 4\mu\mathrm{E}) = \mathrm{I}_i\mathrm{D} + \mathrm{DI}_i = \mathrm{Q}\frac{\partial \mathrm{Q}}{\partial \alpha_i} + \frac{\partial \mathrm{Q}}{\partial \alpha_i}\mathrm{Q}$$

where the last equality we use the product rule. This type of matrix equation is known as a Lyapunov equation. Now $-\mathrm{Q}$ is negative definite and the solution of the equation was given as $\frac{\partial \mathrm{Q}}{\partial \alpha_i} = \int_0^\infty e^{-\mathrm{Q}x}\mathrm{S}_i e^{-\mathrm{Q}x}dx$ where $\mathrm{S}_i = \mathrm{I}_i\mathrm{D} + \mathrm{DI}_i$ (see Corollary 11.9.4 on [10]). Thus if matrix M commutes with Q, we have

$$\mathrm{Tr}(\mathrm{M}\frac{\partial \mathrm{Q}}{\partial \alpha_i}) = \int_0^\infty \mathrm{Tr}(\mathrm{M}e^{-\mathrm{Q}x}\mathrm{S}_i e^{-\mathrm{Q}x})dx = \mathrm{Tr}(\mathrm{MS}_i\int_0^\infty e^{-2\mathrm{Q}x}dx) = \frac{1}{2}\mathrm{Tr}(\mathrm{MS}_i\mathrm{Q}^{-1})$$

where the last equality we use $\int_0^\infty e^{-2\lambda x}dx = \frac{1}{2\lambda}$ and the property of matrix exponential with eigenvalue decomposition. Using this derivative with the fact that D and $(\mathrm{Q}+\mathrm{D})^{-1}$ are commutable with Q, we get the derivative of the Lagrangean;

$$\frac{\partial}{\partial \alpha_i}Lag(\boldsymbol{\alpha}) = \frac{1}{8}\mathrm{Tr}(\mathrm{DS}_i\mathrm{Q}^{-1}) + \frac{1}{4}\mathrm{Tr}(\mathrm{QI}_i) - \frac{1}{2}\mathrm{Tr}(\mathrm{I}_i\mathrm{D}) + \mu\mathrm{Tr}\left((\mathrm{Q}+\mathrm{D})^{-1}\left(\frac{\mathrm{S}_i\mathrm{Q}^{-1}}{2} + \mathrm{I}_i\right)\right) - \beta.$$

4.8 Conclusion

In this chapter, we consider the binary matrix completion problem and derive the optimal mistake bounds under a mild realizability assumption. We also apply our online algorithm in the batch setting by applying a standard online-to-batch conversion technique, and derive a generalization error bound which is compatible with the best kernel in hindsight without explicitly knowledge what the best kernel is.

One of our future work is experimental evaluation of our algorithm over real data. In Section 4.7, we provide some implementation techniques of our algorithm, making an update under the FTRL faster by avoiding the projection onto $\mathbb{S}_+^{N\times N}$. But an implementation of the log-determinant regularization is still harder than existing methods using the negative entropy. Reducing a constant factor is also our task. As mentioned in Chapter 3, the constant factor on the regret bound of log-determinant regularization is very large.

Chapter 5

Tighter generalization error bounds for matrix completion

5.1 Introduction

Learning preferences of users over a set of items is an important task in recommendation systems. In particular, the collaborative filtering approach is known to be quite effective and popular [66, 52, 80]. Simply put, the collaborative filtering is an approach of inferring user's rating for an item, which the user has not rated yet, from the existing ratings of other users. The approach is formulated as the *matrix completion problem*, that is, learning a user-item rating matrix from given partial entries of the matrix. More formally, we consider a (true) rating matrix $\mathrm{X} \in \mathbb{R}^{m \times n}$ to be learned, where m and n are the numbers of users and items, respectively, and each component $X_{i,j}$ corresponds to user i's rating for item j. The task is to find a hypothesis matrix $\hat{\mathrm{X}} \in \mathbb{R}^{m \times n}$ that approximates the true matrix X when only some of components of X are given as a sample.

A common assumption in the previous work is that the true matrix $\mathrm{X} \in \mathbb{R}^{m \times n}$ can be well approximated by a matrix of low rank (or low trace norm, as a convex relaxation of the rank constraint). In other words, we assume that our hypothesis matrix $\hat{\mathrm{X}} \in \mathbb{R}^{m \times n}$ can be decomposed as $\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^{\mathsf{T}}$ for some $\mathrm{U} \in \mathbb{R}^{m \times K}$ and $\mathrm{V} \in \mathbb{R}^{n \times K}$ with a small number K, where K gives an upper bound of the rank of $\hat{\mathrm{X}}$. Generalization ability of algorithms (such as the empirical risk minimization) using low rank or low trace norm matrices is intensively studied in the literature (see, e.g., [70, 71, 72, 31]).

Recently, further additional constraints on the class of hypothesis matrices turns out to be effective in practice. One of the major approach is the regularization by bounding some norm

of U and V [51]. Another important approach is to impose the constraints that U and V are non-negative [48, 33, 79, 54, 28]. Such a decomposition is called the non-negative matrix factorization (NMF, for short). More precisely, a typical scheme of the NMF approach on collaborative filtering is the empirical risk minimization with the norm regularization, formulated as the following optimization problem: Find non-negative matrices $\mathrm{U} \in \mathbb{R}_+^{m\times K}$ and $\mathrm{V} \in \mathbb{R}_+^{n\times K}$ that minimizes

$$\sum_{(i,j)} \ell((\mathrm{U}\mathrm{V}^\mathsf{T})_{i,j}, X_{i,j}) + \alpha\|\mathrm{U}\|^2 + \beta\|\mathrm{V}\|^2,$$

where the sum is over all components (i, j) in the sample that the learner observes, ℓ is a fixed loss function, α, β are regularization parameters, and $\|\mathrm{U}_{i,*}\|$ and $\|\mathrm{V}_{j,*}\|$ are some norm. The L_2-norm regularization is widely adopted [48, 54]. The Mahalanobis norm defined via graph which models the relation between users and items are also used for regularization [33]. Note that this formulation is essentially equivalent to the empirical risk minimization:

$$\min_{\mathrm{U}\in\mathbb{R}_+^{m\times K},\mathrm{V}\in\mathbb{R}_+^{n\times K}} \sum_{(i,j)} \ell((\mathrm{U}\mathrm{V}^\mathsf{T})_{i,j}, \mathrm{X}_{i,j}) \quad \text{sub.to} \quad \|\mathrm{U}\| \le a, \quad \|\mathrm{V}\| \le b$$

for appropriate choices of a and b. Despite the empirical success of the NMF approach, no theoretical justification has been given.

In this chapter, we consider different but closely related classes of hypothesis matrices and give generalization bounds of these classes. Our bounds are of $O(\sqrt{(nK + m\log K)/T})$, where T is the sample size. These bounds improve the previously known $\tilde{O}(\sqrt{(nK + mK)/T})$ bound that is derived for the class where only the rank K constraint is imposed [70]. Therefore, our results would give theoretical evidence for the empirical success of the NMF. However, our new bounds hold even when U and V have negative values in some components. This result suggests that our analysis may not yet fully capture the property of the non-negativity constraint, or the empirical success of the NMF may not rely on the non-negativity very much but mostly on the regularization. Now we give our hypothesis classes.

Convex combination constraints: The first one is the class of all matrices $\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^\mathsf{T} \in \mathbb{R}^{m\times n}$ with $\mathrm{U} \in \mathbb{R}_+^{m\times K}$ and $\mathrm{V} \in \mathbb{R}^{n\times K}$ such that for every $i \in [m]$ and $j \in [n]$,

$$\|\mathrm{U}_{i,*}\|_1 = 1, \quad \|\mathrm{V}_{j,*}\|_\infty \le B.$$

That is, each row $\hat{\mathrm{X}}_{i,*}$ is a convex combination of the vectors $\mathrm{V}_{k,*}^\mathsf{T}$ with weights $\mathrm{U}_{i,k}$. This class

has a natural interpretation, just as the topic model for document classification [12, 13, 4], that the rating of each user i (row of $\hat{\mathrm{X}}$) is a convex combination of the ratings of K latent "model" users (rows of V^{T}), where the model users are allowed to assign ratings in $[-B, B]$. There are some studies on the topic model like approach for recommendation tasks[76, 7] and showing some experimental results.

L_1 and L_∞ norm constraints: The second one is the class of all matrices $\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^{\mathsf{T}} \in \mathbb{R}^{m \times n}$ with $\mathrm{U} \in \mathbb{R}^{m \times K}$ and $\mathrm{V} \in \mathbb{R}^{n \times K}$ such that for every $i \in [m]$ and $j \in [n]$,

$$\|\mathrm{U}_{i,*}\|_1 \leq 1, \quad \|\mathrm{V}_{j,*}\|_\infty \leq B.$$

Note that any matrix $\hat{\mathrm{X}}$ of rank K has a factorization $\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^{\mathsf{T}}$ with $\|\mathrm{U}_{i,*}\|_1 \leq 1$, because otherwise we can replace U by $c\mathrm{U}$ and V by V/c for a sufficiently small constant c. In other words, any matrix $\hat{\mathrm{X}}$ of rank K is in the second class for some parameter B.

Our technical contributions are twofold. The first one is that we develop a new technique for bounding the Rademacher complexity to derive generalization bounds. The second one is that we prove a matching lower bound of the Rademacher complexity of the first hypothesis class. This means that our generalization bounds are tightest among those derived from the Rademacher complexity argument. There are few results in the literature on deriving lower bounds of the Rademacher complexity.

Our generalization bounds suggest to use the structural risk minimization scheme, where the rank K and the bound B of L_∞ constraints on V are parameters to tune. Note that, by setting $B = \infty$, the scheme is reduced to the standard scheme with the low-rank constraint only. In our experiments on real data sets, our scheme with appropriate choices of B performs better than other schema with only the rank constraint (i.e., the case $B = \infty$) or the trace norm constraint.

5.2 Related work

Before describing our results, we briefly review some related results on the collaborative filtering problem. As mentioned in the previous section, Srebro et al. [70] give a generalization bound $\tilde{O}(\sqrt{K(m+n)/T})$ for the class of rank K matrices. There are many results for the class of matrices with the trace norm constraint, which is a convex relaxation of the rank constraint. Srebro, Shraibman and Foygel [71, 31] estimate the expected Rademacher complexity of the class of low trace norm matrices and their generalization bound is $O(\tau\sqrt{(n+m)/(Tmn)})$

where τ is the bound of the trace-norm. Shamir and Shalev-Shwartz [69] also give a generalization bound $O(\sqrt{\tau(\sqrt{m}+\sqrt{n})/T})$. The parameter τ is usually scaled as $\tau = O(\sqrt{nmK})$ [69], and in this setting the both bounds are not less than $\tilde{O}(\sqrt{K(m+n)/T})$.

There are some results which exploit prior knowledge of the distribution $\mathcal{D}$ that generates the sample [47, 46], while in our setting we only consider the worst case generalization ability. [3] derives a generalization bound under Bernstein condition, which is between $\sqrt{(m+n)K\log(m+n)/T}$ and $(m+n)K\log(m+n)/T$. Natarajan and Jain [61] handle non-decomposable loss function, which cannot be written as a sum of element wise loss functions.

We emphasize that all of these result have $(n+m)K$ or $(n+m)A$ factors in the generalization bound, where A is an upper bound of the trace norm scaled to be compatible to the rank. We conjecture that if we use our hypothesis classes in the distribution-dependent settings above, we would improve the factors in the generalization bounds to $nK + m\log K$.

5.3 Problem setting

In this section, we describe the problem setting. Then we briefly review a standard method of deriving generalization bounds via the Rademacher complexity arguments.

Now we fix a true rating matrix $\mathrm{X} \in \mathbb{R}^{m\times n}$ to be learned and a probability distribution $\mathcal{D}$ over $[m]\times[n]$, which is unknown to the learner. Note that the matrix X can be regarded as a function $(i,j) \mapsto X_{i,j}$. So we consider the problem under the standard PAC learning framework. Let $\mathcal{X} \subseteq \mathbb{R}^{m\times n}$ be a set of matrices called the *hypothesis class*. The input to the learner is a sample of X, i.e., a sequence of triples

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

where each $(i_t, j_t) \in [m]\times[n]$ is independently chosen according to the distribution $\mathcal{D}$. When given the sample $\mathcal{S}$, the learner is required to produce a hypothesis matrix $\hat{\mathrm{X}}$ chosen from $\mathcal{X}$, so that its generalization error is as small as possible (with high probability over the random choice of $\mathcal{S}$), where the generalization error of $\hat{\mathrm{X}}$ is defined as

$$L(\hat{\mathrm{X}}; \mathrm{X}) = \mathbb{E}_{(i,j)\sim\mathcal{D}}[\ell(\hat{X}_{i,j}; X_{i,j})]$$

for a fixed loss function $\ell : \mathbb{R}\times\mathbb{R} \to \mathbb{R}$. Throughout this chapter, we assume that the function ℓ is C_L-Lipschitz with respect to the first argument and $|\ell|$ is bounded by a constant.

A typical strategy for achieving the goal is the empirical risk minimization, i.e., to output a matrix $\hat{\mathrm{X}}$ in $\mathcal{X}$ that minimizes the empirical error, defined as

$$\hat{L}_{\mathcal{S}}(\hat{\mathrm{X}};\mathrm{X}) = \frac{1}{T}\sum_{t=1}^{T}\ell(\hat{X}_{i_t,j_t}, X_{i_t,j_t}).$$

A *generalization bound* of the hypothesis class $\mathcal{X}$ with respect to a sample size T and a confidence parameter δ is a real number ξ such that for any distribution $\mathcal{D}$, it holds that

$$\Pr_{\mathcal{S}\sim\mathcal{D}^T}\left(\forall\hat{\mathrm{X}}\in\mathcal{X}, L(\hat{\mathrm{X}};\mathrm{X}) \le \hat{L}_{\mathcal{S}}(\hat{\mathrm{X}};\mathrm{X}) + \xi\right) \ge 1-\delta.$$

So, if we have a generalization bound ξ that converges 0 as T goes to infinity, then the inequality above justifies to use the empirical risk minimization.

Now we show a general theorem that gives a generalization bound of $\mathcal{X}$ in terms of the Rademacher complexity. The (empirical) Rademacher complexity of $\mathcal{X}$ with respect to the sample $\mathcal{S}$ is defined as

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) = \frac{1}{T}\mathbb{E}_{\sigma}\left[\sup_{\hat{\mathrm{X}}\in\mathcal{X}}\sum_{t=1}^{T}\sigma_t\hat{X}_{i_t,j_t}\right],$$

where σ_t's are independent random variables taking value 1 or -1 with probability $1/2$.

Using the Rademacher complexity, we can bound the generalization error. To show this fact, we need the following theorem and lemma.

Theorem 21 (Theorem 3.1 of [57])**.** *Let $\mathcal{G} = \{g : \mathcal{Z} \to [-B, B]\}$ for some space $\mathcal{Z}$. Let $\mathcal{S} \in \mathcal{Z}^T$ is drawn independently according to unknown distribution $\mathcal{D}$. With probability at least $1-\delta$ over the random choice of $\mathcal{S}$, it holds that for any $g \in \mathcal{G}$,*

$$\mathbb{E}_{z\sim\mathcal{D}}[g(z)] \le \frac{1}{T}\sum_{t=1}^{T}g(z_t) + 2\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G}) + 3\sqrt{\frac{\ln(2/\delta)}{2T}}.$$

Proof. For any sample $\mathcal{S} = (z_1, z_2, \ldots, z_T)$, we define function Φ as

$$\Phi(\mathcal{S}) = \sup_{g\in\mathcal{G}}\left(\mathbb{E}_{z\sim\mathcal{D}}[g] - \frac{1}{T}\sum_{t=1}^{T}g[z_t]\right).$$

Let $\mathcal{S} = (z_1, z_2, \ldots, z_T)$ and $\mathcal{S}' = (z_1, z_2, \ldots, z'_T)$ be two samples differs only one instance

$z_T \neq z'_T$. Then we have

$$\Phi(\mathcal{S}) - \Phi(\mathcal{S}') \leq \sup_{g \in \mathcal{G}} \frac{1}{T} \left(\sum_{z_t \in \mathcal{S}} g(z_t) - \sum_{z_t \in \mathcal{S}'} g(z_t) \right) = \sup_{g \in \mathcal{G}} \frac{g(z_T) - g(z'_T)}{T} \leq \frac{2B}{T}$$

As same argument, we have $\Phi(\mathcal{S}') - \Phi(\mathcal{S}) \leq 2B/T$ and thus $|\Phi(\mathcal{S}) - \Phi(\mathcal{S}')| \leq 2B/T$ Using McDirmid's inequality, for any $\delta > 0$,

$$\Phi(\mathcal{S}) \leq \mathbb{E}_{\mathcal{S}}[\Phi(\mathcal{S})] + B\sqrt{\frac{\log 2/\delta}{2T}}$$

holds with probability at least $1 - \delta/2$.

Next we bound $\mathbb{E}_{\mathcal{S}}[\Phi(\mathcal{S})]$ as

$$\begin{aligned}
\mathbb{E}_{\mathcal{S}}[\Phi(\mathcal{S})] &= \mathbb{E}_{\mathcal{S}}[\sup_{g \in \mathcal{G}}(\mathbb{E}_{z \sim \mathcal{D}}[g] - \frac{1}{T}\sum_{z_t \in \mathcal{S}} g(z_t))] \\
&= \mathbb{E}_{\mathcal{S}}[\sup_{g \in \mathcal{G}}(\mathbb{E}_{\mathcal{S}'}[\frac{1}{T}\sum_{z_t \in \mathcal{S}'} g(z_t)] - \frac{1}{T}\sum_{z_t \in \mathcal{S}} g(z_t))] \\
&\leq \mathbb{E}_{\mathcal{S},\mathcal{S}'}[\sup_{g \in \mathcal{G}} \frac{1}{T}(\sum_{z'_t \in \mathcal{S}'} g(z'_t) - \sum_{z_t \in \mathcal{S}} g(z_t))] \\
&= \mathbb{E}_{\mathcal{S},\mathcal{S}'}[\sup_{g \in \mathcal{G}} \frac{1}{T}\sum_{t=1}^{T}(g(z'_t) - g(z_t))] \\
&= \mathbb{E}_{\sigma,\mathcal{S},\mathcal{S}'}[\sup_{g \in \mathcal{G}} \frac{1}{T}\sum_{t=1}^{T}\sigma_t(g(z'_t) - g(z_t))] \\
&\leq \mathbb{E}_{\sigma,\mathcal{S}'}[\sup_{g \in \mathcal{G}} \frac{1}{T}\sum_{t=1}^{T}\sigma_t g(z'_t)] + \mathbb{E}_{\sigma,\mathcal{S}}[\sup_{g \in \mathcal{G}} \frac{1}{T}\sum_{t=1}^{T}\sigma_t g(z_t)] \\
&= 2\mathbb{E}_{\sigma,\mathcal{S}}[\sup_{g \in \mathcal{G}} \frac{1}{T}\sum_{t=1}^{T}\sigma_t g(z_t)] = 2\mathbb{E}[\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})]
\end{aligned}$$

Finally we reduce the bound by $\mathbb{E}[\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})]$ to $\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})$. By the definition of the Rademacher complexity, changing one instance $z_t \in \mathcal{S}$ to z'_t changes $\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})$ by at most B/T. Using McDirmid's inequality again, we have

$$\mathbb{E}[\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})] \leq \hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G}) + B\sqrt{\frac{\log 2/\delta}{2T}}$$

with probability $1 - \delta/2$. Combining this inequality and $\Phi(\mathcal{S}) \leq 2\mathbb{E}[\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G})] + B\sqrt{\frac{\log 2/\delta}{2T}}$ by

using the the union bound, we get the theorem. □

Theorem 22 (Lemma 4.2 of [57]). *Let $\Phi : \mathbb{R} \to \mathbb{R}$ be C_L-Lipschitz function. Then for any class $\mathcal{G} : \mathcal{Z} \to \mathbb{R}$, the following holds;*

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\Phi \circ \mathcal{G}) \leq C_L \hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{G}).$$

Proof. Let $\mathcal{S} = (z_1, \ldots, z_T) \in \mathcal{Z}^{\mathsf{T}}$ be a fixed sample. By the definition of $\hat{\mathfrak{R}}_{\mathcal{S}}(\Phi \circ \mathcal{G})$,

$$\begin{aligned}
\hat{\mathfrak{R}}_{\mathcal{S}}(\Phi \circ \mathcal{G}) &= \frac{1}{T}\mathbb{E}_{\sigma}[\sup_{g \in \mathcal{G}} \sum_{t=1}^{T} \sigma_t (\Phi \circ g)(z_t)] \\
&= \frac{1}{T}\mathbb{E}_{\sigma_1, \sigma_2, \ldots, \sigma_{T-1}}[\mathbb{E}_{\sigma_T}[\sup_{g \in \mathcal{G}} u_{T-1}(g) + \sigma_T (\Phi \circ g)(z_T)]],
\end{aligned}$$

where $u_{T-1}(g) = \sum_{t=1}^{T-1} \sigma_t (\Phi \circ g)(z_t)$. By the definition of the supremum, for any $\epsilon > 0$, there exists $g_1, g_2 \in \mathcal{G}$ such that

$$\begin{aligned}
u_{T-1}(g_1) + (\Phi \circ g_1)(z_T) &\geq (1-\epsilon)[\sup_{g \in \mathcal{G}} u_{T-1}(g) + (\Phi \circ g)(z_T)] \\
u_{T-1}(g_2) - (\Phi \circ g_1)(z_T) &\geq (1-\epsilon)[\sup_{g \in \mathcal{G}} u_{T-1}(g) - (\Phi \circ g)(z_T)]
\end{aligned}$$

Thus for any $\epsilon > 0$, we have

$$\begin{aligned}
&(1-\epsilon)\mathbb{E}_{\sigma_T}[\sup_{g \in \mathcal{G}} u_{T-1}(g) + \sigma_T (\Phi \circ g)(z_T)] \\
&= (1-\epsilon)\frac{1}{2}(\sup_{g \in \mathcal{G}} u_{T-1}(g) + (\Phi \circ g)(z_T) + \sup_{g \in \mathcal{G}} u_{T-1}(g) - (\Phi \circ g)(z_T)) \\
&\leq \frac{1}{2}(u_{T-1}(g_1) + (\Phi \circ g_1)(z_T) + u_{T-1}(g_2) - (\Phi \circ g_2)(z_T)).
\end{aligned}$$

Let $s = \operatorname{sgn}(g_1(z_T) - g_2(z_T))$ then this inequality implies

$$\begin{aligned}
&(1-\epsilon)\mathbb{E}_{\sigma_T}[\sup_{g \in \mathcal{G}} u_{T-1}(g) + \sigma_T (\Phi \circ g)(z_T)] \\
&\leq \frac{1}{2}(u_{T-1}(g_1) + u_{T-1}(g_2) - sC_L(g_1(z_T) - g_2(z_T))) \\
&= \frac{1}{2}(u_{T-1}(g_1) + sC_L g_1(z_T)) + \frac{1}{2}(u_{T-1}(g_2) - sC_L g_2(z_T)) \\
&\leq \frac{1}{2}\sup_{g \in \mathcal{G}}(u_{T-1}(g) + sC_L g(z_T)) + \frac{1}{2}\sup_{g \in \mathcal{G}}(u_{T-1}(g) - sC_L g(z_T)) \\
&= \mathbb{E}_{\sigma_T}[\sup_{g \in \mathcal{G}}(u_{T-1}(g) - \sigma C_L g(z_T))]
\end{aligned}$$

This inequality holds for any $\epsilon > 0$ thus we have

$$\mathbb{E}_{\sigma_T}[\sup_{g\in\mathcal{G}} u_{T-1}(g) + \sigma_T(\Phi \circ g)(z_T)] \leq \mathbb{E}_{\sigma_T}[\sup_{g\in\mathcal{G}} u_{T-1}(g) + \sigma_T C_L g(z_T)].$$

Using same argument for other σ_t for $t \in [T-1]$, we get the lemma. □

The following theorem is a combination of Theorem 3.1 of [57] and Lemma 4.2 of [57].

Theorem 23 (Generalization bounds [57])**.** *With probability at least* $1-\delta$ *over the random choice of* $\mathcal{S} \in ([m] \times [n] \times \mathbb{R})^T$*, it holds that for any* $\hat{\mathrm{X}} \in \mathcal{X}$*,*

$$L(\hat{\mathrm{X}}; \mathrm{X}) \leq \hat{L}_{\mathcal{S}}(\hat{\mathrm{X}}; \mathrm{X}) + 2C_L \hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) + O\left(\sqrt{\frac{\ln(1/\delta)}{T}}\right).$$

Thus, to derive a generalization bound from this theorem, we need to estimate the Rademacher complexity $\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X})$.

5.4 Generalization bounds for our hypothesis classes

In this section, we give upper bounds of the Rademacher complexity of our hypothesis classes, whereby we give generalization bounds of these classes.

5.4.1 Matrix factorization with convex combination constraints

Here we assume that the preferences of each user can be expressed as the convex combination of a small number K of model users' preferences. Let $\mathcal{P} \subseteq \mathbb{R}^n$ be a finite set of vectors, so that its convex hull $\mathrm{conv}(\mathcal{P})$ defines the class of model user's preferences. In particular, if $\mathcal{P} = \{-B, B\}^n$, then $\mathrm{conv}(\mathcal{P}) = [-B, B]^n$. For a set of vectors $\mathcal{Y} \subseteq \mathbb{R}^n$, we define $\mathcal{Y}^K$ as the set of all matrices in $\mathbb{R}^{n\times K}$ whose columns are chosen from $\mathcal{Y}$. That is, $\mathcal{Y}^K = \{(\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_K) : \forall k \in [K], \boldsymbol{y}_k \in \mathcal{Y}\}$. Let $\mathcal{U}$ and $\mathcal{V}$ be the sets of matrices defined as

$$\mathcal{U} = \left\{\mathrm{U} \in [0,1]^{m\times K} : \forall i \in [m], \|\mathrm{U}_{i,*}\|_1 = 1\right\}, \tag{5.1}$$

$$\mathcal{V} = (\mathrm{conv}(\mathcal{P}))^K. \tag{5.2}$$

Then, first we examine the following hypothesis class:

$$\mathcal{X} = \left\{\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^\mathsf{T} \in \mathbb{R}^{m\times n} : \mathrm{U} \in \mathcal{U},\ \mathrm{V} \in \mathcal{V}\right\}. \tag{5.3}$$

In order to derive an upper bound of the Rademacher complexity of $\mathcal{X}$, we need some technical lemmas.

Lemma 15. *Let $\mathcal{F} \subseteq \{f : \mathcal{Y} \to \mathcal{Z}\}$ and $\mathcal{G} \subseteq \{g : \mathcal{W} \to \mathcal{Y}\}$ be finite sets of functions. Then* $\mathrm{conv}(\mathcal{F}) \circ \mathcal{G} \subseteq \mathrm{conv}(\mathcal{F} \circ \mathcal{G})$.

Proof. By the definition of the convex hull, we can write

$$\mathrm{conv}(\mathcal{F}) = \left\{ \sum_{f \in \mathcal{F}} a_f f(\cdot) : \boldsymbol{a} \in \Delta_{\mathcal{F}} \right\},$$

where $\Delta_{\mathcal{F}}$ denotes the probability simplex over $\mathcal{F}$. Therefore, for any $h \in \mathrm{conv}(\mathcal{F}) \circ \mathcal{G}$, there exist some $\boldsymbol{a} \in \Delta_{\mathcal{F}}$ and $g \in \mathcal{G}$ such that $h = (\sum_{f \in \mathcal{F}} a_f f) \circ g$. Then, for any $\boldsymbol{w} \in \mathcal{W}$, $h(\boldsymbol{w}) = \sum_{f \in \mathcal{F}} a_f f(g(\boldsymbol{w})) = \sum_{f \in \mathcal{F}} a_f (f \circ g)(\boldsymbol{w})$. This implies that $h \in \mathrm{conv}(\mathcal{F} \circ \mathcal{G})$. □

The following lemma is crucial in our analysis.

Lemma 16. *Let $\mathcal{P}$ be a finite set of vectors. Then* $(\mathrm{conv}(\mathcal{P}))^K \subseteq \mathrm{conv}(\mathcal{P}^K)$.

Proof. Let $\mathrm{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_K) \in (\mathrm{conv}(\mathcal{P}))^K$. Then we can write $\boldsymbol{v}_i = \sum_{\boldsymbol{p} \in \mathcal{P}} a_{i,\boldsymbol{p}} \boldsymbol{p}$ for some $\boldsymbol{a}_i \in \Delta_{\mathcal{P}}$. Let $b_{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K} = \prod_{i=1}^K a_{i,\boldsymbol{p}_i}$. Then observe that

$$\begin{aligned}
\sum_{(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K) \in \mathcal{P}^K} b_{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K} &= \sum_{(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K) \in \mathcal{P}^K} \prod_{i=1}^K a_{i,\boldsymbol{p}_i} = \sum_{\boldsymbol{p}_1 \in \mathcal{P}} \sum_{\boldsymbol{p}_2 \in \mathcal{P}} \cdots \sum_{\boldsymbol{p}_K \in \mathcal{P}} \prod_{i=1}^K a_{i,\boldsymbol{p}_i} \\
&= \Big(\sum_{\boldsymbol{p}_1 \in \mathcal{P}} a_{1,\boldsymbol{p}_1} \Big) \cdots \Big(\sum_{\boldsymbol{p}_K \in \mathcal{P}} a_{K,\boldsymbol{p}_K} \Big) = 1.
\end{aligned}$$

Thus $\{b_{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K} : (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_K) \in \mathcal{P}^K\}$ defines a probability distribution over $\mathcal{P}^K$.

The K-th column $\boldsymbol{v}_K$ can be represented as follows:

$$\begin{aligned}
\boldsymbol{v}_K &= \sum_{\boldsymbol{p} \in \mathcal{P}} a_{K,\boldsymbol{p}} \boldsymbol{p} \\
&= \Big(\sum_{\boldsymbol{p}_1 \in \mathcal{P}} a_{1,\boldsymbol{p}_1} \Big) \times \Big(\sum_{\boldsymbol{p}_2 \in \mathcal{P}} a_{2,\boldsymbol{p}_2} \Big) \times \cdots \times \Big(\sum_{\boldsymbol{p}_{K-1} \in \mathcal{P}} a_{K-1,\boldsymbol{p}_{K-1}} \Big) \sum_{\boldsymbol{p}_K \in \mathcal{P}} a_{K,\boldsymbol{p}_K} \boldsymbol{p}_K \\
&= \left(\sum_{(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_{K-1}) \in \mathcal{P}^{K-1}} \prod_{i=1}^{K-1} a_{i,\boldsymbol{p}_i} \right) \sum_{\boldsymbol{p}_K \in \mathcal{P}} a_{K,\boldsymbol{p}_K} \boldsymbol{p}_K \\
&= \sum_{\boldsymbol{p}_K \in \mathcal{P}} \sum_{(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_{K-1}) \in \mathcal{P}^{K-1}} \prod_{i=1}^{K} a_{i,\boldsymbol{p}_i} \boldsymbol{p}_K = \sum_{(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K) \in \mathcal{P}^K} b_{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K} \boldsymbol{p}_K.
\end{aligned}$$

One can easily get the same results for other columns $\boldsymbol{v}_j$ s.t. $j \in [K-1]$. Note that $b_{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K}$ depends only on $\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K$ and not on j. This implies for any $\mathrm{V} \in (\mathrm{conv}(P))^K$,

$$\mathrm{V} = \sum_{(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K)\in\mathcal{P}^K} b_{\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K}(\boldsymbol{p}_1,\ldots,\boldsymbol{p}_K) \in \mathrm{conv}(\mathcal{P}^K).$$

□

In 5.9, we show an alternative "algorithmic" proof for Lemma 16. Now we are ready to prove one of our main results.

Theorem 24 (Main theorem of Chapter 5)**.** *Let $\mathcal{P} \subseteq [-B,B]^n$ be a finite set of vectors and $\mathcal{X}$ be the hypothesis class defined as in (5.1), (5.2) and (5.3). Then,*

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) \leq B\sqrt{8\frac{K\ln|\mathcal{P}| + m\ln K}{T}}.$$

Proof. Any $\hat{\mathrm{X}} \in \mathcal{X}$ can be decomposed as $\hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^\mathsf{T}$ and hence $\hat{X}_{i_t,j_t} = \mathrm{U}_{i_t,*}\cdot\mathrm{V}_{j_t,*}$. So,

$$\begin{aligned} T\,\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) &= \mathbb{E}_\sigma\left[\sup_{\mathrm{U}\in\mathcal{U},\mathrm{V}\in\mathcal{V}}\sum_{t=1}^{T}\sigma_t\mathrm{U}_{i_t,*}\cdot\mathrm{V}_{j_t,*}\right] \\ &= \mathbb{E}_\sigma\left[\sup_{\mathrm{U}\in\mathcal{U},\mathrm{V}\in\mathcal{V}}\sum_{i=1}^{N}\mathrm{U}_{i,*}\cdot\left(\sum_{t:i_t=i}\sigma_t\mathrm{V}_{j_t,*}\right)\right] \\ &= \mathbb{E}_\sigma\left[\sup_{\mathrm{V}\in\mathcal{V}}\sum_{i=1}^{m}\max_{k\in[K]}\left(\sum_{t:i_t=i}\sigma_t V_{j_t,k}\right)\right]. \end{aligned}$$

The last equation is from the fact that $\mathrm{U}_{i,*}$ is a probability distribution. One can regard the operator $\max_{k\in[K]}$ inside the sum over all i as a mapping from $i \in [m]$ to $k \in [K]$. Let $\mathcal{K} = \{\kappa : [m] \to [K]\}$ be the set of all mappings from $[m]$ to $[K]$. So, the formula above can be rewritten as

$$\begin{aligned} \mathbb{E}_\sigma\left[\sup_{\mathrm{V}\in\mathcal{V}}\max_{\kappa\in\mathcal{K}}\sum_{i=1}^{m}\sum_{t:i_t=i}\sigma_t V_{j_t,\kappa(i)}\right] &= \mathbb{E}_\sigma\left[\sup_{\mathrm{V}\in\mathcal{V}}\max_{\kappa\in\mathcal{K}}\sum_{t=1}^{T}\sigma_t V_{j_t,\kappa(i_t)}\right] \\ &= \mathbb{E}_\sigma\left[\sup_{\mathrm{V}\circ\kappa\in\mathcal{V}\circ\mathcal{K}}\sum_{t=1}^{T}\sigma_t V_{j_t,\kappa(i_t)}\right] = T\,\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{V}\circ\mathcal{K}). \end{aligned}$$

Thus, we have $\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) \leq \hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{V}\circ\mathcal{K})$. Since $\mathcal{V} = (\mathrm{conv}(\mathcal{P}))^K$, Lemmas 15 and 16 imply that $\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{V}\circ\mathcal{K}) \leq \hat{\mathfrak{R}}_{\mathcal{S}}(\mathrm{conv}(\mathcal{P}^K\circ\mathcal{K}))$, where $\mathcal{P}^K\circ\mathcal{K}$ is a finite set. Using the well known property

of the Rademacher complexity of the convex hull of a finite set (See, e.g., Theorem 3.3 of [57]), we get

$$\begin{aligned}\hat{\mathfrak{R}}_{\mathcal{S}}(\mathrm{conv}(\mathcal{P}^K \circ \mathcal{K})) &\leq B\sqrt{\frac{8\ln|\mathcal{P}^K \circ \mathcal{K}|}{T}} \\ &= B\sqrt{\frac{8(K\ln|\mathcal{P}| + \ln|\mathcal{K}|)}{T}}. \end{aligned} \tag{5.4}$$

Using the fact $|\mathcal{K}| = K^m$, we obtain the theorem. □

Now we choose $\mathcal{P} = \{-B, B\}^n$ so that $\mathrm{conv}(\mathcal{P}) = [-B, B]^n$, i.e., the n-dimensional L_∞-norm ball. Clearly, $|\mathcal{P}| = 2^n$. Combining Theorem 23 and Theorem 24, we get the following generalization bound for the class $\mathcal{X}$ induced by $\mathcal{P}$.

Corollary 25. *Let $\mathcal{P} = \{-B, B\}^n$ and $\mathcal{X}$ be defined as in (5.1), (5.2) and (5.3). Note here that $V = [-B, B]^{n\times K}$. Then*

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) \leq B\sqrt{8\frac{nK\ln 2 + m\ln K}{T}},$$

and an generalization bound of $\mathcal{X}$ is

$$O\left(C_L B\sqrt{\frac{nK + m\ln K}{T}} + \sqrt{\frac{\ln(1/\delta)}{T}}\right).$$

Another choice of $\mathcal{P}$ may be useful in some applications and lead an even smaller generalization bound. For example, if we take $\mathcal{P} = \{-\boldsymbol{e}_1, \boldsymbol{e}_1, -\boldsymbol{e}_2, \boldsymbol{e}_2, \ldots, -\boldsymbol{e}_n, \boldsymbol{e}_n\}$ where each $\boldsymbol{e}_i \in \{0, 1\}^n$ is the unit vector whose i-th element is 1, then we have $\mathrm{conv}(\mathcal{P}) = \{\boldsymbol{v} \in \mathbb{R}^n : \|\boldsymbol{v}\|_1 \leq 1\}$, which is the n-dimensional L_1-norm ball. In this case $|\mathcal{P}| = 2n$. Therefore, we have the following corollary.

Corollary 26. *Let $\mathcal{P} = \{-\boldsymbol{e}_1, \boldsymbol{e}_1, -\boldsymbol{e}_2, \boldsymbol{e}_2, \ldots, -\boldsymbol{e}_n, \boldsymbol{e}_n\}$ and $\mathcal{X}$ be defined as in (5.1), (5.2) and (5.3). Then, a generalization bound of $\mathcal{X}$ is*

$$O\left(C_L B\sqrt{\frac{K\ln n + m\ln K}{T}} + \sqrt{\frac{\ln(1/\delta)}{T}}\right).$$

5.4.2 Matrix factorization with L_1 and L_∞ norm constraints

Now we consider a slightly generalized hypothesis class. For a finite set $\tilde{\mathcal{P}} \in \mathbb{R}^n$, let

$$\mathcal{U} = \left\{ \mathrm{U} \in \mathbb{R}^{m \times \tilde{K}} : \forall i \in [m], \|\mathrm{U}_{i,*}\|_1 \leq 1 \right\}, \tag{5.5}$$

$$\tilde{\mathcal{V}} = \left(\mathrm{conv}(\tilde{\mathcal{P}}) \right)^{\tilde{K}}, \tag{5.6}$$

and

$$\tilde{\mathcal{X}} = \left\{ \hat{\mathrm{X}} = \mathrm{U}\mathrm{V}^\mathsf{T} \in \mathbb{R}^{m \times n} : \mathrm{U} \in \tilde{\mathcal{U}},\ \mathrm{V} \in \tilde{\mathcal{V}} \right\}. \tag{5.7}$$

Only the difference from the class $\mathcal{X}$ discussed in the previous subsection is that U in $\tilde{\mathcal{U}}$ can contain negative components and the L_1-norm of each row $\mathrm{U}_{i,*}$ is not necessarily 1.

The next theorem says that the problem of learning with the hypothesis class $\tilde{\mathcal{X}}$ can be reduced to that with the hypothesis class $\mathcal{X}$.

Theorem 27. *$\tilde{\mathcal{X}} \subseteq \mathcal{X}$, where $\mathcal{X}$ is defined as in (5.1), (5.2) and (5.3) with $K = 2\tilde{K} + 1$ and $\mathcal{P} = \tilde{\mathcal{P}} \cup \{-\boldsymbol{a} : \boldsymbol{a} \in \tilde{\mathcal{P}}\}$.*

Proof. It suffices to show that any $\tilde{\mathrm{X}} = \tilde{\mathrm{U}}\tilde{\mathrm{V}}^\mathsf{T} \in \tilde{\mathcal{X}}$ can be represented as $\mathrm{U}\mathrm{V}^\mathsf{T}$ for some $\mathrm{U} \in \mathcal{U}$ and $\mathrm{V} \in (\mathrm{conv}(\mathcal{P}))^K$. To see this, let $\mathrm{U}^+ \in [0,1]^{m \times \tilde{K}}$ and $\mathrm{U}^- \in [0,1]^{m \times \tilde{K}}$ be non-negative matrices such that $U^+_{i,j} = [\tilde{U}_{i,j}]_+$ and $U^-_{i,j} = [-\tilde{U}_{i,j}]_+$, respectively, where $[x]_+ = x$ if $x \geq 0$ and $[x]_+ = 0$ if $x < 0$. Moreover, let $\boldsymbol{u} \in [0,1]^m$ be a vector whose i-th component is $u_i = 1 - \|\tilde{\mathrm{U}}_{i,*}\|$. Then, it is straightforward to show that

$$\tilde{\mathrm{U}}\tilde{\mathrm{V}}^\mathsf{T} = [\mathrm{U}^+, \mathrm{U}^-, \boldsymbol{u}][\tilde{\mathrm{V}}, -\tilde{\mathrm{V}}, \mathbf{0}]^\mathsf{T}.$$

Moreover, it is easy to show that $\mathrm{U} = [\mathrm{U}^+, \mathrm{U}^-, \boldsymbol{u}] \in \mathcal{U}$ and $\mathrm{V} = [\tilde{\mathrm{V}}, -\tilde{\mathrm{V}}, \mathbf{0}] \in (\mathrm{conv}(\mathcal{P}))^K$. □

Since $\hat{\mathfrak{R}}_S(\tilde{\mathcal{X}}) \leq \hat{\mathfrak{R}}_S(\mathcal{X})$ by the theorem above, we have the same order of generalization bound of $\tilde{\mathcal{X}}$ as shown in Theorem 24.

5.5 A matching lower bound on the Rademacher complexity

We show that the bound stated in Corollary 25 is almost tight and cannot be significantly improved in the worst case.

Theorem 28. *Let $\mathcal{X}$ be the matrix class defined in Corollary 25. For any sufficiently large T, there exists a sample $\mathcal{S}$ such that*

$$\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) = \Omega\left(B\sqrt{\frac{nK + m\log K}{T}}\right).$$

Proof. Let $L = \lfloor \log_2 K \rfloor$ and $J_1 \cup J_2 \cup \cdots \cup J_L = [n]$ be a partition of the column set. Consider a sample $\mathcal{S}$ that satisfies $|\mathcal{T}(i,u)| = T/(mL)$ for each $i \in [m]$ and $u \in [L]$, and $|\{t : j_t = j\}| = T/n$ for each $j \in [n]$, where $\mathcal{T}(i,u) = \{t : i_t = i, j_t \in J_u\}$.

From the proof of Theorem 24, we have

$$\begin{aligned}
\frac{T}{B}\hat{\mathfrak{R}}_{\mathcal{S}}(\mathcal{X}) &= \frac{1}{B}\mathbb{E}\left[\sup_V \max_{\kappa\in\mathcal{K}} \sum_{t=1}^{T} \sigma_t V_{j_t,\kappa(i_t)}\right] \\
&= \frac{1}{B}\mathbb{E}\left[\max_{\kappa\in\mathcal{K}} \sup_V \sum_{j=1}^{M} \sum_{t:j_t=j} \sigma_t V_{j,\kappa(i_t)}\right] \\
&= \frac{1}{B}\mathbb{E}\left[\max_{\kappa\in\mathcal{K}} \sup_V \sum_{j=1}^{n}\sum_{\ell=1}^{K} \sum_{t:j_t=j,\kappa(i_t)=\ell} \sigma_t V_{j,\ell}\right] \\
&= \mathbb{E}\left[\max_{\kappa\in\mathcal{K}} \sum_{j=1}^{n}\sum_{\ell=1}^{K} \left|\sum_{t:j_t=j,\kappa(i_t)=\ell} \sigma_t\right|\right]. \qquad (5.8)
\end{aligned}$$

So, it suffices to show that (5.8) is lower bounded by $\Omega(\sqrt{nKT})$ and $\Omega(\sqrt{mLT})$.

First we give a proof of the $\Omega(\sqrt{nKT})$ part. To this end, we replace the max operator in (5.8) by the expectation under the uniform distribution over $\mathcal{K}$. Note then that $T_{j,\ell} = |\{t : j_t = j, \kappa(i_t) = \ell\}|$ is a random variable and $\mathbb{E}_\kappa[T_{j,\ell}] = T/(nK)$ for each $j \in [n]$ and $\ell \in [K]$, since κ is uniformly chosen. So, by Markov inequality, we have $\Pr[T_{j,\ell} \geq T/(2nK)] \geq 1/2$. Moreover, $\mathbb{E}_\sigma[|\sum_{t:j_t=j,\kappa(i_t)=\ell} \sigma_t|]) \geq c\sqrt{T_{j,\ell}}$ for some constant $c > 0$ by the random-walk argument. Therefore,

$$\begin{aligned}
\mathbb{E}\left[\max_{\kappa\in\mathcal{K}} \sum_{j=1}^{n}\sum_{\ell=1}^{K} \left|\sum_{t:j_t=j,\kappa(i_t)=\ell} \sigma_t\right|\right] &\geq \mathbb{E}\left[\mathbb{E}_\kappa\left[\sum_{j=1}^{n}\sum_{\ell=1}^{K} \left|\sum_{t:j_t=j,\kappa(i_t)=\ell} \sigma_t\right|\right]\right] \\
&\geq \frac{c}{2}\sum_{j=1}^{n}\sum_{\ell=1}^{K}\sqrt{\frac{T}{2nK}} = \Omega\left(\sqrt{nKT}\right).
\end{aligned}$$

Next we give a proof of the $\Omega(\sqrt{mLT})$ part. For each $\sigma \in \{-1,1\}^T$, we define $\kappa \in \mathcal{K}$ as

described below, and replace the max operator in (5.8) by this particular κ, which gives a lower bound. For each $i \in [m]$ and $u \in [L]$, let

$$\ell_{i,u} = \begin{cases} 1 & \text{if } \sum_{t \in \mathcal{T}(i,u)} \sigma_t \geq 0, \\ -1 & \text{otherwise} \end{cases}$$

and let the L-bit sequence $\ell_i = (\ell_{i,1}, \ell_{i,2}, \ldots, \ell_{i,L})$ be identified with an integer in $[K]$ whose binary representation is ℓ_i. For example, $(-1,-1,-1) = 1$, $(-1,-1,1) = 2$, …, $(1,1,1) = 8$ when $L = 3$. Now, our κ is given by $\kappa(i) = \ell_i$ for each i. Then, (5.8) is lower bounded by

$$\mathbb{E}\left[\sum_{j=1}^{n}\sum_{\ell=1}^{K}\left|\sum_{t:j_t=j,\ell_{i_t}=\ell}\sigma_t\right|\right].$$

Moreover,

$$\begin{aligned}
\sum_{j=1}^{n}\sum_{\ell=1}^{K}\left|\sum_{t:j_t=j,\ell_{i_t}=\ell}\sigma_t\right| &= \sum_{u=1}^{L}\sum_{\ell=1}^{K}\sum_{j\in J_u}\left|\sum_{t:j_t=j,\ell_{i_t}=\ell}\sigma_t\right| \\
&\geq \sum_{u=1}^{L}\sum_{\ell=1}^{K}\left|\sum_{j\in J_u}\sum_{t:j_t=j,\ell_{i_t}=\ell}\sigma_t\right| \\
&= \sum_{u=1}^{L}\sum_{\ell=1}^{K}\left|\sum_{i\in[n]}\sum_{j\in J_u}\sum_{t:i_t=i,j_t=j}\mathbf{1}[\ell_i=\ell]\sigma_t\right| \\
&= \sum_{u=1}^{L}\sum_{\ell=1}^{K}\left|\sum_{i:\ell_i=\ell}\sum_{t\in\mathcal{T}(i,u)}\sigma_t\right|.
\end{aligned}$$

Now by the definition of $\ell_{i,u}$, we have

$$\sum_{t\in\mathcal{T}(i,u)}\sigma_t = \ell_{i,u}\left|\sum_{t\in\mathcal{T}(i,u)}\sigma_t\right|.$$

Therefore, we get

$$\begin{aligned}\mathbb{E}\left[\sum_{j=1}^{n}\sum_{\ell=1}^{K}\left|\sum_{t:j_t=j,\ell_{i_t}=\ell}\sigma_t\right|\right] &\geq \mathbb{E}\left[\sum_{u=1}^{L}\sum_{\ell=1}^{K}\sum_{i:\ell_i=\ell}\left|\sum_{t\in\mathcal{T}(i,u)}\sigma_t\right|\right] \\ &= \sum_{u=1}^{L}\sum_{i\in[n]}\mathbb{E}\left[\left|\sum_{t\in\mathcal{T}(i,u)}\sigma_t\right|\right] \\ &\geq \sum_{u=1}^{L}\sum_{i\in[n]} c\sqrt{|\mathcal{T}(i,u)|} = c\sqrt{nLT},\end{aligned}$$

where the last inequality is derived from the random-walk argument. □

5.6 Experimental results

We conduct two kind of experiments using three data sets. First data set is the MovieLens-100k data set [35] [1] which contains 100000 ratings of 1682 movies by 943 users. All ratings are valued in $\{1, 2, 3, 4, 5\}$. Second one is the sushi preference data set [45] [2] which contains 50000 ratings of 100 kind of sushi items by 5000 users. All ratings are valued in $\{0, 1, 2, 3, 4\}$. The last data is a synthetic data. We made this data as follows; (i) generate $\mathrm{U}_{i,k}$ and $\mathrm{V}_{j,k}$ according to the uniform distribution over $[0, 1)$, (ii) generate a random noise matrix $\mathrm{R}_{i,j}$ according to the standard normal distribution and (iii) then $\mathrm{X} = \alpha\mathrm{U}\mathrm{V}^\intercal + (1-\alpha)\mathrm{R}$ and use randomly chosen 50% entries of X as the sample and the test set. We adopt $m = 200, n = 600, K = 10, \alpha = 0.75$. This data contains 59736 entries with range $[-0.347, 4.569]$. The mean value of $\mathrm{X}_{i,j}$ is 1.886 and its variance is 0.349.

We adopt the squared loss $\ell(x, y) = \frac{1}{2}(x - y)^2$ in our experiments. We conduct 5-fold cross validation, and then measure the mean squared error,

$$\mathrm{MSE}(\hat{\mathrm{X}}, \mathcal{T}) = \frac{1}{|\mathcal{T}|}\sum_{(i,j,\mathrm{X}_{i,j})\in\mathcal{T}} \ell(\hat{\mathrm{X}}_{i,j}, \mathrm{X}_{i,j})$$

where $\mathcal{T}$ is the test set.

[1] http://grouplens.org/datasets/movielens/
[2] http://www.kamishima.net/sushi/

5.6.1 Changing L_∞ norm constraint

In first experiment, we change the value of L_∞ norm constraint of our hypothesis class defined in (5.1),(5.2) and (5.3), namely B. We seek a local minimizer $\hat{\mathrm{X}}$ of the empirical loss $\hat{\ell}_{\mathcal{S}}$ from the hypothesis class and then measure the training error $\mathrm{MSE}(\hat{\mathrm{X}}, \mathcal{S})$ and test error $\mathrm{MSE}(\hat{\mathrm{X}}, \mathcal{T})$. We use the alternating least square optimization scheme to get a local minimizer (See 5.8 for implementation details). For all data sets, we choose the rank $K = 10$ in experiment.

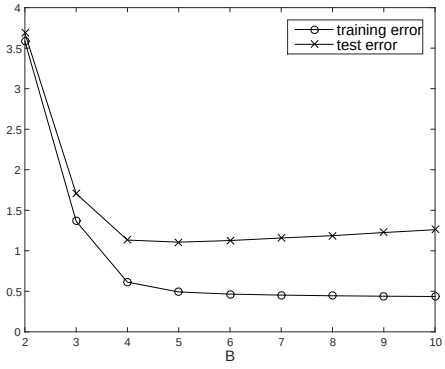

Figure 5.1: MovieLens 100k data set

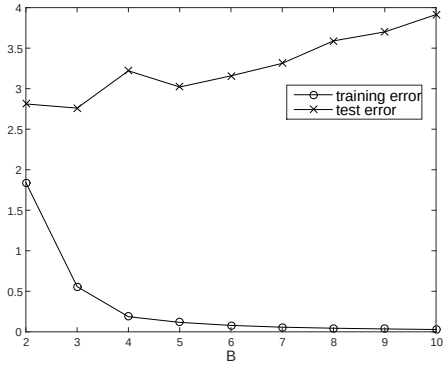

Figure 5.2: sushi3b data set

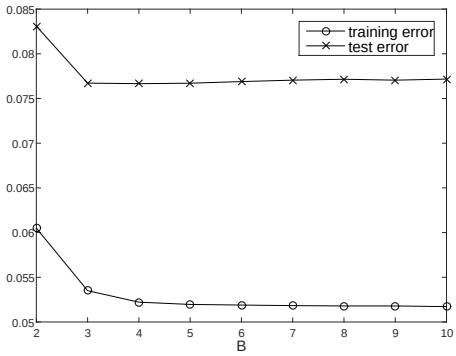

Figure 5.3: synthetic data

Note that our hypothesis class with large B behaves like the class of matrices with the low rank constraint only.

The results of training error and test error are shown in Figure 5.1, Figure 5.2 and Figure 5.3. One can see that, in both data sets, the training error is small enough when B reaches its original range of rating values. This implies bounding B with some reasonable value does not increase the upper bound of the generalization error.

5.6.2 Comparison with trace norm regularization

In the second experiments, we compare our norm-constrained matrix factorization with conventional trace norm regularized empirical risk minimization.

We seek a local minimizer of the empirical loss $\hat{\ell}_{\mathcal{S}}$ from the hypothesis class defined as (5.1), (5.2) and (5.3). We choose the best rank of our hypothesis K by grid searching from $\{1, 2, 4, 8, 10\}$ for MovieLens and sushi dataset, $\{4, 8, 11, 12, 14\}$ for synthetic data set.

We compare our hypothesis class with the typical trace norm regularization approach, which minimizes the empirical loss and its trace norm as the regularization term as follows;

$$\hat{\mathrm{X}}_{\mathrm{Tr}} = \arg\min_{\hat{\mathrm{X}} \in \mathbb{R}^{m \times n}} \frac{1}{2} \sum_{(i,j) \in \mathcal{S}} (\mathrm{X}_{i,j} - \hat{\mathrm{X}}_{i,j})^2 + \lambda \|\hat{\mathrm{X}}\|_{\mathrm{Tr}}$$

where λ is a parameter. We also choose the best $\lambda \in \{0.1, 0.2, 0.4, 0.8, 1, 2, 4, 6, 8, 10, 20, 40\}$ by grid searching. It is well known that this is the tightest convex relaxation of the rank constraint. To solve this minimization problem, we use SoftImpute implemented at fancyimpute 0.0.19 [3]. This is an implementation of SoftImpute algorithm proposed by Mazumder et al. [56].

Result of experiments are summarized in Table 5.1.

Table 5.1: MSE of matrix completion results

Data set	Our approach	SoftImpute
MovieLens	0.881 ($K = 2, B = 5$)	1.011 ($\lambda = 8.0$)
sushi	1.363 ($K = 1, B = 4$)	1.388 ($\lambda = 8.0$)
synthetic data	0.079 ($K = 11, B = 4.569$)	0.399 ($\lambda = 4.0$)

One can see that the empirical risk minimization with our hypothesis class performs better than trace norm regularization.

In both data sets, the rank of the local minimizer $\hat{\mathrm{X}}$ is very small. Especially, on the sushi data set, it is 1. This indicates that all users have almost the same preference over items.

5.7 Conclusion

In this chapter, we focus on the collaborative filtering problem and derive the generalization error bounds for matrix factorization with L_1 and L_∞ norm constraints.

[3] https://pypi.python.org/pypi/fancyimpute

One of our future work is, to derive generalization bounds with other norm constraints such as the Frobenius norm. Also, one could improve our results for strongly convex loss functions or some easy data setting such as the Bernstein conditions.

5.8 Implementation details

In this section we describe an algorithm to find a hypothesis from the hypothesis class. We adopt the empirical risk minimization which selects the hypothesis $\hat{\mathrm{X}}$ as

$$\hat{\mathrm{X}} = \arg\min_{\hat{\mathrm{X}} \in \mathcal{X}} \hat{L}_{\mathcal{S}}(\hat{X}_{i,j}, X_{i,j}) = \arg\min_{\hat{\mathrm{U}} \in \mathcal{U}, \hat{\mathrm{V}} \in \mathcal{V}} \hat{L}_{\mathcal{S}}(\hat{\mathrm{U}}\hat{\mathrm{V}}^{\mathsf{T}}; \mathrm{X})$$

where $\mathcal{X}$ is the hypothesis class.

In the implementation of experiments, we use the conventional alternating minimization which iterates

$$\hat{\mathrm{U}} = \arg\min_{\mathrm{U} \in \mathcal{U}} \hat{L}_{\mathcal{S}}(\mathrm{U}\hat{\mathrm{V}}^{\mathsf{T}}; \mathrm{X}) \text{ and } \hat{\mathrm{V}} = \arg\min_{\mathrm{V} \in \mathcal{V}} \hat{L}_{\mathcal{S}}(\hat{\mathrm{U}}\mathrm{V}^{\mathsf{T}}; \mathrm{X})$$

until it converges. Each optimization step is just a convex optimization. Especially, if we choose ℓ as squared loss, it is a quadratic programming. So we can solve optimization of U and V efficiently using common QP solvers.

Of course our optimization of $\hat{\mathrm{X}}$ is not convex due to the rank constraint, we get a local minimizer $\hat{\mathrm{X}} = \hat{\mathrm{U}}\hat{\mathrm{V}}^{\mathsf{T}}$ in the experiments and we use it as the hypothesis instead of the global minimizer.

Because of our bounds based on Theorem 23 holds for any $\hat{\mathrm{X}}$ in the hypothesis class $\mathcal{X}$, we can use the generalization error bounds under this optimization scheme.

5.9 Yet another proof of Lemma 16

Proof. Given $\mathrm{V} = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_K) \in (\mathrm{conv}(\mathcal{P})^K)$ with $\boldsymbol{v}_i = \sum_{\boldsymbol{p} \in \mathcal{P}} \alpha_{\boldsymbol{p}}^{(i)} \boldsymbol{p}$ $(i \in [K])$, we can construct a convex combination $\mathrm{X} = \sum_{(\boldsymbol{p}_1, \ldots, \boldsymbol{p}_K) \in \mathcal{P}^K} \alpha_{\boldsymbol{p}_1 \boldsymbol{p}_2 \cdots \boldsymbol{p}_K} (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_K)$ by the following procedure.

1. Let $\mathrm{Y} \leftarrow \mathrm{V}$ and $k \leftarrow 1$.

2. While $\mathrm{Y} \geq \mathbf{0}$, repeat:

(a) Let $\mathcal{P}^{(i)} = \{\boldsymbol{p} \in \mathcal{P} \mid \alpha_{\boldsymbol{p}}^{(i)} > 0\}$.

(b) Let $(i_k, \boldsymbol{p}_{i_k}) = \arg\min_{i \in [K]} \min_{\boldsymbol{p} \in \mathcal{P}^{(i)}} \alpha_{\boldsymbol{p}}^{(i)}$.

(c) Pick up any $\boldsymbol{p}_j \in \mathcal{P}^{(j)}$ for $j \neq i_k$ and construct a vector $\boldsymbol{q}_k = (\boldsymbol{p}_1, \ldots, \boldsymbol{p}_{i_k}, \ldots, \boldsymbol{p}_K)$.

(d) Update $\mathrm{Y} \leftarrow \mathrm{Y} - \alpha_{\boldsymbol{p}_{i_k}}^{(i_k)} \boldsymbol{q}_k$.

(e) Update all $\alpha_{\boldsymbol{p}}^{(i)}$ and $k \leftarrow k + 1$.

3. Output $\sum_{k \geq 1} \alpha_{\boldsymbol{p}_{i_k}}^{(i_k)} \boldsymbol{q}_k$.

Note that the while loop terminates in $K|\mathcal{P}|$ trials since the procedure increases a zero entry by at least one at each loop. Also, one can observe that each $\alpha_{\boldsymbol{p}_{i_k}}^{(i_k)} \geq 0$ and $\sum_{k \geq 1} \alpha_{\boldsymbol{p}_{i_k}}^{(i_k)} = 1$. Therefore the procedure outputs a convex decomposition of Y. □

Chapter 6

A Combinatorial Metrical Task System Problem under the Uniform Metric

6.1 Introduction

The metrical task system is defined as a repeated game between the player and the adversary. Given a fixed set C of states a metric $\delta : C \times C \to \mathbb{R}_+$ and a initial state $c_0 \in C$, for each round $t = 1, \ldots, T$, (i) the adversary reveals a (processing) cost function $f_t : C \to \mathbb{R}_+$, (ii) the player chooses a state $c_t \in C$, and (iii) the player incurs the processing cost $f_t(c_t)$ and the moving cost $\delta(c_t, c_{t-1})$. The goal of the algorithm is minimizing the cumulative (processing and moving) cost. The performance of the algorithm is measured by the competitive ratio, that is, the ratio of the cumulative cost of the algorithm to the cumulative cost of the best fixed sequence of states in hindsight.

In the expert setting, i.e., where the decision set consists of n states, there are many existing works on the MTS [16, 40, 8, 29, 9]. In particular, for the uniform metric δ (which is defined as $\delta(i, j) = 1$ if $i \neq j$ and otherwise $\delta(i, j) = 0$), the MTS problem is well studied [16, 8, 40, 1]. Borodin et al. show the lower bound of the competitive ratio of any randomized algorithm is H_n, where H_n is the n-th harmonic number [16]. Especially, Abernethy et al. provide an algorithm which uses the method of convex optimization, and shows the upper bound of the competitive ratio of the algorithm is $H_n + O(\log \log n)$ [1].

When we consider the situation where the decision set C is a combinatorial set from $\{0, 1\}^d$ (e.g., the set of spanning trees or s-t paths of a graph), the computational issue arises. A natural example of a combinatorial MTS is a routing problem. For example, we consider a routing problem. Consider a fixed network $G = (V, E)$ where V is the set of routers (nodes) and

$E \subseteq V \times V$ is the set of d edges between routers and V includes two routers, the source s and the sink t. The decision set C is the set of paths from s to t, whose size is exponential in d. In general, for typical combinatorial sets, the size could be exponential in the dimension size d as well and straightforward implementations of known algorithms for the MTS take exponential time as well since time complexity of these algorithms is proportional to the size n of the decision set.

In this chapter, for the uniform metric, we propose a modification of the Marking algorithm [16], which we call the Weighted Marking algorithm. The weighted Marking algorithm employs an exponential weighting scheme and can be viewed as an analogue of the Hedge algorithm [32] for the MTS problem, whereas the Marking algorithm is an analogue of the classical Halving algorithm. We prove that the Weighted Marking algorithm retains $O(\log n)$ competitive ratio for the standard MTS problem with n states. The expected running time of the Weighted Marking algorithm at each round is the same as that of the original one.

On the other hand, combining with efficient sampling techniques w.r.t. exponential weights on combinatorial objects (k-sets, s-t paths [73], stars in a graph [19] permutation matrices [19, 41], permutation vectors [2]), the Weighted Marking algorithm works efficiently for various classes of combinatorial sets.

6.1.1 Related Work

There are some existing works for combinatorial metrical task systems. Blum et al. provide algorithms for the list update problem [14]. For the k-server problem, which can be viewed as a combinatorial MTS problem, Koutsoupias et al. provide a deterministic algorithm [49]. Bansal et al. improve the results of Koutsoupias et al. by a randomization technique [6]. These algorithms are efficient and perform well for specific problems, i.e., the list update problem and the k-server problem. However, these algorithms are specialized for limited decision sets and we cannot use them for other problems.

Buchbinder et al. consider combinatorial MTS problems where the decision space is defined as a matroid [17]. The concept of matroid can express various classes of combinatorial objects such as spanning trees. They show a unified algorithm with a guaranteed competitive ratio. Their analysis is, however, applicable for a continuous "relaxed" space only. It is not known if there exists a rounding scheme that approximately preserves the moving cost over the relaxed space. Gupta et al. also consider combinatorial MTS problems over the basis of a matroid [34].

They give a rounding algorithm and prove the competitive ratio of a rounded solution, for a class of metrics including the Hamming distance but not the uniform metric.

6.2 Preliminaries

A metrical task system (MTS) is a pair (C, δ) where C is a set of states and $\delta : C \times C \to \mathbb{R}_+$ is a metric. In particular, we consider a combinatorial setting where C is a subset of $\{0,1\}^d$ for some dimension $d > 0$. We denote by n the size of C, that is, $n = |C|$. Typically, n is exponentially large in d. Moreover, we only consider the uniform metric δ, that is,

$$\delta(\boldsymbol{c}_1, \boldsymbol{c}_2) = \begin{cases} 1 & \text{if } \boldsymbol{c}_1 \neq \boldsymbol{c}_2, \\ 0 & \text{if } \boldsymbol{c}_1 = \boldsymbol{c}_2. \end{cases}$$

The combinatorial MTS problem for (C, δ) is defined as the following protocol between the algorithm and the adversary.

First the adversary chooses a task sequence $\sigma = (\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots, \boldsymbol{\ell}_T)$, where each $\boldsymbol{\ell}_t \in [0,1]^d$ is called a loss vector. In other words, we assume the oblivious setting. For a given initial state $\boldsymbol{c}_0 \in C$, the protocol proceeds in rounds, where in each round $t = 1, 2, \ldots, T$,

1. the algorithm receives the loss vector $\boldsymbol{\ell}_t \in [0,1]^d$,
2. the algorithm chooses a state $\boldsymbol{c}_t \in C$, and
3. the algorithm suffers a cost given by $\boldsymbol{c}_t \cdot \boldsymbol{\ell}_t + \delta(\boldsymbol{c}_t, \boldsymbol{c}_{t-1})$.

The first term $\boldsymbol{c}_t \cdot \boldsymbol{\ell}_t$ of the cost is called the processing cost at round t, and the second term $\delta(\boldsymbol{c}_t, \boldsymbol{c}_{t-1})$ is called the moving cost at round t.

For a task sequence σ, the cumulative cost of an algorithm A is defined as

$$\mathrm{cost}_A(\sigma) = \sum_{t=1}^{T} (\boldsymbol{c}_t \cdot \boldsymbol{\ell}_t + \delta(\boldsymbol{c}_t, \boldsymbol{c}_{t-1})),$$

and the cumulative cost of the best offline solution is defined as

$$\mathrm{cost}_{\mathrm{OPT}}(\sigma) = \min_{(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, \ldots, \boldsymbol{c}_T^*) \in C^T} \sum_{t=1}^{T} (\boldsymbol{c}_t^* \cdot \boldsymbol{\ell}_t + \delta(\boldsymbol{c}_t^*, \boldsymbol{c}_{t-1}^*)).$$

We measure the performance of algorithm A by its competitive ratio, which is defined as

$$\mathrm{CR}(\sigma) = \frac{\mathbb{E}[\mathrm{cost}_A(\sigma)]}{\mathrm{cost}_{\mathrm{OPT}}(\sigma)},$$

where the expectation is with respect to the internal randomness of A. The goal of the algorithm is to minimize the worst case competitive ratio $\max_\sigma \mathrm{CR}(\sigma)$. Note that the usual (non-combinatorial) MTS problem is a special case where C consists of unit vectors.

We also require the algorithm A to produce a state $\boldsymbol{c}_t$ in time polynomial in d for each round t. Typically, the size n of C is exponential in d, and so we cannot directly maintain all states c in C. Therefore, we assume two oracles to access the state set C efficiently. The first one is the linear optimization oracle, which solves the following decision problem:

$\underline{\text{OPT}(C)}$

Input: $\boldsymbol{L} \in \mathbb{R}^d_+$

Output: $\begin{cases} 0 & \text{if } \min_{\boldsymbol{c}\in C} \boldsymbol{c}\cdot\boldsymbol{L} < 1, \\ 1 & \text{otherwise.} \end{cases}$

The assumption of this oracle is natural since the linear optimization problem has a polynomial time algorithm for many useful state sets C.

The second one is a sampling oracle, which chooses a state $\boldsymbol{c}$ randomly according to a certain probability distribution over C, where the distribution is specified by a given parameter $\boldsymbol{L} \in \mathbb{R}^d_+$. In particular, we consider two kinds of sampling oracles, which will be defined later.

6.3 The Marking algorithm

Here we apply the Marking algorithm [16] to the combinatorial MTS problem. The Marking algorithm is a simple randomized algorithm whose competitive ratio is upper bounded by $2H_n \le 2(\ln n + 1)$, where H_n is the n-th harmonic number.

Below we describe how the Marking algorithm works. For a naive implementation, it maintains the cumulative processing costs $l[\boldsymbol{c}]$ for all states $\boldsymbol{c} \in C$. For each round t,

1. Observe the loss vector $\boldsymbol{\ell}_t$ and update $l[\boldsymbol{c}] = l[\boldsymbol{c}] + \boldsymbol{c}\cdot\boldsymbol{\ell}_t$ for all $\boldsymbol{c} \in C$.
2. If $l[\boldsymbol{c}_{t-1}] < 1$ then output $\boldsymbol{c}_t = \boldsymbol{c}_{t-1}$.
3. Else choose a state $\boldsymbol{c}_t$ uniformly at random from the set of states $\boldsymbol{c}$ with $l[\boldsymbol{c}] < 1$, and output $\boldsymbol{c}_t$.

4. If no such states exist, then reset $l[\boldsymbol{c}] = 0$ for all $\boldsymbol{c} \in C$ and choose a state $\boldsymbol{c}_t$ uniformly at random from C, and output $\boldsymbol{c}_t$.

Note that Line 2 and Line 3 intuitively mean that the Marking algorithm does not change states until $l[\boldsymbol{c}_t] \geq 1$. As is well known as a folklore (See, e.g., [15]), we can assume without loss of generality that the loss vectors $\boldsymbol{\ell}_t$ are small enough so that $l[\boldsymbol{c}_t] \leq 1$ always holds. In the appendix we give more detailed discussion. In other words, the Marking algorithm changes states only when $l[\boldsymbol{c}_t] = 1$.

Of course, the naive implementation of the Marking algorithm is not efficient because it maintains the cumulative processing cost $l[\boldsymbol{c}]$ for all states $\boldsymbol{c} \in C$. Instead, we can maintain the cumulative loss vector $\boldsymbol{L} = \sum_t \boldsymbol{\ell}_t$, which implicitly maintains $l[\boldsymbol{c}]$ as $l[\boldsymbol{c}] = \boldsymbol{c} \cdot \boldsymbol{L}$ for all $\boldsymbol{c}$. Furthermore, the sampling problem at Line 3 can be restated as the following problem in terms of $\boldsymbol{L}$, which we call Sampling 1.

Sampling 1
Input: $\boldsymbol{L} \in \mathbb{R}^d_+$,
Output: $\boldsymbol{c} \in C_{\boldsymbol{L}} = \{\boldsymbol{c} \in C \mid \boldsymbol{c} \cdot \boldsymbol{L} < 1\}$ uniformly at random.

Note that the problem Sampling 1 is only defined when $C_{\boldsymbol{L}} \neq \emptyset$, but we can check whether the condition holds by using the linear optimization oracle for OPT(C). Moreover, the uniform sampling at Line 4 is also restated as Sampling 1 with $\boldsymbol{L} = \boldsymbol{0}$. So, if we assume a linear optimization oracle for OPT(C) and a sampling oracle for Sampling 1, then we can emulate the Marking algorithm in $O(d)$ time per round. We give this implementation of the Marking algorithm in Algorithm 5.

Algorithm 5 An implementation of the Marking algorithm

Input: A linear optimization oracle for OPT(C) and a sampling oracle for Sampling 1
Initialize: Let $\boldsymbol{L} = \boldsymbol{0}$.
For each round $t = 1, 2, \ldots, T$,

1. Observe the loss vector $\boldsymbol{\ell}_t$ and update $\boldsymbol{L} = \boldsymbol{L} + \boldsymbol{\ell}_t$.
2. Let $\boldsymbol{c}_t = \boldsymbol{c}_{t-1}$ and output $\boldsymbol{c}_t$.
3. If $\boldsymbol{c}_t \cdot \boldsymbol{L} \geq 1$, then
   (a) If $\min_{\boldsymbol{c} \in C} \boldsymbol{c} \cdot \boldsymbol{L} \geq 1$, then reset $\boldsymbol{L} = \boldsymbol{0}$. // use the linear optimization oracle
   (b) Choose a state $\boldsymbol{c}_t \in C_{\boldsymbol{L}}$ uniformly at random. // use the sampling oracle

The question that naturally arises is that for what state set C, the problem Sampling 1 is efficiently solved. Unfortunately, we do not know any non-trivial sets C that have polynomial time algorithm for Sampling 1. We could use MCMC sampling methods to design approximate sampling, but it seems hard to show theoretically guaranteed performance bounds for many natural state sets C.

6.4 The Weighted Marking algorithm

The computational cost of the sampling problem Sampling 1 would be due to the fact that the support of the sampling distribution is restricted to the set $C_{\boldsymbol{L}}$. So, we relax the distribution to a continuous distribution whose support is not restricted to $C_{\boldsymbol{L}}$.

Specifically, we propose the following sampling problem, called Sampling 2.

Sampling 2

Input: $\boldsymbol{L} \in \mathbb{R}^d_+$,

Output: $\boldsymbol{c} \in C$ chosen with probability $\pi_{\boldsymbol{L}}(\boldsymbol{c}) = \dfrac{\exp(-\eta \boldsymbol{c} \cdot \boldsymbol{L})}{\sum_{\boldsymbol{c} \in C} \exp(-\eta \boldsymbol{c} \cdot \boldsymbol{L})}$,
where $\eta > 0$ is a parameter.

In words, the new sampling distribution $\pi_{\boldsymbol{L}}$ is such that $\pi_{\boldsymbol{L}}(\boldsymbol{c})$ is a monotone decreasing function with respect to its cumulative processing cost $l[\boldsymbol{c}] = \boldsymbol{c} \cdot \boldsymbol{L}$. So, the probability that a state $\boldsymbol{c}$ with large $l[\boldsymbol{c}]$ is chosen is very low, and thus we will see that the support of $\pi_{\boldsymbol{L}}$ is essentially restricted to a set $\{\boldsymbol{c} \in C \mid \boldsymbol{c} \cdot \boldsymbol{L} < L\}$ for some $L > 1$.

Unlike Sampling 1, there are known efficient implementations of Sampling 2 for several combinatorial objects such as k-sets, s-t paths [73], permutation matrices [19, 41], stars in a graph [19] and permutation vectors [2].

Now we modify the Marking algorithm by assuming the sampling oracle for Sampling 2, as well as assuming the linear optimization oracle for OPT(C). The modified version is called the Weighted Marking algorithm. The difference from the Marking algorithm is that (1) it does not change states until its cumulative processing cost reaches L instead of 1, and (2) it uses $\pi_{\boldsymbol{L}}$ as the sampling distribution instead of the uniform distribution over $C_{\boldsymbol{L}}$. Note that the Weighted Marking algorithm resets the cumulative loss vector as $\boldsymbol{L} = \boldsymbol{0}$ when $\min_{\boldsymbol{c} \in C} \boldsymbol{c} \cdot \boldsymbol{L}$ reaches 1, which is the same condition as the Marking algorithm. So, unlike the Marking algorithm, resetting $\boldsymbol{L}$ may happen at some round where the cumulative processing cost of the current state does not reach L, since $L \neq 1$.

The detailed description of the Weighted Marking algorithm is given in Algorithm 6.

Algorithm 6 Weighted Marking algorithm

Input: A linear optimization oracle for OPT(C) and a sampling oracle for Sampling 2
Parameter: $\eta > 0$ and $L > 1$ such that $ne^{-\eta L} \leq e^{-\eta}/2$.
Initialize: Let $\boldsymbol{L} = \mathbf{0}$.
For each round $t = 1, 2, \ldots, T$,

1. Observe the loss vector $\boldsymbol{\ell}_t$ and update $\boldsymbol{L} = \boldsymbol{L} + \boldsymbol{\ell}_t$.
2. Let $\boldsymbol{c}_t = \boldsymbol{c}_{t-1}$ and output $\boldsymbol{c}_t$.
3. If $\min_{\boldsymbol{c} \in C} \boldsymbol{c} \cdot \boldsymbol{L} \geq 1$ then // use the linear optimization oracle
   (a) Reset $\boldsymbol{L} = \mathbf{0}$.
   (b) Choose a state $\boldsymbol{c}_t \in C$ with probability $\pi_{\boldsymbol{L}}(\boldsymbol{c})$ // use the sampling oracle
4. Else if $\boldsymbol{c}_t \cdot \boldsymbol{L} \geq L$, then
   (a) Repeat
   Choose a state $\boldsymbol{c}_t \in C$ with probability $\pi_{\boldsymbol{L}}(\boldsymbol{c})$ // use the sampling oracle
   Until $\boldsymbol{c}_t \cdot \boldsymbol{L} < L$.

For convenience, we define the notion of phases for analyzing the behavior of the Weighted Marking algorithm. A phase is an interval $\{t \mid t_b \leq t \leq t_e\}$ of rounds such that the resetting happens at round $t_b - 1$ and t_e but does not happen at every round $t_b \leq t < t_e$.

Again, as is well known as a folklore, we assume without loss of generality that the loss vectors $\boldsymbol{\ell}_t$ are small enough so that it always holds that $\min_{\boldsymbol{c} \in C} \boldsymbol{c} \cdot \boldsymbol{L} \leq 1$ at Line 3 and it always hold that $\boldsymbol{c}_t \cdot \boldsymbol{L} \leq L$ at Line 4. In other words, a phase ends (resetting happens) only when $\min_{\boldsymbol{c} \in C} \boldsymbol{c} \cdot \boldsymbol{L} = 1$ and states $\boldsymbol{c}_t$ are changed only when $\boldsymbol{c}_t \cdot \boldsymbol{L} = L$. These assumptions greatly simplifies the analysis.

More formally, the assumption is described as follows:

Assumption 29. *Whenever the previous state $\boldsymbol{c}_{t-1}$ satisfies $\boldsymbol{c}_{t-1} \cdot \boldsymbol{L} < L$, where $\boldsymbol{L}$ is the cumulative loss vectors up to round $t-1$ in the current phase, and the phase did not end at round $t-1$, i.e., $\min_{\boldsymbol{c}^* \in C} \boldsymbol{c}^* \cdot \boldsymbol{L} < L$, then $\boldsymbol{\ell}_t$ satisfies the two conditions:*

1. $\boldsymbol{c}_{t-1} \cdot (\boldsymbol{L} + \boldsymbol{\ell}_t) \leq L$, *and*
2. $\min_{\boldsymbol{c}^* \in C} \boldsymbol{c}^* \cdot (\boldsymbol{L} + \boldsymbol{\ell}_t) \leq 1$.

As is well known as a folklore, we assume Assumption 29 holds throughout this section. We can assume without loss of generality that the loss vectors $\boldsymbol{\ell}_t$ are small enough, so that Assumption 29 is satisfied. This is because, when $\boldsymbol{\ell}_t$ violates the assumption, then we can replace $\boldsymbol{\ell}_t$ by a sequence of non-negative loss vectors $\boldsymbol{\ell}_{t_1}, \boldsymbol{\ell}_{t_2}, \dots, \boldsymbol{\ell}_{t_k}$ so that $\boldsymbol{\ell}_t = \boldsymbol{\ell}_{t_1} + \cdots + \boldsymbol{\ell}_{t_k}$ and the new sequence of loss vectors satisfy the assumption in the following way:

1. If the first condition is violated, i.e., $\boldsymbol{c}_{t-1} \cdot (\boldsymbol{L} + \boldsymbol{\ell}_t) = a > L$, then we let

$$\alpha_1 = \frac{L - \boldsymbol{c}_{t-1} \cdot \boldsymbol{L}}{a - \boldsymbol{c}_{t-1} \cdot \boldsymbol{L}}.$$

   Otherwise, we let $\alpha_1 = 1$. In the former case, we can easily verify that $0 < \alpha_1 < 1$ and $\boldsymbol{c}_{t-1} \cdot (\boldsymbol{L} + \alpha_1 \boldsymbol{\ell}_t) = L$.

2. If the second condition is violated, i.e., $\min_{\boldsymbol{c}^* \in C} \boldsymbol{c}^* \cdot (\boldsymbol{L} + \boldsymbol{\ell}_t) > 1$, then we let $0 < \alpha_2 < 1$ be such that $\min_{\boldsymbol{c}^* \in C} \boldsymbol{c}^* \cdot (\boldsymbol{L} + \alpha_2 \boldsymbol{\ell}_t) = 1$. Otherwise, we let $\alpha_2 = 1$. Note that, in the former case, we can find such α_2 efficiently by binary search.

3. Let $\alpha = \min\{\alpha_1, \alpha_2\}$ and $\boldsymbol{\ell}_{t_1} = \alpha \boldsymbol{\ell}_t$ and $\boldsymbol{\ell}_{t_2} = (1 - \alpha)\boldsymbol{\ell}_t$. Then, clearly $\boldsymbol{\ell}_{t_1}$ satisfies Assumption 1. If $\boldsymbol{\ell}_{t_2}$ still violates the assumption, then repeat the same procedure for $\boldsymbol{\ell}_{t_2}$ recursively.

In the next theorem, we give an upper bound of the competitive ratio of the Weighted Marking algorithm.

Theorem 30 (Main theorem of Chapter 6)**.** *Let $\eta = \ln 2n$, and $L = 2$. Then for any task sequence $\sigma = (\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \dots, \boldsymbol{\ell}_T)$, the competitive ratio of the Weighted Marking algorithm is upper bounded by*

$$\mathrm{CR}(\sigma) \leq 6e \ln n + 9.$$

Moreover, the expected running time per round is $O(d + T_{\mathrm{lin}} + T_{\mathrm{Samp2}})$, where T_{lin} is the running time of the linear optimization oracle and T_{Samp2} is that of the sampling oracle for Sampling 2.

To prove this theorem, we show that the cumulative moving cost in each phase is $O(\log n)$. So in the following, we fix a particular phase $I = \{t_b, \dots, t_e\}$. For each round $t \in I$, $\boldsymbol{L}_t$ denotes the cumulative loss vector $\boldsymbol{L}$ at Line 1 at round t. Note by definition that $\min_{\boldsymbol{c}^* \in C} \boldsymbol{c}^* \cdot \boldsymbol{L}_{t_e} = 1$.

Let $G = \{\boldsymbol{c} \in C \mid \boldsymbol{c} \cdot \boldsymbol{L}_{t_e} < L\}$ be the goal set, meaning that if we choose a state in G at some round $t \in I$, i.e., $\boldsymbol{c}_t \in G$, then the Weighted Marking algorithm never changes the state

until the end of the phase. Note that $\boldsymbol{c}^* \in G$ and so $G \neq \emptyset$. Let $\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_n$ be the members of C. (This is an abuse of notation. Do not confuse them with the states $\boldsymbol{c}_t$ the algorithm chooses at round t.) For any $\boldsymbol{c}_i \notin G$, we can define $t_i \in I$ such that $\boldsymbol{c}_i \cdot \boldsymbol{L}_{t_i} = L$. Then, without loss of generality, we assume $t_1 \leq t_2 \leq \cdots \leq t_{n-|G|}$ and $\boldsymbol{c}_n = \boldsymbol{c}^*$, i.e., $\boldsymbol{c}_n \cdot \boldsymbol{L}_{t_e} = 1$. Moreover, we assume $|G| = 1$ just for simplicity. Clearly, the algorithm changes states only at some rounds in $\{t_1, \ldots, t_{n-1}\}$. Let $t^{(k)}$ be the round where the algorithm makes the k-th change of states. For any state $\boldsymbol{c} \in C$, we define the weight function $W_k(\boldsymbol{c})$ as

$$W_k(\boldsymbol{c}) := \begin{cases} e^{-\eta \boldsymbol{c} \cdot \boldsymbol{L}_{t^{(k)}}} & \text{if } \boldsymbol{c} \cdot \boldsymbol{L}_{t^{(k)}} < L, \\ 0 & \text{if } \boldsymbol{c} \cdot \boldsymbol{L}_{t^{(k)}} \geq L. \end{cases}$$

Let $\overline{W}_k := \sum_{\boldsymbol{c} \in C} W_k(\boldsymbol{c})$. Then $W_k(\boldsymbol{c})/\overline{W}_k$ is the probability of choosing state $\boldsymbol{c}$ at the k-th change of states. One can see that $W_k(\boldsymbol{c})$ is monotonically decreasing w.r.t. k because L_t is monotonically increasing vector w.r.t. t.

If the best offline solution changes its state in the phase, then its cumulative moving cost is at least 1, and otherwise its cumulative processing cost is at least 1 by the definition of the phase. This immediately implies the following lemma.

Lemma 17. *For any sequence of loss vectors $(\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \cdots, \boldsymbol{\ell}_T)$, the best offline solution suffers cost at least 1 on each phase.*

On the other hand, whenever the Weighted Marking algorithm changes states (i.e., suffers the moving cost of 1) from $\boldsymbol{c}_{t^{(k-1)}}$ to $\boldsymbol{c}_{t^{(k)}}$, then its cumulative processing cost from $t^{(k-1)}$ to $t^{(k)}$ is at most L. This implies the following lemma.

Lemma 18. *For any sequence of loss vectors $(\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \cdots, \boldsymbol{\ell}_T)$, the cumulative processing cost of the Weighted Marking algorithm is at most L times the cumulative moving cost on each phase.*

The following lemma provides the probability of ending a phase.

Lemma 19. *For any $\alpha \in (0, 1)$ and for any k, if $\alpha \overline{W}_k \leq e^{-\eta}$ holds then the phase will end at the $k+1$-th change of the state with probability at least α.*

Proof. By the assumption $\boldsymbol{c}_n \cdot \boldsymbol{L}_{t_e} = 1$, if the algorithm choose $\boldsymbol{c}_n$ then the algorithm will change its state at the end of the phase t_e, i.e. if the state $\boldsymbol{c}_n$ is chosen then the phase rests only 1 change. By $\boldsymbol{c}_n \cdot \boldsymbol{L} \leq 1$, we get $W_k(\boldsymbol{c}_n) \geq e^{-\eta}$ for any k. Using this and the condition of the lemma, we get

$$\alpha \leq \frac{e^{-\eta}}{\overline{W}_k} \leq \frac{W_k(\boldsymbol{c}_n)}{\overline{W}_k}.$$

Here, the right hand side is the probability of the state $\boldsymbol{c}_n$ will be chosen by the Weighted Marking algorithm. □

The following lemma guarantees the probability of choosing $\boldsymbol{c}_n$ becomes higher at each change of the state.

Lemma 20. *For any $\alpha \in (0,1)$, for any k, if $\alpha\overline{W}_k \geq e^{-\eta}$ holds then*

$$\Pr[\overline{W}_{k+1} \leq \alpha\overline{W}_k] > \alpha.$$

Proof. Summing up weights of states from $n, n-1, \cdots$ and consider when the sum gets greater than $\alpha\overline{W}_k$. E.g. consider i_k s.t. $\sum_{i=i_k+1}^{n} W_k(\boldsymbol{c}_i) \leq \alpha\overline{W}_k$ and $\sum_{i=i_k}^{n} W_k(\boldsymbol{c}_i) > \alpha\overline{W}_k$.

Assume that the Weighted Marking algorithm chooses the state $\boldsymbol{c}_s$ at the k-th change of the state. If $s \geq i_k$, the algorithm changes its state at $t^{(k+1)}$ and then $W_{k+1}(\boldsymbol{c}_i) = 0$ for any $i \geq i_k$ by the definition of W and i_k. Thus,

$$\overline{W}_{k+1} = \sum_{i=1}^{n} W_{k+1}(\boldsymbol{c}_i) = \sum_{i=i_k+1}^{n} W_{k+1}(\boldsymbol{c}_i).$$

Because W_k is monotonically decreasing w.r.t. k, one can get

$$\sum_{i=i_k+1}^{n} W_{k+1}(\boldsymbol{c}_i) \leq \sum_{i=i_k+1}^{n} W_k(\boldsymbol{c}_i) \leq \alpha\overline{W}.$$

So we get if $s \geq i_k$ then $\overline{W}_{k+1} \leq \alpha\overline{W}$. The probability of the Weighted Marking algorithm choosing the state $\boldsymbol{c}_s$ such that $s \geq i_k$ satisfies

$$\Pr[s \geq i_k] = \frac{\sum_{i=i_k}^{n} W_k(\boldsymbol{c}_i)}{\overline{W}_k} > \frac{\alpha\overline{W}_k}{\overline{W}_k} = \alpha.$$

□

By Lemma 19, one can get the following immediately.

Lemma 21. *For any $\alpha \in (0,1)$ and round $t^{(k)}$, if $\alpha\overline{W}_k \leq e^{-\eta}$ then the expected number of remaining changes of states in the phase is less than $\frac{1}{\alpha} + 1$.*

Because of W_k is monotonically decreasing w.r.t. k and Lemma 20, one can get the following lemma.

Lemma 22. *For any k, for any $\alpha \in (0,1)$, if $\alpha\overline{W}_k \leq e^{-\eta}$ then the expectation of m such that $\overline{W}_{k+m} \leq \alpha\overline{W}_k$ is $\mathbb{E}[m] < \frac{1}{\alpha}$.*

We say that a sequence $\overline{W} = \{\overline{W}_1, \overline{W}_2, \cdots, \overline{W}_K\}$ of weights is α-fast decreasing at the round $t^{(k+1)}$ if $\overline{W}_{k+1} \geq \alpha\overline{W}_k$ holds.

Proof of Theorem 30. Assume that the Weighted Marking algorithm changes its state at K times in a phase. By Lemma 21, if $\alpha\overline{W}_{k'} \leq e^{-\eta}$ holds then we have

$$\mathbb{E}[K] \leq k' + \frac{1}{\alpha} + 1.$$

Thus, we need to estimate k' s.t. $\alpha\overline{W}_{k'} \leq e^{-\eta}$ to bound $\mathbb{E}[K]$.

Let $\alpha\overline{W}_{k'} \leq e^{-\eta}$ holds after α-fast decreasing K' times, then

$$\alpha^{K'}\overline{W}_0 \leq \overline{W}_{k'} \leq \frac{e^{\eta}}{\alpha}.$$

By $\overline{W}_0 = n$, we get $\alpha^{K'}n \leq \frac{e^{-\eta}}{\alpha}$ and rearranging, $K' \leq \frac{1}{\ln\frac{1}{\alpha}}(\ln n + \eta) - 1$. Using Lemma 22,

$$\mathbb{E}[k'] \leq \mathbb{E}[m]K' = \frac{1}{\alpha}K' = \frac{1}{\alpha\ln\frac{1}{\alpha}}(\ln n + \eta) - \frac{1}{\alpha}.$$

Thus, the number of changing of states at a phase is

$$\mathbb{E}[K] \leq \mathbb{E}[k'] + \frac{1}{\alpha} + 1 = \frac{1}{\alpha\ln\frac{1}{\alpha}}(\ln n + \eta) + 1.$$

The bound of $\mathbb{E}[K]$ is minimized when $\alpha = 1/e$. So we get $\mathbb{E}[K] \leq e(\ln n + \eta) + 1$.

Setting $\eta = \ln 2n$, we get $\mathbb{E}[K] \leq 2e\ln n + 3$. By Lemma 18,

$$\mathbb{E}[\text{(cumulative processing cost)}] \leq L \times \mathbb{E}[\text{(cumulative moving cost)}].$$

At each phase, we have

$$\begin{aligned}
&\mathbb{E}[\text{Cumulative loss}] \\
&= \mathbb{E}[\text{Cumulative processing cost}] + \mathbb{E}[\text{Cumulative moving cost}] \\
&\leq 3 \times \mathbb{E}[K] \\
&\leq 6e\ln n + 9.
\end{aligned}$$

By Lemma 17, at each phase the best offline solution has the cumulative processing cost at least 1. Thus we get the bound of the competitive ratio. □

Next, we prove the running time of the Weighted Marking algorithm. The key point of analysis of the Weighted Marking algorithm is the number of calls to the oracle for Sampling 2 at Line 4-(a) of the pseudo code. The following lemma gives a theoretical bound of retrying.

Lemma 23. *The expected number of calls to the sampling oracle at Line 4-(a) is at most* 2*.*

Proof. For any state $\boldsymbol{c}$ such that $\boldsymbol{c} \cdot \boldsymbol{L} \geq L$, the probability that the sampling oracle chooses $\boldsymbol{c}$ is

$$\begin{aligned} \frac{\exp(-\eta \boldsymbol{c} \cdot \boldsymbol{L})}{\sum_{\boldsymbol{c}'} \exp(-\eta \boldsymbol{c}' \cdot \boldsymbol{L})} &\leq \frac{\exp(-\eta L)}{\exp(-\eta \boldsymbol{c}_n \cdot \boldsymbol{L})} \\ &\leq \frac{\exp(-\eta L)}{\exp(-\eta)} \end{aligned}$$

since $\boldsymbol{c}_n \cdot \boldsymbol{L} < 1$. By the union bound, the probability that the sampling oracle chooses some $\boldsymbol{c}$ with $\boldsymbol{c} \cdot \boldsymbol{L} \geq L$ is at most

$$\frac{n \exp(-\eta L)}{\exp(-\eta)} = \frac{1}{2}$$

by our choice of η and L. □

Finally, we show our upper bound of the competitive ratio given in Theorem 30 is tight if C is an k-set by deriving the asymptotic lower bound of the asymptotic competitive ratio of the combinatorial MTS problem. The asymptotic lower bound of the competitive ratio for MTS problem in expert setting under the uniform metric is nH_n where $H_n = \sum_{i=1}^{n} \frac{1}{i}$ [16].

We call $CR(\sigma)$ is a asymptotic competitive ratio if there exists some constant K such that

$$\mathbb{E}[\mathrm{cost}_A(\sigma)] - CR(\sigma)\mathrm{cost}_{\mathrm{OPT}}(\sigma) \leq K.$$

If $\mathrm{cost}_{\mathrm{OPT}}(\sigma) \to \infty$ with $T \to \infty$, $CR(\sigma) \to \mathrm{CR}(\sigma)$.

Theorem 31. *Let $C = \{\boldsymbol{c} \in \{0,1\}^d : \|\boldsymbol{c}\|_1 = k\}$ and δ be the uniform metric. For any algorithm for MTS problem (C, δ), there exists a task sequence $\sigma = (\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots, \boldsymbol{\ell}_T)$ such that the competitive ratio of the algorithm is lower bounded by*

$$\mathrm{CR}(\sigma) \geq k(H_d - H_k) = O(k \log \frac{d}{k}).$$

To prove the lower bound, we use Lemma 7.2 of [16]. If $\sigma = (\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots)$ is infinite sequence of tasks, let $\sigma_T = (\boldsymbol{\ell}_1, \boldsymbol{\ell}_2, \ldots, \boldsymbol{\ell}_T)$ be the first T tasks.

Lemma 24 (Lemma 7.2 of [16])**.** *Let D be any distribution over infinite task sequence σ. Suppose that $\lim_{T\to\infty} \mathbb{E}_D[\mathrm{cost}_{\mathrm{OPT}}(\sigma_T)] \to \infty$. Then*

$$\mathrm{CR}(\sigma) \geq \lim_{T\to\infty} \sup \frac{\min_{A\in\mathcal{A}} \mathbb{E}_D[\mathrm{cost}_A(\sigma_T)]}{\mathbb{E}_D[\mathrm{cost}_{\mathrm{OPT}}(\sigma_T)]}$$

where $\mathcal{A}$ is the set of all deterministic algorithm.

No we give the proof of the theorem 31.

Proof. We show that $\min_{A\in\mathcal{A}} \mathbb{E}_D[\mathrm{cost}_A(\sigma_T)] \geq kT/d$ and $\mathbb{E}_D[\mathrm{cost}_{\mathrm{OPT}}(\sigma_T)] \leq T/d(H_d - H_k)$.

Let D be the distribution over the loss sequences obtained by choosing $\boldsymbol{\ell}_t = \boldsymbol{e}_{i_t}$ with probability $1/d$ at each round t, where $\boldsymbol{e}_i$ has 1 in the i-th component and others are all 0.

In round t, any algorithm suffers loss 1 if $\boldsymbol{\ell}_t = \boldsymbol{e}_{i_t}$ and the i_t-th component of the current state $c_{t-1,i_t} = 1$. Since $\boldsymbol{c}_{t-1}$ represents a k-set, this happens with probability k/d and hence $\min_{A\in\mathcal{A}} \mathbb{E}_D[\mathrm{cost}_A(\sigma_T)] \geq kT/d$.

Let $a(t)$ be minimum of the round such that the set $\{i_{t+1}, i_{t+2}, \ldots, i_{a(t)}\} = \mathcal{I}$ contains $d-k$ indices. Now we consider the following offline algorithm A.

$$A := \begin{cases} \text{Stay current state } \boldsymbol{c}_{t-1} \text{ unless } c_{t-1,i_t} = 1. \\ \text{Change the state from } \boldsymbol{c}_{t-1} \text{ to } \boldsymbol{c}_t \text{ where } c_{t,i} = \mathbf{1}_{[i\in\mathcal{I}]}. \end{cases}$$

Let X_b be the time that the total cost incurred by algorithm A reaches b. Then $\mathrm{cost}_A(\sigma_T) = \max\{b : X_b \leq T\}$ and random variables $Y_b = X_b - X_{b-1}$ are independently and identically distributed. Using the elementary renewal theorem,

$$\lim_{T\to\infty} \frac{\mathbb{E}_D[\mathrm{cost}_A(\sigma_T)]}{T} = \frac{1}{\mathbb{E}_D[Y_b]}.$$

Now we estimate $\mathbb{E}_D[Y_b]$. Y_b is the number of rounds need to collect $d-k$ kind of indices using uniformly generating i_t. When m indices are appeared then the probability of randomly generated next index is new is $(d-m)/d$. Hence the expected number of states between m-th and $(m+1)$-st indices has appeared is $d/(d-m)$. Thus, summing up this to $d-k$,

$$\mathbb{E}_D[Y_b] = \sum_{m=1}^{d-k} \frac{d}{d-m} = d(H_d - H_k).$$

Combining with the well-known fact that $H_d = \Theta(\log d)$, we complete the proof. □

6.5 Conclusion and future work

In this chapter, we proposed the Weighted Marking algorithm for combinatorial MTS problems under the uniform metric space, and proved its competitive ratio is at most $6e \ln n + 9 = O(\log n)$. We showed that, by combining with existing sampling techniques for exponential weights over combinatorial objects, the proposed algorithm runs efficiently for several combinatorial classes, e.g., s-t paths and k-sets.

There are several open problems to investigate. First one is to provide a lower bound of the competitive ratio of the combinatorial MTS. In particular, it still remains open to prove $\Omega(\log d)$ or $\Omega(\log n)$ lower bounds for some combinatorial class of the decision set.

Secondly, it is not known if FPL [43] is applicable for the combinatorial MTS problem. If so, the sampling oracle is no longer necessary and we could efficiently solve MTS problems for more classes of combinatorial objects.

Finally, the hardness of the Sampling $1(C)$ is not known, either. Our conjecture is, it is $\#P$ hard for a specific class.

Chapter 7

Conclusion

In this thesis, we developed algorithms which run efficiently under some structural constraints with theoretical guarantees of its robustness.

In Chapter 3, we considered the online semi-definite programming problem and an FTRL-based algorithm which the log-determinant regularizer. To analyze the performance of the algorithm, we introduced a new notion of strong convexity, namely the strong convexity with respect to the loss space, and gave a general method of deriving a regret bound of the FTRL with a regularizer that is strongly convex in the sense above. We then proved that the log-determinant is actually strongly convex and thus obtained a regret bound of the algorithm. Our result implies that the log-determinant regularization works well when the loss matrices are sparse. We also applied the algorithm to various online matrix prediction problems [38], and showed that our algorithm realizes optimal regret bounds.

In Chapter 4, we considered the binary matrix completion problem in the online prediction model and the statistical learning model. We reduced the online binary matrix completion into an online semi-definite programming problem, and then applied the log-determinant regularization proposed in Chapter 3. The proposed algorithm achieves the optimal mistake bound. We applied our online algorithm to the statistical learning setting by the standard online-batch conversion technique, and gave a generalization error bound. Our mistake bound and generalization error bound depend on the margin loss, which, in some sense, measures the easiness of the given sample. The generalization error bound obtained is somewhat surprising, since it implies that our algorithm is competetive with that of the SVM equipped with the optimal feature map, despite the fact that our algorithm is not given the optimal feature map.

In Chapter 5, we studied some matrix classes defined by the norm constrained matrix factorization. We used these classes as the hypothesis classes of the matrix completion prob-

lem in the statistical learning model, and proved generalization error bounds by estimating the Rademacher complexity of these classes. We also gave matching lower bounds of the Rademacher complexity, and thus our estimations of the Rademacher complexity are tight. We showed by some experiments that our hypothesis classes yield better performance than the class of low-rank matrices.

In Chapter 6, we proposed an algorithm for the combinatorial metrical task systems problem under the uniform metric. Our algorithm is the first efficient algorithm in running time, which runs in time polynomial in the dimension. We proved a competitive ratio of our algorithm, which is optimal for some decision set.

Bibliography

[1] J. Abernethy, P. L. Bartlett, N. Buchbinder, and I. Stanton. A Regularization Approach to Metrical Task Systems. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory (ALT'10)*, volume LNCS 6331, pages 270–284, 2010.

[2] N. Ailon, K. Hatano, and E. Takimoto. Bandit Online Optimization Over the Permutahedron. In *Proceedings of 25th International Conference on Algorithmic Learning Theory(ALT 2014)*, volume 8776 of *LNCS*, pages 215–229, 2014.

[3] P. Alquier, V. Cottet, and G. Lecué. Estimation bounds and sharp oracle inequalities of regularized procedures with Lipschitz loss functions. *ArXiv e-prints*, Feb. 2017.

[4] S. Arora, R. Ge, Y. Halpern, D. M. Mimno, A. Moitra, D. Sontag, Y. Wu, and M. Zhu. A practical algorithm for topic modeling with provable guarantees. In *Proceedings of the 30th International Conference on Machine Learning (ICML'13)*, pages 280–288, 2013.

[5] E. Balkanski and Y. Singer. The sample complexity of optimizing a convex function. In *Proceedings of the 30th Conference on Learning Theory (COLT'17)*, pages 275–301, 2017.

[6] N. Bansal, N. Buchbinder, A. Madry, and J. S. Naor. A polylogarithmic-competitive algorithm for the k-server problem. *Journal of the ACM*, 62(5):40:1–40:49, Nov. 2015.

[7] Y. Bao, H. Fang, and J. Zhang. Topicmf: Simultaneously exploiting ratings and reviews for recommendation. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence (AAAI'14)*, pages 2–8, 2014.

[8] Y. Bartal, A. Blum, C. Burch, and A. Tomkins. A polylog(n)-competitive algorithm for metrical task systems. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing (STOC'97)*, pages 711–719, 1997.

[9] Y. Bartal, B. Bollobás, and M. Mendel. A ramsey-type theorem for metric spaces and its applications for metrical task systems and related problems. In *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS'01)*, pages 396–405, 2001.

[10] D. S. Bernstein. *Matrix Mathematics: Theory, Facts, and Formulas (Second Edition)*. Princeton University Press, Princeton, N.J., Woodstock, 2009.

[11] S. A. Bhaskar and A. Javanmard. 1-bit matrix completion under exact low-rank constraint. In *49th Annual Conference on Information Sciences and Systems (CISS'15)*, pages 1–6, 2015.

[12] D. M. Blei. Introduction to probabilistic topic models. In *In Communications of the ACM*, 2011.

[13] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, Mar. 2003.

[14] A. Blum, S. Chawla, and A. Kalai. Static optimality and dynamic search-optimality in lists and trees. In *Proceedings of the Thirteenth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA'02)*, pages 1–8, 2002.

[15] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, New York, NY, USA, 1998.

[16] A. Borodin, N. Linial, and M. E. Saks. An optimal on-line algorithm for metrical task system. *Journal of the ACM*, 39(4):745–763, Oct. 1992.

[17] N. Buchbinder, S. Chen, J. S. Naor, and O. Shamir. Unified algorithms for online learning and competitive analysis. *Mathematics of Operations Research*, 41(2):612–625, 2016.

[18] T. Cai and W. Zhou. A max-norm constrained minimization approach to 1-bit matrix completion. *Journal of Machine Learning Research*, 14(1):3619–3647, 2013.

[19] N. Cesa-Bianchi and G. Lugosi. Combinaotrial Bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422, 2012.

[20] N. Cesa-Bianchi and O. Shamir. Efficient online learning via randomized rounding. In *Advances in Neural Information Processing Systems 24 (NIPS'11)*, pages 343–351, 2011.

[21] P. Christiano. Online local learning via semidefinite programming. In *Symposium on Theory of Computing (STOC'14)*, pages 468–474, 2014.

[22] T. M. Cover and J. A. Thomas. *Elements of information theory (2. ed.)*. Wiley, 2006.

[23] A. Cutkosky and K. A. Boahen. Online learning without prior information. In *Proceedings of the 30th Conference on Learning Theory (COLT'17)*, pages 643–677, 2017.

[24] M. A. Davenport, Y. Plan, E. van den Berg, and M. Wootters. 1-bit matrix completion. *Information and Inference: A Journal of the IMA*, 3(3):189–223, 2014.

[25] J. V. Davis and I. Dhillon. Differential entropic clustering of multivariate gaussians. In *Advances in Neural Information Processing Systems 19 (NIPS'06)*, pages 337–344, 2007.

[26] J. V. Davis, B. Kulis, P. Jain, S. Sra, and I. S. Dhillon. Information-theoretic metric learning. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, pages 209–216, 2007.

[27] R. Degenne and V. Perchet. Combinatorial semi-bandit with known covariance. In *Advances in Neural Information Processing Systems 29 (NIPS'16)*, pages 2964–2972, 2016.

[28] S. S. Du, Y. Liu, B. Chen, and L. Li. Maxios: Large scale nonnegative matrix factorization for collaborative filtering. In *Neural Information Processing Systems 27 (NIPS'14), workshop on Distributed Machine Learning and Matrix Computations*, 2014.

[29] A. Fiat and M. Mendel. Better algorithms for unfair metrical task systems and applications. *SIAM Journal on Computing*, 32(6):1403–1422, 2003.

[30] S. Forth, P. Hovland, E. Phipps, J. Utke, and A. Walther. *Recent Advances in Algorithmic Differentiation*. Lecture Notes in Computational Science and Engineering. Springer, 2012.

[31] R. Foygel and N. Srebro. Concentration-based guarantees for low-rank matrix reconstruction. In *The 24th Annual Conference on Learning Theory (COLT'11)*, pages 315–340, 2011.

[32] Y. Freund and R. 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.

[33] Q. Gu, J. Zhou, and C. H. Q. Ding. Collaborative filtering: Weighted nonnegative matrix factorization incorporating user and item graphs. In *Proceedings of the 2010 SIAM International Conference on Data Mining (SDM'10)*, pages 199–210, 2010.

[34] A. Gupta, K. Talwar, and U. Wieder. Changing bases: Multistage optimization for matroids and matchings. In *41st International Colloquium on Automata, Languages, and Programming (ICALP'14), Part I*, pages 563–575, 2014.

[35] F. M. Harper and J. A. Konstan. The movielens datasets: History and context. *ACM Transactions on Interactive Intelligent Systems*, 5(4):19:1–19:19, Dec. 2015.

[36] E. Hazan. The convex optimization approach to regret minimization. In Suvrit Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*, chapter 10, pages 287–304. MIT Press, 2011.

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

[38] E. Hazan, S. Kale, and S. Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. In *The 25th Annual Conference on Learning Theory (COLT'12)*, pages 38.1–38.13, 2012.

[39] M. Herbster, S. Pasteris, and M. Pontil. Mistake bounds for binary matrix completion. In *Advances in Neural Information Processing Systems 29 (NIPS'16)*, pages 3954–3962, 2016.

[40] S. Irani and S. Seiden. Randomized algorithms for metrical task systems. *Theoretical Computer Science*, 194(12):163 – 182, 1998.

[41] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM*, 51:671–697, 2004.

[42] C. Jin, S. M. Kakade, and P. Netrapalli. Provable efficient online matrix completion via non-convex stochastic gradient descent. In *Advances in Neural Information Processing Systems 29 (NIPS'16)*, pages 4520–4528, 2016.

[43] A. Kalai and S. Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[44] S. Kale, C. Lee, and D. Pál. Hardness of online sleeping combinatorial optimization problems. In *Advances in Neural Information Processing Systems 29 (NIPS'16)*, pages 2181–2189, 2016.

[45] T. Kamishima. Nantonac collaborative filtering: Recommendation based on order responses. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'03)*, pages 583–588, 2003.

[46] O. Klopp. Noisy low-rank matrix completion with general sampling distribution. *Bernoulli*, 20(1):282–303, 2014.

[47] V. Koltchinskii, K. Lounici, and A. B. Tsybakov. Nuclear-norm penalization and optimal rates for noisy low-rank matrix completion. *The Annals of Statistics*, 39(5):2302–2329, 2011.

[48] Y. Koren, R. Bell, and C. Volinsky. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, Aug. 2009.

[49] E. Koutsoupias and C. H. Papadimitriou. On the k-server conjecture. *Journal of the ACM*, 42(5):971–983, Sept. 1995.

[50] B. Kulis, M. A. Sustik, and I. S. Dhillon. Low-rank kernel learning with bregman matrix divergences. *Journal of Machine Learning Research*, 10:341–376, June 2009.

[51] D. D. Lee and H. S. Seung. Learning the parts of objects by nonnegative matrix factorization. *Nature*, 401:788–791, 1999.

[52] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik. Data poisoning attacks on factorization-based collaborative filtering. In *Advances in Neural Information Processing Systems 29 (NIPS'16)*, pages 1885–1893, 2016.

[53] N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman. Complexity measures of sign matrices. *Combinatorica*, 27(4):439–463, 2007.

[54] X. Luo, M. Zhou, Y. Xia, and Q. Zhu. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. *IEEE Transactions on Industrial Informatics*, 10(2):1273–1284, 2014.

[55] J. R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications in Statistics and Econometrics*. Wiley series in probability and mathematical statistics. Wiley, 1999.

[56] R. Mazumder, T. Hastie, and R. Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *Journal of Machine Learning Research*, 11:2287–2322, Aug. 2010.

[57] M. Mohri, A. Rostamizadeh, and A. Talwalkar. *Foundations of Machine Learning*. Adaptive computation and machine learning series. MIT Press, 2012.

[58] K. Moridomi, K. Hatano, and E. Takimoto. Online linear optimization with the log-determinant regularizer. *ArXiv e-prints*, Oct. 2017.

[59] K. Moridomi, K. Hatano, E. Takimoto, and K. Tsuda. Online matrix prediction for sparse loss matrices. In *Proceedings of the Sixth Asian Conference on Machine Learning (ACML'14)*, 2014.

[60] T. Nakazono, K. Moridomi, K. Hatano, and E. Takimoto. A combinatorial metrical task system problem under the uniform metric. In *Algorithmic Learning Theory - 27th International Conference (ALT'16)*, pages 276–287, 2016.

[61] N. Natarajan and P. Jain. Regret bounds for non-decomposable metrics with missing labels. In *Advances in Neural Information Processing Systems 29 (NIPS 2016)*, pages 2874–2882, 2016.

[62] Y. Nesterov. *Introductory lectures on convex optimization : A basic course*, volume 87 of *Applied optimization*. Springer US, 2004.

[63] N. Parikh and S. Boyd. Proximal algorithms. *Foundations and Trends in Machine Learning*, 1(3):127–239, Jan. 2014.

[64] K. B. Petersen and M. S. Pedersen. The matrix cookbook, Nov. 2012. Version 20121115.

[65] A. Rakhlin, J. Abernethy, A. Agarwal, P. Bartlett, E. Hazan, and A. Tewari. Lecture notes on online learning draft, 2009.

[66] N. Rao, H. Yu, P. K. Ravikumar, and I. S. Dhillon. Collaborative filtering with graph information: Consistency and scalable methods. In *Advances in Neural Information Processing Systems 28 (NIPS'15)*, pages 2107–2115, 2015.

[67] P. Ravikumar, M. J. Wainwright, G. Raskutti, and B. Yu. High-dimensional covariance estimation by minimizing ℓ_1-penalized log-determinant divergence. *Electronic Journal of Statistics*, 5:935–980, 2011.

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

[69] O. Shamir and S. Shalev-Shwartz. Collaborative filtering with the trace norm: Learning, bounding, and transducing. In *The 24th Annual Conference on Learning Theory (COLT'11)*, pages 661–678, 2011.

[70] N. Srebro, N. Alon, and T. S. Jaakkola. Generalization error bounds for collaborative prediction with low-rank matrices. In *Advances In Neural Information Processing Systems 17 (NIPS'04)*, pages 1321–1328, 2004.

[71] N. Srebro, J. D. M. Rennie, and T. S. Jaakola. Maximum-margin matrix factorization. In *Advances in Neural Information Processing Systems 17 (NIPS'04)*, pages 1329–1336, 2005.

[72] N. Srebro and A. Shraibman. Rank, trace-norm and max-norm. In *18th Annual Conference on Learning Theory (COLT'05)*, pages 545–560, 2005.

[73] E. Takimoto and M. K. Warmuth. Path kernels and multiplicative updates. *Journal of Machine Learning Research*, 4(5):773–818, 2004.

[74] K. Tsuda, G. Rätsch, and M. K. Warmuth. Matrix exponentiated gradient updates for on-line learning and Bregman projection. *Journal of Machine Learning Research*, 6:995–1018, June 2005.

[75] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, Nov. 1984.

[76] C. Wang and D. M. Blei. Collaborative topic modeling for recommending scientific articles. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'11)*, pages 448–456, 2011.

[77] M. K. Warmuth. Winnowing subspaces. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, pages 999–1006, 2007.

[78] A. Xu and M. Raginsky. Information-theoretic analysis of generalization capability of learning algorithms. In *Advances in Neural Information Processing Systems 30 (NIPS'17)*, pages 2521–2530, 2017.

[79] S. Zhang, W. Wang, J. Ford, and F. Makedon. Learning from incomplete ratings using non-negative matrix factorization. In *Proceedings of the 2006 SIAM International Conference on Data Mining (SDM'06)*, pages 549–553, 2006.

[80] Y. Zheng, B. Tang, W. Ding, and H. Zhou. A neural autoregressive approach to collaborative filtering. In *Proceedings of The 33rd International Conference on Machine Learning (ICML'16)*, pages 764–773, 2016.