Online Matrix Completion with Side Information

Written by Fai Yu Lisa Tse
Supervisor: Mark Herbster

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of
University College London

Department of Computer Science
UCL
August 13, 2023

I, Fai Yu Lisa Tse, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

This thesis considers the problem of binary matrix completion with side information in the online setting and the applications thereof. The side information provides additional information on the rows and columns and can yield improved results compared to when such information is not available. We present efficient and general algorithms in transductive and inductive models. The performance guarantees that we prove are with respect to the matrix complexity measures of the max-norm and the margin complexity. We apply our bounds to the hypothesis class of biclustered matrices. Such matrices can be permuted through the rows and columns into homogeneous latent blocks. This class is a natural choice for our problem since the margin complexity and max-norm of these matrices have an upper bound that is easy to interpret in terms of the latent dimensions. We also apply our algorithms to a novel online multitask setting with RKHS hypothesis classes. In this setting, each task is partitioned in a sequence of segments, where a hypothesis is associated with each segment. Our algorithms are designed to exploit the scenario where the number of associated hypotheses is much smaller than the number of segments. We prove performance guarantees that hold for any segmentation of the tasks and any association of hypotheses to the segments. In the single-task setting, this is analogous to switching with long-term memory in the sense of [Bousquet and Warmuth; 2003].

Impact Statement

Matrix completion is a mathematical problem that has applications ranging from e-commerce to computer vision. Its most notable application area is possibly that of recommendation systems, popularized by the Netflix challenge. As the demand for such applications rise, it is natural that algorithms become increasingly better adapted to the different needs for each application. In this thesis, we cater for one such need; we develop online algorithms which are particularly suited for the case where additional information is given about the rows and the columns of the matrix. There are direct applications for the problem that we solve, such as movie recommendation systems where we have user demographics and movie metadata.

However, we also identify another less evident application: multitask learning with memory. This setting is applicable when multiple learning agents collaborate in changing but recurring environments. Examples include drones working together to identify environmental waste, and servers attempting to identify suspicious activity in networks. Aside from having useful applications, the setting is novel which could spark further ideas in the research community.

Regardless of the exact application, the algorithms and bounds that we develop are contributions to the field of theoretical machine learning. With the rise of machine learning and its ubiquity in many domains, a good theoretical understanding remains important to avoid any pitfalls. For instance, our algorithm cannot fare too badly, even when presented with adversarial data. This characteristic makes the algorithm more robust against adversarial attacks.

UCL Research Paper Declaration Form

referencing the doctoral candidate's own published work(s)

1. **For a research manuscript that has already been published** (if not yet published, please skip to section 2)**:**

   (a) **What is the title of the manuscript?**

   i. Online Matrix Completion with Side Information
   ii. Online Multitask Learning with Long-Term Memory

   (b) **Please include a link to or doi for the work:** [Link i], [Link ii].

   (c) **Where was the work published?**

   Both were published at NeuRIPS 2020.

   (d) **Who published the work?** Curran Associates.

   (e) **When was the work published?** 2020.

   (f) **List the manuscript's authors in the order they appear on the publication:** Mark Herbster, Stephen Pasteris, Lisa Tse

   (g) **Was the work peer reviewed?** Yes.

   (h) **Have you retained the copyright?** Yes.

   If 'No', please seek permission from the relevant publisher and check the box next to the below statement:

   □ *I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.*

   (i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or doi**

   Yes: [ArXiv Link i], [ArXiv Link ii].

2. **For a research manuscript prepared for publication but that has not yet been published** (if already published, please skip to section 3)**:**

   (a) **What is the current title of the manuscript?**
   (b) **Has the manuscript been uploaded to a preprint server 'e.g. medRxiv'?**

   If 'Yes', please please give a link or doi:
   (c) **Where is the work intended to be published?**
   (d) **List the manuscript's authors in the intended authorship order:**
   (e) **Stage of publication:**

3. **For multi-authored work, please give a statement of contribution covering all authors** (if single-author, please skip to section 4)**:**

   For both works, the research and the paper-writing was a joint effort between Mark, Stephen and I. The experiments were my own work.

4. **In which chapter(s) of your thesis can this material be found?**

   Chapters 3, 4 and 5.

e-Signatures confirming that the information above is accurate (this form should be co-signed by the supervisor/ senior author unless this is not appropriate, e.g. if the paper was a single-author work)**:**

Candidate: Lisa Tse

Date: 2 March 2022

Supervisor signature: Mark Herbster

Date: 2 March 2022

Acknowledgements

First and foremost, I would like to thank Mark Herbster, who has been an extremely supportive supervisor and a source of inspiration. His insight to mathematical problems continues to amaze me to this day, and his unwavering support has made my life as a PhD student a breeze instead of what could've been an endless rabbit hole. Never one to apply pressure on his students, yet prepared to help at all times, Mark's approach to supervision struck the perfect balance between laissez-faire and constant involvement. Without him, I may well be stuck on Chapter 1 of this thesis.

Many thanks are also due to Stephen Pasteris. The clarity of his mathematical proofs is a quality that I aspire to reproduce one day. On top of that, his technical expertise and cheerful attitude made him a great researcher and person to work with.

Academic research relies on funding, and I'm grateful to my funding body, the Engineering and Physical Sciences Research Council [grant number EP/L015242/1], without whose generous funding I would not have been able to conduct the research required for the thesis. Thanks also to the Quantum Technologies CDT for giving me the opportunity to receive this grant, and to the directors of the CDT, Dan Browne and Paul Warburton.

I'd also like to thank James Robinson and Shota Saito for helpful discussions; to the anonymous reviewers who helped to improve our work and to the other students in my CDT cohort for organizing lunches and giving me some semblance of a social life in the isolation that is so common in theoretical PhDs. As for the hardware of the experiments, I'm grateful to UCL Myriad High Performance Computing

Facility (Myriad@UCL), as well as their associated support services.

Finally, I am indebted to my family and friends for their love. My parents and grandparents have supported me in whichever endeavour I wanted to pursue throughout my life, and their confidence in my abilities has given me the courage to walk down the path that I did. Keeping a family afloat as immigrants is no easy feat, and they had to suffer through more than most bringing me up, but they still did their best to make sure that I wasn't deprived of anything compared to my peers. I'd also like to thank my partner Shyam Amrith, who accompanied me for my entire PhD journey. He provided me with emotional support, guiding me with a clarity reminiscent of Ariadne's thread when I felt lost in a maze of doctoral confusion.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The matrix completion problem arises in numerous application areas, including the well-known "Netflix challenge" [1] and more generally, collaborative filtering [2]. Crudely speaking, the aim is to recover the missing entries of a given matrix. When applied to the Netflix challenge, the rows of the matrix represent the users of the platform while the columns represent the movies that can be recommended to these users. Each entry of the matrix would then be a numerical value which indicates whether the corresponding user likes the movie. The value could be continuous or categorical, commensurate with the strength of preference, or it could be binary, where a '1' encodes a positive response and a '0' indicates a negative response. This problem is not fully defined in the absence of further constraints. In collaborative filtering, it is further assumed that the matrix is of low rank, which assumes clusters of similar users and movies. Since the rank is non-convex, the nuclear or trace norm is often used as a convex proxy for the rank. However, the use of the max norm as a proxy has also been gaining traction [3] and in this thesis, we have bounds in terms of the margin complexity and the max norm.

As for many learning tasks, we can formulate both batch and online learning variants of this problem. In the batch setting, all the known entries are accessible to the learner from the onset, and the task is to predict all the missing entries. The online learning setting instead proceeds in trials, where on each trial, only a single matrix entry is considered. For this entry, the learner gives a prediction, receives

feedback and then incurs a loss. In this thesis, we will focus on the online variant of the problem.

In many situations, we may have access to additional information on the rows and columns, the *side information*. Reverting back to the Netflix example, this information could be derived from demographic data from the users and movie metadata. The side information is hoped to aid the learner, resulting in improved performance guarantees. We consider side information with kernel functions which output the similarity between the users (or equivalently the movies). These functions could be derived from graph Laplacians, or be standard kernels such as the Gaussian kernel.

1.1 Problem Statement

The general problem that we consider is binary matrix completion with side information in the online setting. First we introduce the setting without side information. On each trial $t = 1, \ldots, T$:

1. the learner is queried by the *environment* to predict matrix entry (i_t, j_t)
2. the learner predicts a label $\hat{y}_t \in \{-1, 1\}$
3. the learner receives a label $y_t \in \{-1, 1\}$ from the environment and
4. a mistake is incurred if $y_t \neq \hat{y}_t$.

The aim is to minimize the number of mistakes. With the introduction of side information, the learner not only receives the entry (i_t, j_t) at each trial but also additional information on the rows and columns. We consider both *transductive* and *inductive* models. In the former model, we are given the side information of all the rows and columns completely in advance as a pair of positive definite matrices. One of these informs the similarity between the rows, and the other for the columns. The limitation with this model is that we cannot introduce new rows or columns during the learning process. The inductive model bypasses this limitation by only giving two kernel functions in advance, which again inform the similarity between the rows and the columns respectively. As learning proceeds, we are asked to predict elements

from the domains of the row and column kernel functions. The mapping of the domain of the kernel functions to the rows and columns of the matrix is thus created on the go. When the domains of the kernel functions are infinite-dimensional, there are no restrictions on the number of rows and columns.

We will consider two forms of performance guarantees. The first is an upper bound on the number of mistakes as a function of a comparator matrix $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$, under the assumption that $\text{sign}(U_{i_t j_t}) = y_t$ for all $t \in [T]$. This assumption is also denoted the realizability assumption, and is violated for instance when the received labels y_t are noisy and hence do not exactly correspond to the entries in $\boldsymbol{U}$. The second is an upper bound on the *expected c-regret*, $\sum_{t=1}^{T} \mathbb{E}[y_t \neq \hat{y}_t] - c(\boldsymbol{U}) \sum_{t=1}^{T} [y_t \neq U_{i_t j_t}]$, where the expectation is with respect to the learner's internal randomization. Contrary to the mistake bound, such a guarantee is indicative of a robustness to noise. As is standard in online learning, the guarantees that we provide do not require probabilistic assumptions on how the environment generates the instances or their labels; in fact this process may even be adversarial. The only restriction on the environment for the regret bound to hold is that the label y_t cannot depend on the learner's prediction $\hat{y}_t$. In the case of the mistake bound, this requirement is not even necessary.

In this thesis, we consider variants of the matrix exponentiated gradient (MEG) algorithm [4] and the matrix gradient descent (MGD) algorithm [5, 6]. In particular, we will present efficient, polynomial-time algorithms that perform online matrix completion with side information in both transductive and inductive models, and provide meaningful performance guarantees. The MEG-based algorithms offer superior performance guarantees at the expense of a higher run-time. We also discuss the applications on matrices with a latent block structure and online multitask learning with long-term memory. The contents of the thesis are based on the following publications:

1. Mark Herbster, Stephen Pasteris, and Lisa Tse. Online matrix completion with side information. *Neural Information Processing Systems*, (33), 2020

2. Mark Herbster, Stephen Pasteris, and Lisa Tse. Online multitask learning

with long-term memory. *Neural Information Processing Systems*, (33), 2020

1.2 Thesis Structure

Chapter 2: Background

This chapter includes short introductions and brief literature reviews on the key areas of the thesis. Since all the algorithms presented in the thesis consider the online setting, we introduce the general field of online learning along with some representative works. We also review other works that perform matrix completion in the online setting. Finally, we discuss matrix completion algorithms in the batch setting, as this setting has an extensive body of work that consider side information.

Chapter 3: General Algorithms and Bounds

In Chapter 3, we give our base transductive and inductive algorithms for online binary matrix completion with side information. We prove regret bounds, in addition to mistake bounds for a MEG-based algorithm and a MGD-based algorithm in Theorems 1 and 4 respectively. The MEG-based algorithm has improved mistake bounds at the expense of a higher time complexity. We also prove that the transductive and inductive algorithms are prediction-equivalent, so that the bounds for the transductive algorithms hold for their inductive counterparts. The bounds hold for general comparator matrices and are written in terms of the max-norm and margin complexity. The chapter presents the results in [7] with the minor modification that the chapter gives performance guarantees for real-valued instead of binary comparator matrices. This is useful in the application of the Gaussian kernel on biclustered matrices (Section 4.5.2) and the analysis in the main theorem of Chapter 5. While most of the chapter is written in collaboration with Mark Herbster and Stephen Pasteris, Sections 3.2.2 and 3.4 are my own work.

Chapter 4: Latent Block Structure

In Chapter 4, we give an interpretation of the bounds for the hypothesis class of biclustered matrices in Theorem 57. These matrices have a latent block structure and we consider various forms of side information, such as graph side information in the

transductive setting and vectors in $\mathfrak{R}^d$ in the inductive setting. The chapter is based on the contents of [7], which is written in collaboration with Mark Herbster and Stephen Pasteris. Sections 4.5.2, 4.5.3 and 4.6.2 are extensions to the publication and my own work.

Chapter 5: Online Multitask Learning with Long-Term Memory

In Chapter 5, we apply our inductive algorithm to solve the problem of multitask learning with long-term memory on RKHS hypothesis classes. In this treatment, an alternative interpretation is used for Theorem 57 through Theorem 71. This chapter extends the contents of [8], so that all results hold for functions with a bounded, continuous range instead of the binary range in [8]. The original publication is written in collaboration with Mark Herbster and Stephen Pasteris, but the extension to continuous RKHS functions and Section 5.4 are my own work.

1.3 Notation

For any positive integer m, we define $[m] := \{1, 2, \ldots, m\}$. For any predicate $[\text{PRED}] := 1$ if PRED is true and equals 0 otherwise, and $[x]_+ := x[x > 0]$. We define the hinge loss at margin γ as $\mathcal{L}^{\gamma}_{\text{hi}}(y_1, y_2) = \frac{1}{\gamma}[\gamma - y_1 y_2]_+$, and the zero-one loss as $\mathcal{L}_{01}(y_1, y_2) := [y_1 \neq y_2]$.

We denote the inner product of vectors $\boldsymbol{x}, \boldsymbol{w} \in \mathfrak{R}^n$ as both $\langle \boldsymbol{x}, \boldsymbol{w} \rangle = \boldsymbol{x} \cdot \boldsymbol{w} = \sum_{i=1}^{n} x_i w_i$, and the norm as $\|\boldsymbol{w}\| = \sqrt{\langle \boldsymbol{w}, \boldsymbol{w} \rangle}$. The *ith* coordinate m-dimensional vector is denoted $\boldsymbol{e}^i_m := ([j = i])_{j \in [m]}$; we will often abbreviate the notation and use $\boldsymbol{e}^i$ on the assumption that the dimensionality of the space may be inferred. More generally, the xth-coordinate vector is denoted $\boldsymbol{e}^x_X := ([x = z])_{z \in X}$; we commonly abbreviate this to $\boldsymbol{e}^x$.

For vectors $\boldsymbol{p} \in \mathfrak{R}^m$ and $\boldsymbol{q} \in \mathfrak{R}^n$ we define $[\boldsymbol{p}; \boldsymbol{q}] \in \mathfrak{R}^{m+n}$ to be the concatenation of $\boldsymbol{p}$ and $\boldsymbol{q}$, which we regard as a column vector. Hence $[\boldsymbol{p}; \boldsymbol{q}]^\top [\bar{\boldsymbol{p}}; \bar{\boldsymbol{q}}] = \boldsymbol{p}^\top \bar{\boldsymbol{p}} + \boldsymbol{q}^\top \bar{\boldsymbol{q}}$. We let $\mathfrak{R}^{m \times n}$ be the set of all $m \times n$ real-valued matrices. If $\boldsymbol{X} \in \mathfrak{R}^{m \times n}$ then $\boldsymbol{X}_i$ denotes the i-th n-dimensional row vector and the $(i, j)^{th}$ entry of $\boldsymbol{X}$ is X_{ij}. We define $\boldsymbol{X}^+$ and $\boldsymbol{X}^\top$ to be its pseudoinverse and transpose, respectively. The trace norm of a matrix $\boldsymbol{X} \in \mathfrak{R}^{m \times n}$ is $\|\boldsymbol{X}\|_1 = \text{tr}(\sqrt{\boldsymbol{X}^\top \boldsymbol{X}})$, where $\sqrt{\cdot}$ indicates the

unique positive square root of a positive semi-definite matrix, and $\mathrm{tr}(\cdot)$ denotes the trace of a square matrix. This is given by $\mathrm{tr}(\boldsymbol{Y}) = \sum_{i=1}^{n} Y_{ii}$ for $\boldsymbol{Y} \in \mathfrak{R}^{n\times n}$. The $m \times m$ identity matrix is denoted $\boldsymbol{I}^m$. In addition, we define $\boldsymbol{S}^m$ to be the set of $m \times m$ symmetric matrices and let $\boldsymbol{S}^m_+$ and $\boldsymbol{S}^m_{++}$ be the subset of positive semidefinite and strictly positive definite matrices respectively. Recall that the set of symmetric matrices $\boldsymbol{S}^m_+$ has the following partial ordering: for every $\boldsymbol{M}, \boldsymbol{N} \in \boldsymbol{S}^m_+$, we say that $\boldsymbol{M} \preceq \boldsymbol{N}$ if and only if $\boldsymbol{N} - \boldsymbol{M} \in \boldsymbol{S}^m_+$. We also define the squared radius of $\boldsymbol{M} \in \boldsymbol{S}^m_+$ as $\mathcal{R}_{\boldsymbol{M}} := \max_{i\in[m]} M^+_{ii}$, where the nomenclature comes from an identical term that arises in the kernel perceptron bound as per Novikoff's Theorem [76].

For every matrix $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$, we define $\mathrm{SP}(\boldsymbol{U}) = \{\boldsymbol{V} \in \mathfrak{R}^{m\times n} : \forall_{ij} V_{ij}U_{ij} > 0\}$, the set of matrices which are sign consistent with $\boldsymbol{U}$. We also define $\mathrm{SP}^1(\boldsymbol{U}) = \{\boldsymbol{V} \in \mathfrak{R}^{m\times n} : \forall_{ij} V_{ij}\,\mathrm{sign}(U_{ij}) \geq 1\}$, that is the set of matrices which are sign consistent with $\boldsymbol{U}$ with a margin of at least one.

The max-norm (or γ_2 norm [9]) of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$ is defined by

$$\|\boldsymbol{U}\|_{\max} := \min_{\boldsymbol{P}\boldsymbol{Q}^\top = \boldsymbol{U}} \left\{ \max_{1\leq i\leq m} \|\boldsymbol{P}_i\| \times \max_{1\leq j\leq n} \left\|\boldsymbol{Q}_j\right\| \right\}, \tag{1.1}$$

where the minimum is over all matrices $\boldsymbol{P} \in \mathfrak{R}^{m\times d}$, $\boldsymbol{Q} \in \mathfrak{R}^{n\times d}$ and every integer d. For more intuition, we recall that the trace norm of a matrix $\boldsymbol{U}$, given by the sum of the singular values, can be written as $\|\boldsymbol{U}\|_{\mathrm{tr}} = \min_{\boldsymbol{P}\boldsymbol{Q}^\top=\boldsymbol{U}} \left\{ \sqrt{\sum_{1\leq i\leq m} \|\boldsymbol{P}_i\|^2} \sqrt{\sum_{1\leq j\leq n} \left\|\boldsymbol{Q}_j\right\|^2} \right\}$. Hence, the trace norm constrains the sum instead of the maximum over the row norms of $\boldsymbol{P}$ and $\boldsymbol{Q}$, giving a more uniform constraint (for more details, see [53]). A related notion that has been used to study binary matrices is the *margin complexity*. The margin complexity of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$ is

$$\mathrm{mc}(\boldsymbol{U}) := \min_{\boldsymbol{V}\in \mathrm{SP}^1(\boldsymbol{U})} \|\boldsymbol{V}\|_{\max} = \min_{\boldsymbol{P}\boldsymbol{Q}^\top \in \mathrm{SP}(\boldsymbol{U})} \max_{ij} \frac{\|\boldsymbol{P}_i\| \left\|\boldsymbol{Q}_j\right\|}{|\langle \boldsymbol{P}_i, \boldsymbol{Q}_j\rangle|}. \tag{1.2}$$

Observe that for $\boldsymbol{U} \in \{-1, 1\}^{m\times n}$, $1 \leq \mathrm{mc}(\boldsymbol{U}) \leq \|\boldsymbol{U}\|_{\max} \leq \min(\sqrt{m}, \sqrt{n})$, where the lower bound follows from the right hand side of (1.2) and the upper bound follows since we may decompose $\boldsymbol{U} = \boldsymbol{U}\boldsymbol{I}^n$ or as $\boldsymbol{U} = \boldsymbol{I}^m\boldsymbol{U}$. Note there may be

a large gap between the margin complexity and the max-norm. In [9] a matrix in $\boldsymbol{U} \in \{-1,1\}^{n\times n}$ was given such that $\mathrm{mc}(\boldsymbol{U}) = \log n$ and $\|\boldsymbol{U}\|_{\max} = \Theta(\sqrt{n}/\log n)$. We denote the classes of $m \times d$ *row-normalized* and *block expansion* matrices as $\mathcal{N}^{m,d} := \{\hat{\boldsymbol{P}} \subset \Re^{m\times d} : \left\|\hat{\boldsymbol{P}}_i\right\| = 1, i \in [m]\}$ and $\mathcal{B}^{m,d} := \{\boldsymbol{R} \subset \{0,1\}^{m\times d} : \|\boldsymbol{R}_i\| = 1, i \in [m], \mathrm{rank}(\boldsymbol{R}) = d\}$, respectively. Block expansion matrices may be seen as a generalization of permutation matrices, additionally duplicating rows (columns) by left (right) multiplication. We define the *quasi-dimension* of a matrix $\boldsymbol{U} \in \Re^{m\times n}$ with respect to $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ at margin γ as

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma\boldsymbol{U}} \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M}\hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N}\hat{\boldsymbol{Q}}\right), \tag{1.3}$$

where the infimum is over all row-normalized matrices $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ and every integer d. If the infimum does not exist then $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := +\infty$. We also define the *quasi-area* of a matrix $\boldsymbol{U} \in \Re^{m\times n}$ with respect to $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ at margin γ as

$$\mathcal{A}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma\boldsymbol{U}} \mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M}\hat{\boldsymbol{P}}\right) \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N}\hat{\boldsymbol{Q}}\right), \tag{1.4}$$

where the infimum is $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ and every integer d. If the infimum does not exist then $\mathcal{A}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := +\infty$. Note that for both the quasi-dimension and the quasi-area, the infimum exists iff $\|\boldsymbol{U}\|_{\max} \le 1/\gamma$. Finally note that $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) = m + n$ and $\mathcal{A}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) = mn$ if $\|\boldsymbol{U}\|_{\max} \le 1/\gamma$, $\boldsymbol{M} = \boldsymbol{I}^m$ and $\boldsymbol{N} = \boldsymbol{I}^n$.

We now introduce notation specific to the graph setting. Let $\mathcal{G}$ be an m-vertex connected, undirected graph with positive weights. The Laplacian $\boldsymbol{L}$ of $\mathcal{G}$ is defined as $\boldsymbol{D} - \boldsymbol{A}$, where $\boldsymbol{D}$ is the $m \times m$ degree matrix and $\boldsymbol{A}$ is the $m \times m$ adjacency matrix. Observe that as $\mathcal{G}$ is connected, $\boldsymbol{L}$ is a rank $m-1$ matrix with $\mathbf{1}$ in its null space. From $\boldsymbol{L}$ we define the (strictly) positive definite *PDLaplacian* $\boldsymbol{L}^\circ := \boldsymbol{L} + \left(\frac{1}{m}\right)\left(\frac{1}{m}\right)^\top \mathcal{R}_{\boldsymbol{L}}^{-1}$. Observe that if $\boldsymbol{u} \in [-1,1]^m$ then $(\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u})\mathcal{R}_{\boldsymbol{L}^\circ} \le 2(\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u}\,\mathcal{R}_{\boldsymbol{L}} + 1)$, and similarly, $(\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u})\mathcal{R}_{\boldsymbol{L}} \le \frac{1}{2}(\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u})\mathcal{R}_{\boldsymbol{L}^\circ}$ (see [10] for details of this construction).

A function $K : \mathcal{X} \times \mathcal{X} \to \Re$ is a strictly positive definite (SPD) kernel iff for every finite $X \subseteq \mathcal{X}$ the matrix $K(x,x')_{x,x'\in X}$ is symmetric and strictly positive definite, for example, the Gaussian kernel.

Chapter 2

Background

For the reader's convenience, we present short introductions on the main subject areas of this thesis. In Subsection 2.1, we review some of the classical works in online learning and their derivatives, while positioning our algorithms within this vast landscape of related works. This is by no means an exhaustive review of the field, and we encourage the interested reader to peruse references such as [11, 12]. Since we present algorithms for online matrix completion, we proceed by covering the research in this area in Subsection 2.2.1. The works on matrix completion in the batch setting are then reviewed in Subsection 2.2.2, which includes more works on incorporating side information into the problem.

2.1 Online Learning

Online learning is an alternative paradigm to the batch supervised learning setting. Instead of being given all the training data at the start of the learning process, the premise of online learning is that the learner receives the training examples sequentially over T trials. On each trial t, the learner observes an instance $\boldsymbol{x}_t$, gives the prediction $\hat{y}_t$, before the true label y_t is revealed and a loss $\mathcal{L}(\hat{y}_t, y_t)$ is incurred. The goal of the learner is then to minimize the cumulative loss $\sum_{t\in[T]} \mathcal{L}(\hat{y}_t, y_t)$. To prove performance guarantees from here, it is possible to take a probabilistic perspective and assume that the data sequence is generated through an underlying probabilistic process, perhaps drawn i.i.d. from an unknown distribution. However, we will

focus on the more dominant approach in online learning where no such assumptions are made. In fact, in the worst case the data sequence can even be adversarial. Given that a potential adversary has free rein over the data generation process, the performance guarantees that can be given must be restricted in some way. A common guarantee is an upper bound on the *regret*, so that the learner is not expected to perform well in absolute terms, but only relative to a hypothesis class $\mathcal{H}$:

$$\sum_{t\in[T]} \mathcal{L}(\hat{y}_t, y_t) - \min_{h\in\mathcal{H}} \sum_{t\in[T]} \mathcal{L}(h(\boldsymbol{x}_t), y_t)$$

It quantifies the regret that we "feel" for not having predicted with the optimal hypothesis $h^* = \text{argmin}_{h\in\mathcal{H}} \sum_{t\in[T]} \mathcal{L}(h(\boldsymbol{x}_t), y_t)$. The objective is to have an upper bound on the regret that is sublinear in T, so that in the limit where $T \to \infty$, the regret tends to zero. If we further assume that $\min_{h\in\mathcal{H}} \sum_{t\in[T]} \mathcal{L}(h(\boldsymbol{x}_t), y_t) = 0$, also termed the *realizability* assumption, we can obtain an arguably more intuitive performance guarantee: an upper bound on the cumulative loss of the algorithm.

Linear Classification

In the case where the predictions and labels are both binary, i.e., $\hat{y}_t \in \{-1, 1\}$ and $y_t \in \{-1, 1\}$, we have the online binary classification problem. The natural loss to use is the zero-one loss, given by $\mathcal{L}_{01}(y_t, \hat{y}_t) = [y_t \neq \hat{y}_t]$, which simply counts the number of mistakes. If we assume that the problem is realizable, then we can give performance guarantees in terms of an upper bound on the number of mistakes. One of the earliest online classification algorithms is Rosenblatt's perceptron [5, 6] for learning binary classifiers with instances $\boldsymbol{x}_t \in \mathfrak{R}^n$. Assuming a linear hypothesis, it maintains a weight vector $\boldsymbol{w}_t$ that initializes as $\boldsymbol{w}_1 = \boldsymbol{0}$, and predicts according to $\hat{y}_t = \text{sign}(\langle \boldsymbol{w}_t \boldsymbol{x}_t \rangle)$. After receiving the binary label, the weight vector is updated as follows:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + y_t \boldsymbol{x}_t.$$

Being one of the earliest learning algorithms, this work has left new research directions to sprout and bloom in its wake, with a prime example being the emergence of neural networks [13]. The perceptron algorithm can be kernelized and descriptions

thereof can be found in [14, 15] amongst others. Another early online classification algorithm is the Winnow algorithm [16]. The setting considered by the original algorithm assumes instances $\boldsymbol{x}_t \in \{0,1\}^n$, binary predictions and labels. Similar to the perceptron algorithm, it maintains a weight vector which it updates. However, unlike the additive updates of the perceptron, it has multiplicative updates:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t \exp(-\eta y_t \boldsymbol{x}_t),$$

where η is typically set to 2. The Winnow algorithm excels in particular on the problem of learning k-literal disjunctions. For this setting, Winnow achieves a $O(k \log(n))$ mistake bound, whereas the perceptron would only achieve a $O(kn)$ mistake bound.

Expert Advice

A related problem is that of *prediction with expert advice* (for a more in-depth treatment, see [11]). In its classical formulation, we have a set of n "experts" that provide binary predictions on each trial, given by the examples $\boldsymbol{x}_t \in \{-1,1\}^n$, and again we consider both binary predictions and labels $y_t \in \{-1,1\}$. Unlike linear classification, where the performance of the algorithm is compared against the optimal linear combination of the experts' predictions, in this setting the performance is compared against that of the best expert. One of the first algorithms to tackle this problem is the halving algorithm [17, 18], which assumes realizability, i.e., that there exists an expert in the set that predicts consistently with the labels. The algorithm predicts according to the majority of the experts. When a mistake is made, more than half of the experts can be discarded, allowing it to have a mistake bound that scales logarithmically with the number of experts. The weighted majority algorithm [19] circumvents the need for the realizability assumption by maintaining weights on the different experts. Instead of discarding incorrect experts completely, it shrinks the weights of those experts by multiplying it by a constant $\beta \in [0,1)$. In the case where $\beta = 0$, we retrieve the halving algorithm. However, when $\beta \neq 0$, the weights are reduced with an exponential dependence on the number of mistakes. Further de-

velopments allowed for continuous predictions [19, 20] and continuous labels [21]. The allocation setting is a related setting, where we assign a probability vector over the experts on each trial, so that $\boldsymbol{w}_t \in \Delta_n$, where Δ_n is the n-dimensional simplex. We then receive a vector of the losses $\boldsymbol{l}_t \in [0,1]^n$, before suffering the loss $\langle \boldsymbol{w}_t, \boldsymbol{l}_t \rangle$. For this setting, a multiplicative weight update gave rise to the Hedge algorithm [22].

Linear Regression

Although linear classification is an important problem, there are situations where the labels and predictions may need to be continuous, giving online linear regression [23]. For this problem, [23] gives two algorithms. The first has a multiplicative weight update, where the gradient of the loss is exponentiated, similar to the algorithms for prediction with expert advice and the Winnow algorithm. The second has an additive weight update, with a straightforward gradient descent, similar to the perceptron update. The authors of [23] also give a motivation for these updates, where it is shown that the exponentiated gradient update can be rewritten as an optimization which minimizes the loss with a relative entropy regularizer. This is contrasted with the gradient descent update as in the perceptron algorithm, which has an ℓ_2-norm regularization term instead.

From a more modern perspective, many online learning algorithms are often viewed through the unified lens of Online Convex Optimization. In this framework, the learner predicts a vector $\boldsymbol{w}_t$ from a convex set $\mathcal{S}$ on each trial, receives a convex loss function $f_t : \mathcal{S} \to \mathfrak{R}$ and then suffers the loss $f_t(\boldsymbol{w}_t)$. A common algorithm to tackle this problem is Follow the Regularized Leader (FTRL) (see [12] for an overview), which outputs a vector that minimizes the cumulative loss and a regularization term characterized by the function $R : \mathcal{S} \to \mathfrak{R}$, i.e.

$$\boldsymbol{w}_{t+1} = \operatorname*{argmin}_{\boldsymbol{w} \in \mathcal{S}} \sum_{i \in [t]} f_i(\boldsymbol{w}) + R(\boldsymbol{w}).$$

From this, the Online Mirror Descent framework can be derived by assuming linear loss functions and simplifying to remove the explicit optimization. It requires a linking function $g : \mathfrak{R}^n \to \mathcal{S}$ and initializes $\boldsymbol{\theta}_1 = \mathbf{0}$. On each trial, it predicts

according to $\boldsymbol{w}_t = g(\boldsymbol{\theta}_t)$. After that, it updates according to $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \boldsymbol{z}_t$ where $\boldsymbol{z}_t \in \nabla f_t(\boldsymbol{w}_t)$. The general framework encapsulates the gradient descent and exponentiated gradient algorithms, for which the linking functions typically have a linear and exponential dependence on $\boldsymbol{\theta}$ respectively.

In this thesis, we will introduce algorithms that are instances of the matrix gradient descent (MGD) and matrix exponentiated gradient (MEG) [4, 24, 25, 26] algorithms. Being matrix generalizations of the vector algorithms, weight and instance matrices are considered. We adapt this algorithm to our use cases through an appropriate choice of comparator matrix, matrix embeddings and threshold value. Similar to the vector equivalents, the matrix algorithms can be motivated by an optimization problem, where the relative entropy regularizer is replaced by the quantum relative entropy for MEG (see [4, Section 3.1]), and the Frobenius norm instead of the ℓ_2 norm is used for MGD.

2.2 Matrix Completion

Although collaborative filtering popularized matrix completion, the problem was studied historically due to its other applications. In the online learning setting, early applications included learning binary relations over two sets [27], where for instance a clinician may want to predict whether a patient is allergic to a certain allergen. In the batch setting, matrix completion was applied in domains such as global positioning from limited distance information [28], remote sensing [29] where one attempts to infer a full covariance matrix from incomplete observations of correlations, system design in control [30] and the structure-from-motion problem [31]. From heuristics to more well-motivated algorithms, there are now many works that propose to solve different variants of the problem. In the following, we will elaborate on some of the representative works in this field.

2.2.1 Online Setting

Early work [27] was targeted at the problem of learning a binary relation between a set $\mathcal{I}$ of cardinality m and another set $\mathcal{J}$ of cardinality n. Representing the first set $\mathcal{I}$ as the rows and the other set $\mathcal{J}$ as the columns, this could be framed as a $m \times n$

binary matrix completion problem where the entry is 1 if the relation holds true. This work considers rows that are of k types, where rows of the same type have the same values for all entries. [32] also considers the noisy setting where rows of the same row type may not be fully identical to each other and solves it via a variant of the weighted majority algorithm. It maintains weights for each of the $\binom{m}{2}$ row pairs. When predicting for a given entry $(i, j) \in [m] \times [n]$, each of the other rows in $[m] \setminus \{i\}$ is taken as an expert whose prediction corresponds to their value of the j^{th} column entry. If the value is not known, it predicts $1/2$. The experts are combined using the weights for the row pairs of row i and the expert rows.

Regret bounds are subsequently given in [33] for the case of bounded trace norm matrices. The algorithm is based on the batch method of empirical risk minimization and the regret is given in terms of the Rademacher complexity. Although efficient, it is not a fully practical algorithm with a higher complexity than FTRL algorithms. The authors of [25] provide tight upper and lower bounds in terms of (β, τ) matrices, where β and τ are related to the max-norm and trace-norm respectively. Mistake bounds for binary matrix completion with a matrix exponentiated gradient algorithm are given in [34], which present a mistake bound in terms of the margin complexity of the matrix. An FTRL algorithm with the log-determinant regularizer instead of the quantum relative entropy is considered in [35], and provides a logarithmic improvement to the bound in [34], at the expense of an apparently higher computational complexity. Also related are the works on online PCA, which aim to recover a matrix of a restricted matrix complexity class [36, 37]. None of the above references consider the problem of side information.

Instances of matrix completion with side information on specific matrix complexity classes include the results in [38, 39]. The authors of [38] give an algorithm that predicts if vertices in a graph are "similar" and [39] addresses the problem of a switching graph labeling. In both papers, the side information is provided in the form of graph Laplacians. In this thesis, we provide generalizations upon these works in the following capacities. First, the side information considered in this thesis can be given as any pair of positive definite matrices in the transductive model

including (augmented) graph Laplacians and outputs of other kernel functions. In the inductive model, this is further generalized as the side information is given by any pair of kernels. Furthermore, we give regret bounds which are also applicable in the non-realizable case. Finally, the comparator matrix can be an arbitrary matrix.

A noteworthy direction of research which has marginal relevance to our work builds batch algorithms with incremental updates, see for instance [40, 41, 42, 43, 44, 45, 46, 47, 48]. In these works, a batch algorithm is typically used on an existing set of instances to recover a matrix estimate, which is then incrementally updated in an online fashion. These papers do not provide regret or mistake bounds; instead they provide distribution-dependent guarantees. To achieve this, some papers [42] present methods to perform incremental SVD updates, such that given a matrix and the SVD of that matrix, the SVD of the perturbed matrix can be performed efficiently. Not all of these papers consider the setting where the matrix entry is sampled uniformly at random. Some such as [43, 46] sample random subsets of a column at each trial. [49] considers a bandit-like setting where the environment presents only the row, giving the learner the choice of the column to predict.

2.2.2 Batch Setting

The number of studies on the matrix completion problem in the batch setting greatly outweigh those in the online setting. The classical approach frames this as an optimization problem, where the aim is to find a target matrix that is consistent with the matrix entries revealed thus far, under the constraint that it is of low rank. This problem is non-convex, and so a dominant direction of research resorts to convex relaxation by replacing the rank with a convex surrogate, most commonly the trace-norm. This is justified by the fact that the trace norm of a matrix with unit spectral norm is the convex envelope of its rank [50]. This method has shown good practical performance (see e.g. [51, 52]) and some early works that provide non-trivial guarantees for matrices with a low trace norm include [51, 53, 54, 55], in which both approximate and exact recovery of the matrices are considered. These all require the stringent assumption that the observed subset is sampled uniformly at random and in fact, in [53], this requirement is shown to be a necessary condition

to prove generalization error bounds based on the trace-norm for arbitrary matrices. This has spurred research directed towards replacing the trace norm constraint by a weighted variant [56, 57], or yet with the max-norm [53, 3]. A different direction was taken by [58], which considers bounded matrices in a transductive model where the entries are assumed to be sampled without replacement to give distribution-free performance guarantees with the trace-norm. Non-convex methods have also been considered for matrix completion, see e.g. [59, 60, 61].

In the following, we review some works that are concerned with the batch matrix completion problem with side information. Works on inductive matrix completion with side information include [62, 63, 64, 65], which allow for predictions to be made for new rows and columns. In the setting of [62, 63, 64], feature vectors for the rows and the columns are available. For an $m \times n$ observed matrix $\boldsymbol{R}$, we then have the feature matrices $\boldsymbol{M} \in \Re^{m \times p}$ and $\boldsymbol{N} \in \Re^{n \times q}$, where $p < n$ and $q < m$ and the rows of the feature matrices are the feature vectors. The aim is to recover a low-rank matrix $\boldsymbol{U} \in \Re^{p \times q}$ which is related to the observed matrix through rank-1 measurements of the form $\boldsymbol{R} = \boldsymbol{M}\boldsymbol{U}\boldsymbol{N}^\top$. The works [62, 64] utilize non-convex optimization methods, whereas [63] uses the trace-norm of $\boldsymbol{U}$ as a regularization term. [65] minimizes over functions in the RKHS space defined by the tensor product kernel of the kernels over the row and column space. This is similar to our approach with the MGD algorithm in Section 3.2.2. Some examples in the transductive setting include [66, 67]. These papers use graph Laplacians to model the side information. To achieve this, two graph Laplacians are used to define regularization functionals for both the rows and the columns so that rows (columns) with similar side information tend to have the same values. In particular, [67] resembles our approach by applying the Laplacian functionals to the underlying row and column factors directly. An alternate approach is taken in [66], where the regularization is instead applied to the row space (column space) of the target matrix.

Chapter 3

General Algorithms and Bounds

In this chapter, we give general online algorithms for binary matrix completion with side information and prove mistake and regret bounds. The mistake bounds we prove are of the form $\tilde{O}\left(\frac{\mathcal{D}}{\gamma^2}\right)$ and $O\left(\frac{\mathcal{D}^2}{\gamma^2}\right)$. The term $\frac{1}{\gamma^2}$ is analogous to the usual margin term in SVM (perceptron) bounds. More specifically, if we assume that there is some factorization of the underlying $m \times n$ matrix into $\boldsymbol{P}\boldsymbol{Q}^\top$, where the rows of $\boldsymbol{P}$ are interpreted as "classifiers" in $\Re^d$ and the rows of $\boldsymbol{Q}$ as "instances" in $\Re^d$, then γ is the maximum (normalized) margin over all factorizations $\boldsymbol{P}\boldsymbol{Q}^\top$ consistent with the observed matrix. The quasi-dimension term $\mathcal{D}$ measures the quality of side information. In the presence of vacuous side information, $\mathcal{D} = m+n$. We additionally provide a generalization of our algorithm to the inductive setting, where the number of rows and columns are not specified in advance.

3.1 Introduction

In this chapter, we present our general algorithms for online matrix completion with side information. We consider both transductive and inductive settings. In the former setting, the side information is completely specified in advance, whereas in the latter, the side information is provided in an online fashion. For both settings, we provide two algorithms. The first is an adapted Matrix Exponentiated Gradient (MEG) algorithm, where we prove mistake bounds of the form $\tilde{O}(\mathcal{D}/\gamma^2)$ in terms of tunable hyperparameters $\mathcal{D}$ and γ. The other algorithm is an adapted instance

of Matrix Gradient Descent (MGD), which suffers from a larger mistake bound scaling of $O(\mathcal{D}^2/\gamma^2)$, albeit offering a superior time complexity.

The term $1/\gamma^2$ is a parameter of our algorithm which serves as an upper estimate of the squared margin complexity $\mathrm{mc}(\boldsymbol{U})^2$ of the comparator matrix $\boldsymbol{U}$. The notion of margin complexity in machine learning was introduced in [68], where it was used to study the learnability of concept classes via linear embeddings. It was further studied in [9], and in [53] a detailed study of margin complexity, trace complexity and rank in the context of statistical bounds for matrix completion was given. The squared margin complexity is upper bounded by rank. Furthermore, in Chapter 4 it is explained that if our $m \times n$ matrix has a *latent block structure* with $k \times \ell$ homogeneous blocks, then $\mathrm{mc}(\boldsymbol{U})^2 \leq \min(k, \ell)$.

The second term in our bound is the quasi-dimension $\mathcal{D}$ which, to the best of our knowledge, is novel to this work. The quasi-dimension measures the extent to which the side information is "predictive" of the comparator matrix. In Theorem 57, Chapter 4, we provide an upper bound on the quasi-dimension, which measures the predictiveness of the side information when the comparator matrix has a latent block structure. If there is only vacuous side information, then $\mathcal{D} = m + n$. However, if there is a $k \times \ell$ latent block structure and the side information is predictive, then $\mathcal{D} \in O(k + \ell)$; hence our nomenclature "quasi-dimension."

The chapter is organized as follows. In Section 3.2, we present our matrix completion algorithms for the transductive setting as well as Theorems 1 and 4 which characterize their performance. In Section 3.3, we present algorithms for the inductive setting, and Proposition 6, which gives the equivalence of the transductive and inductive MEG algorithms. We provide a motivation for the update of Algorithm 1 in Section 3.4. Proofs are contained in Section 3.6. In Chapters 4 and 5, we will apply these bounds to matrices with a latent block structure and to the problem of online multitask learning with long-term memory respectively.

3.2 Transductive Matrix Completion

In this section, we discuss Algorithms 1 and 2, which perform transductive matrix completion with side information. Algorithm 1 corresponds to an adapted Matrix Exponentiated Gradient (MEG) algorithm [4]. Although the algorithm is a special case of MEG, the bounds that we provide do not follow as a special case of the analysis in [4]. Algorithm 2 corresponds to an adapted Matrix Gradient Descent (MGD) algorithm. Our mistake and regret bounds will be in terms of a real-valued comparator matrix $\boldsymbol{U}$. In all bounds, when exactly tuned, we have a quasi-dimension term and a margin complexity or max-norm term. For convenience, we shall recall the definitions of these quantities below.

The quasi-dimension of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$ with respect to $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ at margin γ is

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma \boldsymbol{U}} \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M} \hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N} \hat{\boldsymbol{Q}}\right),$$

where the infimum is over all row-normalized matrices $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ and every integer d. If the infimum does not exist then $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := +\infty$. Unless otherwise indicated, we assume that $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}})$ is the "optimal" factorization which minimizes the optimization problem in $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$. Recall that this factorization exists iff $\|\boldsymbol{U}\|_{\max} \leq \frac{1}{\gamma}$. The max-norm (or γ_2 norm [9]) of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$ is defined by

$$\|\boldsymbol{U}\|_{\max} := \min_{\boldsymbol{P}\boldsymbol{Q}^\top = \boldsymbol{U}} \left\{ \max_{1 \leq i \leq m} \|\boldsymbol{P}_i\| \times \max_{1 \leq j \leq n} \left\|\boldsymbol{Q}_j\right\| \right\},$$

where the minimum is over all matrices $\boldsymbol{P} \in \mathfrak{R}^{m \times d}$, $\boldsymbol{Q} \in \mathfrak{R}^{n \times d}$ and every integer d. The margin complexity of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$ is

$$\operatorname{mc}(\boldsymbol{U}) := \min_{\boldsymbol{V} \in \mathrm{SP}^1(\boldsymbol{U})} \|\boldsymbol{V}\|_{\max} = \min_{\boldsymbol{P}\boldsymbol{Q}^\top \in \mathrm{SP}(\boldsymbol{U})} \max_{ij} \frac{\|\boldsymbol{P}_i\| \left\|\boldsymbol{Q}_j\right\|}{|\langle \boldsymbol{P}_i, \boldsymbol{Q}_j \rangle|}$$

where the minimum is over all matrices $\boldsymbol{P} \in \mathfrak{R}^{m \times d}$, $\boldsymbol{Q} \in \mathfrak{R}^{n \times d}$ and every integer d.

3.2.1 MEG Updates

Algorithm 1 is our general MEG algorithm, which can be run using either con-

Algorithm 1 MEG algorithm for predicting a binary matrix with side information in the transductive setting.

Parameters: Learning rate: $0 < \eta$, quasi-dimension estimate: $1 \leq \widehat{\mathcal{D}}$, margin estimate: $0 < \gamma \leq 1$, non-conservative flag [NON-CONSERVATIVE] $\in \{0, 1\}$ and side information matrices $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ with $m + n \geq 3$

Initialization: $\mathbb{M} \leftarrow \emptyset$; $\tilde{\boldsymbol{W}}^1 \leftarrow \frac{\widehat{\mathcal{D}}}{(m+n)} \boldsymbol{I}^{m+n}$.

For $t = 1, \ldots, T$

- Receive pair $(i_t, j_t) \in [m] \times [n]$.
- Define

$$\tilde{\boldsymbol{X}}^t := \boldsymbol{x}^t (\boldsymbol{x}^t)^\top := \left[\frac{\sqrt{\boldsymbol{M}^+} \boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_M}}; \frac{\sqrt{\boldsymbol{N}^+} \boldsymbol{e}_n^{j_t}}{\sqrt{2\mathcal{R}_N}} \right] \left[\frac{\sqrt{\boldsymbol{M}^+} \boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_M}}; \frac{\sqrt{\boldsymbol{N}^+} \boldsymbol{e}_n^{j_t}}{\sqrt{2\mathcal{R}_N}} \right]^\top .$$

- Predict

$$Y_t \sim \text{UNIFORM}(-\gamma, \gamma) \times [\text{NON-CONSERVATIVE}] \; ; \; \bar{y}_t \leftarrow \text{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t \right) - 1 \; ; \; \hat{y}_t \leftarrow \text{sign}(\bar{y}_t - Y_t) \, .$$

- Receive label $y_t \in \{-1, 1\}$.
- If $y_t \neq \hat{y}_t$ then $\mathbb{M} \leftarrow \mathbb{M} \cup \{t\}$.
- If $y_t \bar{y}_t < \gamma \times [\text{NON-CONSERVATIVE}]$ then

$$\tilde{\boldsymbol{W}}^{t+1} \leftarrow \exp\left(\log(\tilde{\boldsymbol{W}}^t) + \eta y_t \tilde{\boldsymbol{X}}^t \right) .$$

- Else $\tilde{\boldsymbol{W}}^{t+1} \leftarrow \tilde{\boldsymbol{W}}^t$.

servative or non-conservative updates, as distinguished by the [NON-CONSERVATIVE] flag. In the case of conservative updates, the algorithm does not require the margin estimate as an input. In the following theorem, we give mistake and expected c-regret bounds for Algorithm 1, where $c(\boldsymbol{U}) = \frac{1+\max_{i,j} |\boldsymbol{U}_{ij}|}{2}$.

Theorem 1. *The expected mistakes of Algorithm 1 with* **non-conservative** *updates (*[NON-CONSERVATIVE] $= 1$*) and parameters* $\widehat{\mathcal{D}} \geq \mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$, $\eta = \sqrt{\frac{\widehat{\mathcal{D}} \log(m+n)}{2T}}$, *are bounded by*

$$\mathbb{E}[|\mathbb{M}|] \leq \frac{\left(1 + \max_{i,j} |U_{ij}|\right)}{2} \sum_{t \in [T]} [y_t \neq \text{sign}(U_{i_t j_t})] + \frac{3.5}{\gamma} \sqrt{\widehat{\mathcal{D}} \log(m+n) T} \qquad (3.1)$$

for all $\boldsymbol{U} \in ((-\infty, -1] \cup [1, \infty))^{m \times n}$ *with* $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$.

The mistakes in the **realizable** *case with* **conservative** *updates (*[NON-CONSERVATIVE] $=$

0) and parameters $\eta = \gamma$, $\widehat{\mathcal{D}} \geq \min_{\boldsymbol{V} \in \mathrm{SP}^1(\boldsymbol{U})} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V})$ *are bounded by,*

$$|\mathbb{M}| \leq 3.6(\widehat{\mathcal{D}}/\gamma^2)\log(m+n)\,, \tag{3.2}$$

for all $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$ *with* $\mathrm{mc}(\boldsymbol{U}) \leq 1/\gamma$ *and* $y_t = \mathrm{sign}(U_{i_t j_t})$ *for all* $t \in \mathbb{M}$.

The regret statement of the theorem requires that $\min_{i,j} |U_{ij}| \geq 1$. We note that this is not a restrictive requirement, since we can scale $\boldsymbol{U}$ and γ to obtain bounds for all $\boldsymbol{U} \in ((-\infty, -a] \cup [a, \infty))^{m\times n}$ where $a > 0$. In addition, for both statements, the requirement on $\frac{1}{\gamma}$ in terms of $\boldsymbol{U}$ is superfluous due to the presence of the quasi-dimension term; if the requirement is not fulfilled, the quasi-dimension will be infinite as per its definition. We have nevertheless decided to include it for clarity.

In the special case where $\boldsymbol{U} \in \{-1, 1\}^{m\times n}$, we obtain the following "true" regret bound from the c-regret bound in Theorem 1:

$$\mathbb{E}[|\mathbb{M}|] \leq \sum_{t\in[T]} [y_t \neq U_{i_t j_t}] + \frac{3.5}{\gamma}\sqrt{\widehat{\mathcal{D}}\log(m+n)T}. \tag{3.3}$$

It may seem confusing at first that we have derived a regret bound for the 0-1 loss. However, note that this result is contingent on a strict assumption; the comparator matrices $\boldsymbol{U}$ must be binary. In fact, our regret bounds follow naturally from hinge loss regret bounds, which also appear in our analysis. Although a hinge loss regret bound may appear more familiar to the reader, we have chosen to present our main results in terms of the 0-1 loss, with the view that it is a more natural choice for binary comparator matrices. In particular, although Equation (3.1) in Theorem 1 is valid for any real-valued comparator matrix, all its applications in this thesis will instead derive bounds with binary comparator matrices, where the real-valued matrix is merely used as a more flexible embedding for the binary matrix. Apart from allowing for this flexibility, (3.1) can also give tighter bounds as it is possible for a matrix $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$ with $\max_{i,j} |U_{ij}| > 1$ to have a smaller max-norm than its sign matrix $\boldsymbol{U}' \in \{\mathrm{SP}(\boldsymbol{U}) \cap \{-1, 1\}^{m\times n}\}$, see e.g. [9].

If the side information is vacuous, that is $\boldsymbol{M} = \boldsymbol{I}^m$ and $\boldsymbol{N} = \boldsymbol{I}^n$, then $\mathcal{D} = m+n$.

In this scenario, Equation (3.3) recovers a special case of the analysis of [25] up to constant factors. In [25], a regret bound for general loss functions for matrix completion without side information is given for (β, τ)-decomposable matrices. When β is at its minimum over all possible decompositions, we recover the bound up to constant factors with respect to the zero-one loss. On the algorithmic level, our works are similar except that the algorithm of [25] contains an additional projection step that dominates the computation time of the update. With the additional assumption of realizability, we recover [34, Theorem 3.1]. The term $\mathcal{D}$ is difficult to directly quantify, and we will interpret it further in Chapters 4 and 5 for specific matrix complexity classes and side information. In Chapter 4, we also provide a lower bound and show that our bound is tight up to logarithmic factors for matrices with a latent block structure.

The general form of our regret bound of the MEG-based algorithm comes from a matricization of the regret bound proven for a Winnow-inspired algorithm [16] for linear classification in the vector case given in [69]. Regret bounds for the MEG algorithm were originally proven in [4]. However, that analysis leads to a $\widehat{\mathcal{D}}^2$ dependence in the mistake bound, whereas we derive a $\widehat{\mathcal{D}}$ scaling, for our more restrictive setting. Regret bounds with such scaling for linear classification in the vector case have been previously given in [69] (which themselves are generalisations of the bounds from Littlestone [16] for learning k-literal disjunctions with $O(k \log n)$ mistakes). However, to our knowledge, no such regret bounds for MEG are present in the literature for the matrix case. Our proof uses an amortized analysis of the quantum relative entropy, followed by an extension of the results in [69] to the matrix case. We also note that our bound is reminiscent of Novikoff's perceptron bound when we consider a factorization of $\boldsymbol{U}$ into $\boldsymbol{P}\boldsymbol{Q}^\intercal$ and interpret $\boldsymbol{P}$ as classifiers and $\boldsymbol{Q}$ as instances, as mentioned at the start of the chapter. In that case, the margin terms are analogous, and the quasi-dimension term bears a resemblance to the squared radius term in their definitions.

For more intuition, we give more details on the embeddings that we use in the

analysis for the algorithm. For $\boldsymbol{U} \in ((-\infty, -1] \cup [1, \infty))^{m \times n}$, we define

$$\bar{\boldsymbol{U}} := \gamma \boldsymbol{U}, \tag{3.4}$$

and observe that it has entries at least as large as the margin γ. We cannot apply $\bar{\boldsymbol{U}}$ directly in the analysis of the MEG algorithm, which requires positive semi-definite matrices. In what follows, we define $\tilde{\boldsymbol{U}} \in \boldsymbol{S}_+^{m+n}$, a positive semi-definite embedding for $\bar{\boldsymbol{U}}$ used in the analysis of the algorithm. Its relationship with $\bar{\boldsymbol{U}}$ is shown in Lemma 3. Thus we can understand that if $\tilde{\boldsymbol{W}}^t = \tilde{\boldsymbol{U}}$, the prediction of the algorithm on trial t is equal to the $(i_t, j_t)^{th}$ entry of $\bar{\boldsymbol{U}}$.

Definition 2. *Define the* $(m + n) \times d$ *matrix* $\boldsymbol{Z}$ *as*

$$\boldsymbol{Z} := \begin{pmatrix} \sqrt{\mathcal{R}_M}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}} \\ \sqrt{\mathcal{R}_N}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}} \end{pmatrix}.$$

and construct $\tilde{\boldsymbol{U}}$ *as,*

$$\tilde{\boldsymbol{U}} := \boldsymbol{Z}\boldsymbol{Z}^\top = \begin{pmatrix} \mathcal{R}_M \sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{P}}^\top\sqrt{\boldsymbol{M}} & \sqrt{\mathcal{R}_M\mathcal{R}_N}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top\sqrt{\boldsymbol{N}} \\ \sqrt{\mathcal{R}_M\mathcal{R}_N}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}}\hat{\boldsymbol{P}}^\top\sqrt{\boldsymbol{M}} & \mathcal{R}_N\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}}\hat{\boldsymbol{Q}}^\top\sqrt{\boldsymbol{N}} \end{pmatrix}.$$

Lemma 3. *For all trials* $t \in [T]$*,*

$$\bar{\boldsymbol{U}}_{i_t j_t} = \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) - 1$$

where $\tilde{\boldsymbol{U}}$ *is as constructed from Definition 2.*

Proof. We have:

$$\begin{aligned} \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) &= (\boldsymbol{x}^t)^\top \tilde{\boldsymbol{U}} \boldsymbol{x}^t \\ &= (\boldsymbol{x}^t)^\top \boldsymbol{Z}\boldsymbol{Z}^\top \boldsymbol{x}^t \\ &= \left\|(\boldsymbol{x}^t)^\top \boldsymbol{Z}\right\|^2 . \end{aligned} \tag{3.5}$$

Recall that

$$\boldsymbol{x}^t = \left[\frac{\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_{\boldsymbol{M}}}}; \frac{\sqrt{\boldsymbol{N}^+}\boldsymbol{e}_n^{j_t}}{\sqrt{2\mathcal{R}_{\boldsymbol{N}}}} \right] \text{ and } \boldsymbol{Z} = \begin{pmatrix} \sqrt{\mathcal{R}_{\boldsymbol{M}}}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}} \\ \sqrt{\mathcal{R}_{\boldsymbol{N}}}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}} \end{pmatrix}$$

Hence,

$$\begin{aligned}(\boldsymbol{x}^t)^\top \boldsymbol{Z} &= \frac{(\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t})^\top}{\sqrt{2\mathcal{R}_{\boldsymbol{M}}}}\sqrt{\mathcal{R}_{\boldsymbol{M}}}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}} + \frac{(\sqrt{\boldsymbol{N}^+}\boldsymbol{e}_n^{j_t})^\top}{\sqrt{2\mathcal{R}_{\boldsymbol{N}}}}\sqrt{\mathcal{R}_{\boldsymbol{N}}}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}} \\ &= \frac{1}{\sqrt{2}}(\boldsymbol{e}_m^{i_t})^\top\sqrt{\boldsymbol{M}^+}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}} + \frac{1}{\sqrt{2}}(\boldsymbol{e}_n^{j_t})^\top\sqrt{\boldsymbol{N}^+}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}} \\ &= \frac{1}{\sqrt{2}}(\hat{\boldsymbol{P}}_{i_t} + \hat{\boldsymbol{Q}}_{j_t}) \end{aligned} \tag{3.6}$$

Thus substituting (3.6) into (3.5) gives,

$$\begin{aligned}\operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) &= \frac{1}{2}\left\|\hat{\boldsymbol{P}}_{i_t} + \hat{\boldsymbol{Q}}_{j_t}\right\|^2 \\ &= \frac{1}{2}\left(\left\|\hat{\boldsymbol{P}}_{i_t}\right\|^2 + 2\langle\hat{\boldsymbol{P}}_i, \hat{\boldsymbol{Q}}_{j_t}\rangle + \left\|\hat{\boldsymbol{Q}}_{j_t}\right\|^2\right) \\ &= \left(1 + \langle\hat{\boldsymbol{P}}_{i_t}, \hat{\boldsymbol{Q}}_{j_t}\rangle\right) \\ &= 1 + \bar{\boldsymbol{U}}_{i_t j_t}.\end{aligned}$$

□

3.2.2 MGD Updates

Algorithm 2 is our transductive MGD algorithm. Unlike Algorithm 1, Algorithm 2 does not require a quasi-dimension estimate $\widehat{\mathcal{D}}$. Furthermore, the margin estimate is not necessary in the case of conservative updates. The mistake and expected c-regret bounds for Algorithm 2 are presented in Theorem 4.

Theorem 4. *The expected mistakes of Algorithm 2 with* ***non-conservative*** *updates ([*NON-CONSERVATIVE*]=1), and learning rate* $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{T}}$*, where* $\widehat{\mathcal{D}}^2 \geq \mathcal{A}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$*, are bounded by*

$$\mathbb{E}[|\mathbb{M}|] \leq \frac{\left(1 + \max_{ij}|U_{ij}|\right)}{2}\sum_{t\in[T]}[y_t \neq \operatorname{sign}(U_{i_t j_t})] + \frac{\widehat{\mathcal{D}}}{\gamma}\sqrt{T}$$

Algorithm 2 MGD algorithm for predicting a binary matrix with side information in the transductive setting.

Parameters: Learning rate: $0 < \eta$, margin estimate: $0 < \gamma \leq 1$, non-conservative flag [NON-CONSERVATIVE] $\in \{0, 1\}$ and side information matrices $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ with $m + n \geq 3$

Initialization: $\mathbb{M} \leftarrow \emptyset$; $\tilde{\boldsymbol{W}}^1 \leftarrow \boldsymbol{0}$.

For $t = 1, \ldots, T$

- Receive pair $(i_t, j_t) \in [m] \times [n]$.
- Define
$$\tilde{\boldsymbol{X}}^t := \frac{1}{\sqrt{\mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}}}} \sqrt{\boldsymbol{M}^+} \boldsymbol{e}_{i_t} (\boldsymbol{e}_{j_t})^\top \sqrt{\boldsymbol{N}^+}. \tag{3.7}$$
- Predict

$Y_t \sim \text{UNIFORM}(-\gamma, \gamma) \times [\text{NON-CONSERVATIVE}]$; $\bar{y}_t \leftarrow \text{tr}\left(\tilde{\boldsymbol{W}}^t (\tilde{\boldsymbol{X}}^t)^\top\right)$; $\hat{y}_t \leftarrow \text{sign}(\bar{y}_t - Y_t)$.

- Receive label $y_t \in \{-1, 1\}$.
- If $y_t \neq \hat{y}_t$, then $\mathbb{M} \leftarrow \mathbb{M} \cup \{t\}$.
- If $y_t \bar{y}_t < \gamma \times [\text{NON-CONSERVATIVE}]$ then
$$\tilde{\boldsymbol{W}}^{t+1} \leftarrow \tilde{\boldsymbol{W}}^t + \eta y_t \tilde{\boldsymbol{X}}^t. \tag{3.8}$$
- Else $\tilde{\boldsymbol{W}}^{t+1} \leftarrow \tilde{\boldsymbol{W}}^t$.

for all $\boldsymbol{U} \in ((-\infty, -1] \cup [1, \infty))^{m \times n}$ *with* $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$.

The mistakes in the **realizable** *case with* ***conservative*** *updates ([*NON-CONSERVATIVE*]=0) and learning rate* $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{|\mathbb{M}|}}$, *where* $\widehat{\mathcal{D}}^2 \geq \min_{\boldsymbol{V} \in \text{SP}^1(\boldsymbol{U})} \mathcal{A}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V})$, *are bounded by,*

$$|\mathbb{M}| \leq \frac{\widehat{\mathcal{D}}^2}{\gamma^2},$$

for all $\boldsymbol{U} \in \Re^{m \times n}$ *with* $\text{mc}(\boldsymbol{U}) \leq 1/\gamma$ *and* $y_t = \text{sign}(U_{i_t j_t})$ *for all* $t \in \mathbb{M}$.

The bounds can be rewritten in terms of the quasi-dimension, since $\mathcal{A}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \leq \frac{(\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}))^2}{4}$ from the inequality $4ab \leq (a + b)^2$ for scalars a, b, so that $\widehat{\mathcal{D}} \geq \frac{\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})}{2}$. After this manipulation, we obtain a mistake bound in the realizable case (with exact tuning) of $O(\mathcal{D}^2 \, \text{mc}(\boldsymbol{U})^2)$. This has a quadratic dependence on $\mathcal{D}$, which is worse than the linear dependence in Theorem 1 for the MEG algorithm. It may seem prohibitive that tuning the learning rate exactly for the mistake bound requires knowing the number of mistakes beforehand. However, we can use

the doubling trick to avoid this. In the special case where $\boldsymbol{U} \in \{-1, 1\}^{m \times n}$,

$$\mathbb{E}[|\mathbb{M}|] \leq \sum_{t \in \mathbb{M}} [y_t \neq U_{i_t, j_t}] + \frac{\widehat{\mathcal{D}}}{\gamma} \sqrt{T} \tag{3.9}$$

follows directly from the theorem, which gives us an expected regret bound with $c = 1$. In the case of vacuous side information, that is $\boldsymbol{M} = \boldsymbol{I}^m$ and $\boldsymbol{N} = \boldsymbol{I}^n$, the mistake bound becomes $O(mn \operatorname{mc}(\boldsymbol{U})^2)$ which is vacuous since $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$. Thus the bound is only interesting when the side information provides a significant improvement to the quasi-dimension term.

The MGD algorithm offers the benefit that the prediction can be computed through the dual form, as shown in Proposition 5. The dual form computes updates with a per-trial time complexity of $O(\min(mn, T))$, similar to the direct implementation which has a complexity of $O(mn)$ when taking advantage of the fact that $\tilde{\boldsymbol{X}}^t$ is of rank 1. Both implementations are superior to the time complexity of $O((m+n)^3)$ for the transductive MEG algorithm.

Proposition 5. *The prediction given by Algorithm 2 on trial t can be equivalently computed through:*

$$\bar{y}_t = \frac{\eta}{\mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}}} \sum_{s \in \mathbb{U}_t} y_s M^+_{i_t, i_s} N^+_{j_t, j_s} \tag{3.10}$$

where $\mathbb{U}_t : \{s : y_s \bar{y}_s < \gamma [\textsc{non-conservative}], s < t\}$.

Proof. Recall that $\bar{y}_t = \operatorname{tr}(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t)$, where $\tilde{\boldsymbol{X}}^t = \frac{1}{\sqrt{\mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}}}} \sqrt{\boldsymbol{M}^+} \boldsymbol{e}_{i_t} (\boldsymbol{e}_{j_t})^\top \sqrt{\boldsymbol{N}^+}$. We observe that $\tilde{\boldsymbol{W}}^t$ can be written

$$\tilde{\boldsymbol{W}}^t = \frac{\eta}{\sqrt{\mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}}}} \sum_{s \in \mathbb{U}_t} \sqrt{\boldsymbol{M}^+} \boldsymbol{e}_{i_s} \boldsymbol{e}_{j_s}^\top \sqrt{\boldsymbol{N}^+}$$

Using the linear and cyclic properties of the trace

$$\begin{aligned} \operatorname{tr}(\tilde{\boldsymbol{W}}^t (\tilde{\boldsymbol{X}}^t)^\top) &= \frac{\eta}{\mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}}} \sum_{s \in \mathbb{U}_t} \operatorname{tr}(\sqrt{\boldsymbol{M}^+} \boldsymbol{e}_{i_s} \boldsymbol{e}_{j_s}^\top \boldsymbol{N}^+ \boldsymbol{e}_{j_t} \boldsymbol{e}_{i_t}^\top \sqrt{\boldsymbol{M}^+}) \\ &= \frac{\eta}{\mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}}} \sum_{s \in \mathbb{U}_t} \boldsymbol{e}_{i_t}^\top \boldsymbol{M}^+ \boldsymbol{e}_{i_s} \boldsymbol{e}_{j_s}^\top \boldsymbol{N}^+ \boldsymbol{e}_{j_t} \end{aligned} \tag{3.11}$$

Recalling that $\boldsymbol{M}$ and $\boldsymbol{N}$ are symmetric matrices, and that the inverse of a symmetic matrix is also symmetric, the proposition follows from (3.11). □

It is also possible to run the algorithm using positive semi-definite embeddings for $\tilde{\boldsymbol{W}}^t$ and $\tilde{\boldsymbol{X}}^t$, similar to those in Algorithm 1. By following a similar analysis to that for Theorem 4, we obtain similar bounds, with the only difference being that $\widehat{\mathcal{D}}^2 \geq \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})^2$. The corresponding prediction in the dual form is then given by

$$\bar{y}_t = \eta \sum_{s \in \mathbb{U}_t} y_s \left(\frac{1}{2\mathcal{R}_{\boldsymbol{M}}} M^+_{i_t i_s} + \frac{1}{2\mathcal{R}_{\boldsymbol{N}}} N^+_{i_t i_s} \right)^2 . \tag{3.12}$$

Observe that the predictions given by Equations (3.10) and (3.12) are similar to the (vector) kernel perceptron, where the kernel is respectively given by the product and sum squared of the row and column kernels.

3.3 Inductive Matrix Completion

In the previous section, the learner was assumed to have complete foreknowledge of the side information through the matrices $\boldsymbol{M}$ and $\boldsymbol{N}$. In the inductive setting, the learner has instead kernel side information functions $\mathcal{M}^+$ and $\mathcal{N}^+$. With complete foreknowledge of the rows (columns) that will be observed, one may use $\mathcal{M}^+$ ($\mathcal{N}^+$) to compute $\boldsymbol{M}$ ($\boldsymbol{N}$), which corresponds to an inverse of a submatrix of $\mathcal{M}^+$ ($\mathcal{N}^+$). In the inductive, unlike the transductive setting, we do not have this foreknowledge and thus cannot compute $\boldsymbol{M}$ ($\boldsymbol{N}$) in advance. Notice that the assumption of side information as kernel functions is not particularly limiting, as for instance the side information could be provided by vectors in $\mathfrak{R}^d$ and the kernel could be the positive definite linear kernel $\boldsymbol{K}_\epsilon(\boldsymbol{x}, \boldsymbol{x}') := \langle \boldsymbol{x}, \boldsymbol{x}' \rangle + \epsilon[\boldsymbol{x} = \boldsymbol{x}']$.

For the algorithm with MGD updates, it is trivial to generalize to the inductive setting by considering the dual form shown in (3.10), and the corresponding inductive algorithm retains the same time complexity of $O(\min(mn, T))$. Hence, we will focus our discussion on MEG updates. Algorithm 3 is prediction-equivalent to Algorithm 1 with MEG updates up to the value of $\mathcal{R}_{\boldsymbol{M}}(\mathcal{R}_{\boldsymbol{N}})$. In [70], the authors provide very general conditions for the "kernelization" of algorithms with an

Algorithm 3 Predicting a binary matrix with side information in the inductive setting.

Parameters: Learning rate: $0 < \eta$, quasi-dimension estimate: $1 \leq \widehat{\mathcal{D}}$, margin estimate: $0 < \gamma \leq 1$, non-conservative flag [NON-CONSERVATIVE] $\in \{0, 1\}$, side-information kernels $\mathcal{M}^+ : \mathcal{I} \times \mathcal{I} \to \mathfrak{R}$, $\mathcal{N}^+ : \mathcal{J} \times \mathcal{J} \to \mathfrak{R}$, with $\mathcal{R}_{\mathcal{M}} := \max_{i \in \mathcal{I}} \mathcal{M}^+(i,i)$ and $\mathcal{R}_{\mathcal{N}} := \max_{j \in \mathcal{J}} \mathcal{N}^+(j,j)$, and maximum distinct rows m and columns n, where $m + n \geq 3$.

Initialization: $\mathbb{M} \leftarrow \emptyset\,, \mathbb{U} \leftarrow \emptyset\,, \mathcal{I}^1 \leftarrow \emptyset\,, \mathcal{J}^1 \leftarrow \emptyset$.

For $t = 1, \ldots, T$

- Receive pair $(i_t, j_t) \in \mathcal{I} \times \mathcal{J}$.
- Define

$$(\boldsymbol{M}^t)^+ := (\mathcal{M}^+(i_r, i_s))_{r,s \in \mathcal{I}^t \cup \{i_t\}}\,; \quad (\boldsymbol{N}^t)^+ := (\mathcal{N}^+(j_r, j_s))_{r,s \in \mathcal{J}^t \cup \{j_t\}}\,,$$

$$\tilde{\boldsymbol{X}}^t(s) := \left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\boldsymbol{e}^{i_s}}{\sqrt{2\mathcal{R}_{\mathcal{M}}}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\boldsymbol{e}^{j_s}}{\sqrt{2\mathcal{R}_{\mathcal{N}}}}\right]\left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\boldsymbol{e}^{i_s}}{\sqrt{2\mathcal{R}_{\mathcal{M}}}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\boldsymbol{e}^{j_s}}{\sqrt{2\mathcal{R}_{\mathcal{N}}}}\right]^\top,$$

$$\log(\tilde{\boldsymbol{W}}^t) \leftarrow \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\right)\boldsymbol{I}^{|\mathcal{I}^t|+|\mathcal{J}^t|+2} + \sum_{s \in \mathbb{U}} \eta y_s \tilde{\boldsymbol{X}}^t(s)\,.$$

- Predict

$$Y_t \sim \text{UNIFORM}(-\gamma, \gamma) \times [\text{NON-CONSERVATIVE}]\,;\ \bar{y}_t \leftarrow \text{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t(t)\right) - 1\,;\ \hat{y}_t \leftarrow \text{sign}(\bar{y}_t - Y_t)\,.$$

- Receive label $y_t \in \{-1, 1\}$.
- If $y_t \neq \hat{y}_t$ then $\mathbb{M} \leftarrow \mathbb{M} \cup \{t\}$.
- If $y_t \bar{y}_t < \gamma \times [\text{NON-CONSERVATIVE}]$ then

$$\mathbb{M} \leftarrow \mathbb{M} \cup \{t\}\,,\ \ \mathcal{I}^{t+1} \leftarrow \mathcal{I}^t \cup \{i_t\},\ \text{and}\ \mathcal{J}^{t+1} \leftarrow \mathcal{J}^t \cup \{j_t\}\,.$$

- Else $\mathcal{I}^{t+1} \leftarrow \mathcal{I}^t$ and $\mathcal{J}^{t+1} \leftarrow \mathcal{J}^t$.

emphasis on "matrix" algorithms. They sketch a method to kernelize the MATRIX EXPONENTIATED GRADIENT algorithm based on the relationship between the eigensystems of the kernel matrix and the Gram matrix. We take a different, more direct approach, in which we prove its correctness via Proposition 6.

The intuition behind the algorithm is that, although we cannot efficiently embed the row and column kernel functions $\mathcal{M}^+$ and $\mathcal{N}^+$ as matrices since they are potentially infinite-dimensional, we may instead work with the embedding corresponding to the currently observed rows and columns, recompute the embedding on a per-trial basis, and then "replay" all re-embedded past examples to create the

current hypothesis matrix. The following is our proposition of equivalency, proven in Section 3.6.3.

Proposition 6. *The inductive and transductive algorithms are equivalent up to $\mathcal{R}_{\boldsymbol{M}}$ and $\mathcal{R}_{\boldsymbol{N}}$. Without loss of generality assume $\mathcal{I}^{T+1} \subseteq [m]$ and $\mathcal{J}^{T+1} \subseteq [n]$. Define $\boldsymbol{M} := ((\mathcal{M}^+(i', i''))_{i', i'' \in [m]})^+$ and $\boldsymbol{N} := ((\mathcal{N}^+(j', j''))_{j', j'' \in [n]})^+$. Assume that for the transductive algorithm, the matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ are given whereas for the inductive algorithm, only the strictly positive definite kernel functions $\mathcal{M}^+$ and $\mathcal{N}^+$ are provided. Then, if $\mathcal{R}_{\mathcal{M}} = \mathcal{R}_{\boldsymbol{M}}$ and $\mathcal{R}_{\mathcal{N}} = \mathcal{R}_{\boldsymbol{N}}$, and if the algorithms receive the same label and index sequences, then the predictions of the algorithms are the same.*

Thus, the only case when the algorithms are different is when $\mathcal{R}_{\mathcal{M}} \neq \mathcal{R}_{\boldsymbol{M}}$ or $\mathcal{R}_{\mathcal{N}} \neq \mathcal{R}_{\boldsymbol{N}}$. This is a minor inequivalency, as the only resultant difference is in the term $\mathcal{D}$. Alternatively, if one uses a normalized kernel such as the Gaussian, then $\mathcal{R}_{\mathcal{M}} = \mathcal{R}_{\boldsymbol{M}} = 1$.

For Algorithm 1 with MEG updates, we have a per trial time complexity of $O(\max(m, n)^3)$; Algorithm 3 has a per trial complexity of $O(\min(\max(m, n)^4, T^3))$. In the dominant step on every trial (with an update) of the transductive algorithm, there is an SVD of a $(m + n) \times (m + n)$ matrix; thus, the algorithm requires $O(\max(m, n)^3)$ time. We split the analysis of the inductive algorithm into two cases. In the case that $\max(m, n) \ll T$, the complexity on every trial is dominated by the sum of up to mn matrices of size up to $(m+n) \times (m+n)$ (i.e., in the regret setting we can collapse terms from multiple observations of the same matrix entry) and thus has a per-trial complexity of $O(\max(m, n)^4)$. In the other case, on trial t, we need $O(t^3)$ time since we need to compute the eigendecomposition of three $O(t) \times O(t)$ matrices as well as sum $O(t) \times O(t)$ matrices up to t times. Putting together we have a time complexity of $O(\min(\max(m, n)^4, T^3))$ per trial.

3.4 Motivation of Update for Algorithm 1

As alluded to in Section 2.2.1, the update rules of online algorithms are often motivated by the fact that they are the result of an implicit optimization. This typically aims to find a weight matrix at each time step t that trades off between minimiz-

ing the loss at that time step and a regularization term which ensures that the new weight matrix does not stray too far from the weight matrix at the previous time step $t-1$. For the MEG algorithm, the "distance" between the two weight matrices is measured by the quantum relative entropy (see e.g. [4]). Analogously, many batch matrix completion algorithms also aim to solve an optimization problem (for examples, see Section 2.2.2). Similar to the online counterpart, a typical formulation trades off between minimizing the cumulative loss and a regularization term that is related to a complexity measure of the weight matrix, such as the trace norm. In this section, we motivate the update rule in Algorithm 1 by giving an analogous batch optimization problem.

Let us recall that the prediction of Algorithm 1 before randomization is given by

$$\bar{y}_t = \mathrm{tr}(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t) - 1,$$

where

$$\tilde{\boldsymbol{W}}^t := \begin{cases} \exp\left(\log(\tilde{\boldsymbol{W}}^{t-1}) + \eta y_t \tilde{\boldsymbol{X}}^t\right) & \text{if } y_t \bar{y}_t < \gamma [\textsc{non-conservative}] \\ \tilde{\boldsymbol{W}}^{t-1} & \text{otherwise,} \end{cases} \tag{3.13}$$

$$\tilde{\boldsymbol{X}}^t = \frac{1}{2} \boldsymbol{D}^{-1} \left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right] \left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]^\top \boldsymbol{D}^{-1}, \tag{3.14}$$

$\tilde{\boldsymbol{W}}^1 = \frac{\widehat{\mathcal{D}}}{m+n} \boldsymbol{I}^{m+n}$, $\boldsymbol{D} := \mathrm{diag}\left(\sqrt{\boldsymbol{M}}\sqrt{\mathcal{R}_M}, \sqrt{\boldsymbol{N}}\sqrt{\mathcal{R}_N}\right)$, with row and column side information matrices $\boldsymbol{M} \in \boldsymbol{S}_+^m$, and $\boldsymbol{N} \in \boldsymbol{S}_+^n$. For simplicity, we consider non-conservative updates for the remainder of the section. We can prove similar results for conservative updates by only considering trials with mistakes.

We furthermore define

$$\boldsymbol{X}^t := \frac{1}{2} \left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right] \left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]^\top,$$

the modified hinge loss with respect to matrix $\boldsymbol{W} \in \Re^{(m+n)\times(m+n)}$

$$f_\gamma^t(\boldsymbol{W}) := \frac{1}{\gamma} [\gamma - y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1)]_+,$$

and the quantum relative entropy for positive semi-definite matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ as $\Delta(\boldsymbol{A}, \boldsymbol{B}) := \text{tr}(\boldsymbol{A} \log \boldsymbol{A} - \boldsymbol{A} \log \boldsymbol{B} + \boldsymbol{B} - \boldsymbol{A})$.

In the following, we present the analogous batch optimization problem in (3.15). The minimum of this problem, $\boldsymbol{W}_1^{*,t+1}$, will be shown to be the same as the minimum of the optimization in (3.16) through Proposition 7. It is then straightforward to turn (3.16) into an "online approximate optimization", given by (3.17). As shown in Proposition 8, the minimum $\boldsymbol{W}^{*,t}$ of this optimization problem relates to the prediction $\bar{y}_t$ in Algorithm 1 through the equation $\bar{y}_t = \text{tr}(\boldsymbol{W}^{*,t}\boldsymbol{X}^t) - 1$. The optimization problems are as follows:

1.

$$\boldsymbol{W}_1^{*,t+1} := \underset{\boldsymbol{W}:\boldsymbol{W}=\boldsymbol{Z}\boldsymbol{Z}^\top, \boldsymbol{Z}=[\boldsymbol{P};\boldsymbol{Q}]}{\text{argmin}} \Delta(\mathcal{R}_{\boldsymbol{M}}\boldsymbol{P}^\top\boldsymbol{M}\boldsymbol{P} + \mathcal{R}_{\boldsymbol{N}}\boldsymbol{Q}^\top\boldsymbol{N}\boldsymbol{Q}, \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}^k) + \eta\gamma \sum_{s\in[t]} f_\gamma^s(\boldsymbol{W}) - \frac{\widehat{\mathcal{D}}k}{m+n}, \quad (3.15)$$

where the optimization is over all $\boldsymbol{P} \in \mathfrak{R}^{n\times k}$, $\boldsymbol{Q} \in \mathfrak{R}^{m\times k}$ and k.

2.

$$\boldsymbol{W}_2^{*,t+1} := \underset{\boldsymbol{W}:\boldsymbol{W}\in\boldsymbol{S}_+^{m+n}}{\text{argmin}} \Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \boldsymbol{D}\boldsymbol{W}^{*,1}\boldsymbol{D}) + \eta\gamma \sum_{s\in[t]} f_\gamma^s(\boldsymbol{W}), \quad (3.16)$$

where $\boldsymbol{W}^{*,1} = \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{D}^{-2}$.

3.

$$\boldsymbol{W}^{*,t+1} = \underset{\boldsymbol{W}:\boldsymbol{W}\in\boldsymbol{S}_+^{m+n}}{\text{argmin}} \Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}) + \eta\gamma\, f_\gamma^t(\boldsymbol{W}). \quad (3.17)$$

Proposition 7. *For all t*

$$\boldsymbol{W}_1^{*,t} = \boldsymbol{W}_2^{*,t}$$

where $\boldsymbol{W}_1^{*,t}$ is defined as in (3.15) *and $\boldsymbol{W}_2^{*,t}$ is defined as in* (3.16)*.*

Proposition 8. *For all $t \in [T]$, the predictions $\bar{y}_t$ in Algorithm 1 can be written as*

$$\bar{y}_t = \text{tr}(\boldsymbol{W}^{*,t}\boldsymbol{X}^t) - 1,$$

where $\boldsymbol{W}^{*,t}$ *is as defined in* (3.17) *and we recall that*

$$\boldsymbol{X}^t := \frac{1}{2}\left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]\left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]^\top.$$

We now aim to further our understanding of optimization problem 1 in (3.15). We suggest in the following proposition that the term $-\frac{\widehat{\mathcal{D}}k}{m+n}$ comes from the fact that the optimal decomposition into $\boldsymbol{P}$ and $\boldsymbol{Q}$ is not unique.

Proposition 9. *Let* $\boldsymbol{P}' := (\boldsymbol{P}, \boldsymbol{0}) \in \mathfrak{R}^{m\times k'}$ *and* $\boldsymbol{Q}' := (\boldsymbol{Q}, \boldsymbol{0}) \in \mathfrak{R}^{n\times k'}$*, where* $k' = k + c$ *for some arbitrary* $c > 0$*. Defining* $\boldsymbol{B} := \mathcal{R}_M \boldsymbol{P}^\top \boldsymbol{M}\boldsymbol{P} + \mathcal{R}_N \boldsymbol{Q}^\top \boldsymbol{N}\boldsymbol{Q}$*,* $\boldsymbol{B}' = \mathcal{R}_M (\boldsymbol{P}')^\top \boldsymbol{M}\boldsymbol{P}' + \mathcal{R}_N (\boldsymbol{Q}')^\top \boldsymbol{N}\boldsymbol{Q}'$ *and* $a := \frac{\widehat{\mathcal{D}}}{m+n}$*, we then have*

$$\Delta(\boldsymbol{B}, a\boldsymbol{I}^k) - ak = \Delta(\boldsymbol{B}', a\boldsymbol{I}^{k'}) - ak'$$

Proof. We have

$$\begin{aligned}\Delta(\boldsymbol{B}, a\boldsymbol{I}^k) &= \operatorname{tr}(\boldsymbol{B}\log(\boldsymbol{B})) - \operatorname{tr}\left(\boldsymbol{B}\log\left(a\boldsymbol{I}^k\right)\right) + \operatorname{tr}(a\boldsymbol{I}^k) - \operatorname{tr}(\boldsymbol{B}) \\ &= \operatorname{tr}(\boldsymbol{B}\log(\boldsymbol{B})) - \log(a)\operatorname{tr}(\boldsymbol{B}) - \operatorname{tr}(\boldsymbol{B}) + ak\end{aligned}$$

Similarly, we have

$$\Delta(\boldsymbol{B}', a\boldsymbol{I}^{k'}) = \operatorname{tr}(\boldsymbol{B}'\log(\boldsymbol{B}')) - \log(a)\operatorname{tr}(\boldsymbol{B}') - \operatorname{tr}(\boldsymbol{B}') + ak'.$$

Observing that $\boldsymbol{B}' = \begin{bmatrix}\boldsymbol{B} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}$, we clearly have $\operatorname{tr}(\boldsymbol{B}') = \operatorname{tr}(\boldsymbol{B})$. To evaluate $\operatorname{tr}(\boldsymbol{B}'\log\boldsymbol{B}')$, we observe that for $\boldsymbol{B}$ with eigendecomposition $\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^\top$, $\boldsymbol{B}'$ has the eigendecomposition $\begin{bmatrix}\boldsymbol{V} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{A}\end{bmatrix}\begin{bmatrix}\boldsymbol{\Lambda} & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}\boldsymbol{V}^\top & \boldsymbol{0}\\ \boldsymbol{0} & \boldsymbol{A}^\top\end{bmatrix}$, where $\boldsymbol{A}$ can be any arbitrary $c \times c$ orthogonal matrix. Recalling that the $\boldsymbol{B}$ and $\log(\boldsymbol{B})$ have the same eigensystems, and that the trace of a matrix is the sum of its eigenvalues, we have $\operatorname{tr}(\boldsymbol{B}\log(\boldsymbol{B})) = \operatorname{tr}(\boldsymbol{\Lambda}\log(\boldsymbol{\Lambda}))$ and $\operatorname{tr}(\boldsymbol{B}'\log(\boldsymbol{B}')) = \operatorname{tr}(\boldsymbol{\Lambda}\log(\boldsymbol{\Lambda})) + c\lim_{x\to 0}(x\log(x))$. Since $\lim_{x\to 0} x\log(x) = 0$, we then have that $\operatorname{tr}(\boldsymbol{B}\log(\boldsymbol{B})) = \operatorname{tr}(\boldsymbol{B}'\log(\boldsymbol{B}'))$. This then gives the proposition. □

We also observe that $\boldsymbol{W}_1^{*,t+1}$ can be written as follows

$$\boldsymbol{W}_1^{*,t+1} = \operatorname*{argmin}_{\boldsymbol{W}:\boldsymbol{W}=\boldsymbol{Z}\boldsymbol{Z}^\top,\boldsymbol{Z}=[\boldsymbol{P};\boldsymbol{Q}]} S(\mathcal{R}_{\boldsymbol{M}}\boldsymbol{P}^\top\boldsymbol{M}\boldsymbol{P} + \mathcal{R}_{\boldsymbol{N}}\boldsymbol{Q}^\top\boldsymbol{N}\boldsymbol{Q}) + \eta\gamma \sum_{s\in[t]} f_\gamma^s(\boldsymbol{W}),$$

where $S(\boldsymbol{B}) := \operatorname{tr}(\boldsymbol{B}\log(\boldsymbol{B})) - \log(\frac{\widehat{\mathcal{D}}}{m+n})\operatorname{tr}(\boldsymbol{B}) - \operatorname{tr}(\boldsymbol{B})$, which is the negative von Neumann entropy up to a scalar factor of $\log(\frac{\widehat{\mathcal{D}}}{m+n})$. When the term $S(\mathcal{R}_{\boldsymbol{M}}\boldsymbol{P}^\top\boldsymbol{M}\boldsymbol{P} + \mathcal{R}_{\boldsymbol{N}}\boldsymbol{Q}^\top\boldsymbol{N}\boldsymbol{Q})$ is minimized, we have that $\mathcal{R}_{\boldsymbol{M}}\boldsymbol{P}^\top\boldsymbol{M}\boldsymbol{P} + \mathcal{R}_{\boldsymbol{N}}\boldsymbol{Q}^\top\boldsymbol{N}\boldsymbol{Q} = \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}$, so that $\operatorname{tr}(\mathcal{R}_{\boldsymbol{M}}\boldsymbol{P}^\top\boldsymbol{M}\boldsymbol{P} + \mathcal{R}_{\boldsymbol{N}}\boldsymbol{Q}^\top\boldsymbol{N}\boldsymbol{Q}) = \widehat{\mathcal{D}}$. This is in contrast to when the negative von Neumann entropy of a matrix $\boldsymbol{A}$ is minimized, in which case we have $\boldsymbol{A} = \boldsymbol{I}$.

3.5 Discussion

In this chapter, we presented MEG and MGD algorithms that can be applied in transductive and inductive settings. The mistake and regret bounds of the MEG algorithms show a superior scaling with the quasi-dimension $\mathcal{D}$, at the expense of a higher time complexity. Table 3.1 shows an overall comparison of the algorithms in the transductive and inductive settings. For both settings, the MGD algorithm has a low per-trial complexity of $O(\min(mn, T))$. As for the MEG algorithm, the time complexity is cubic or quartic with respect to $\max(m, n)$, depending on the setting. It remains an open problem whether the time complexity of MEG updates can be further reduced. One possibility is to approximate the update through sketching methods. Although this has been applied to vector EG algorithms such as [80], its theoretical effectiveness in the matrix case has not yet been studied in the literature and it seems to be a challenging problem. Similarly, it may be possible to apply techniques used in the budget Perceptron algorithms, such as those in [115], to lower the complexity of the MGD and inductive MEG algorithms. Another direction is to extend these results to FTRL algorithms with general convex regularizers. We believe that this should be feasible given that our analysis follows the vector case closely. The MEG algorithm requires a few parameters, such as the quasi-dimension estimate $\widehat{\mathcal{D}}$. Apart from using the doubling trick, it may be interesting to investigate whether we can use parameter-free methods (such as [116]) to re-

	Mistake Bound	Time Complexity
MEG transductive	$O(\frac{\mathcal{D}}{\gamma^2}\log(m+n))$	$O(\max(m,n)^3)$
MEG inductive		$O(\min(\max(m,n)^4, T^3))$
MGD transductive	$O(\frac{\mathcal{D}^2}{\gamma^2})$	$O(\min(mn, T))$
MGD inductive		$O(\min(mn, T))$

Table 3.1: The mistake bounds and per-trial time complexities of the general matrix completion algorithms.

move this dependence. In the following chapters, we will show how to evaluate the quasi-dimension term $\mathcal{D}$ for various examples. In particular, in Section 4.4.1 of the following chapter, we will show an example where our MEG bound is tight up to logarithmic factors.

3.6 Proofs

3.6.1 Proof of Theorem 1

We first give some preliminaries for the proof. We introduce the quantum relative entropy, which plays a central role in the amortized analysis of our algorithm.

Definition 10. *The quantum relative entropy of symmetric positive semidefinite square matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ is*

$$\Delta(\boldsymbol{A}, \boldsymbol{B}) := \operatorname{tr}(\boldsymbol{A}\log(\boldsymbol{A}) - \boldsymbol{A}\log(\boldsymbol{B}) + \boldsymbol{B} - \boldsymbol{A}).$$

We now present general inequalities for matrices. The following lemma is the well known Golden-Thompson Inequality, whose proof can be found, for example, in [71].

Lemma 11 (Golden-Thompson Inequality)**.** *For any symmetric matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ we have,*

$$\operatorname{tr}(\exp(\boldsymbol{A} + \boldsymbol{B})) \leq \operatorname{tr}(\exp(\boldsymbol{A})\exp(\boldsymbol{B}))\,.$$

Lemma 12. *For matrix $\boldsymbol{A} \in \boldsymbol{S}^d$ with eigenvalues no less than -1,*

$$\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{A}^2 - \exp(-\boldsymbol{A}) \succeq \boldsymbol{0}\,.$$

Proof. Let $\boldsymbol{B} := \boldsymbol{I} - \boldsymbol{A} + \boldsymbol{A}^2 - \exp(-\boldsymbol{A})$. Observing that $\boldsymbol{A}$, $\boldsymbol{A}^2$ and $\exp(-\boldsymbol{A})$ share the same set of eigenvectors,

$$\boldsymbol{I} - \boldsymbol{A} + \boldsymbol{A}^2 - \exp(-\boldsymbol{A}) = \boldsymbol{U}(\boldsymbol{I} - \boldsymbol{\Lambda} + \boldsymbol{\Lambda}^2 - \exp(-\boldsymbol{\Lambda}))\boldsymbol{U}^\top,$$

where $\boldsymbol{U}$ is the orthogonal matrix and $\boldsymbol{\Lambda}$ is the diagonal matrix in the eigendecomposition of $\boldsymbol{A}$. Therefore, each eigenvalue $\lambda_{\boldsymbol{B},i}$ of the resulting matrix $\boldsymbol{B}$ can be written in terms of an eigenvalue $\lambda_{\boldsymbol{B},i}$ of matrix $\boldsymbol{A}$ for all $i \in [d]$,

$$\lambda_{\boldsymbol{B},i} = 1 - \lambda_{\boldsymbol{A},i} + \lambda_{\boldsymbol{A},i}^2 - \exp(-\lambda_{\boldsymbol{A},i}).$$

The positive semidefinite criterion requires that all eigenvalues be non-negative, so that $\lambda_{\boldsymbol{B},i} \geq 0$. This inequality holds true for $\lambda_{\boldsymbol{A},i} \geq -1$. □

Lemma 13. *For any matrix $\boldsymbol{A} \in \boldsymbol{S}^d_{++}$ and any two matrices $\boldsymbol{B}, \boldsymbol{C} \in \boldsymbol{S}^d$, $\boldsymbol{B} \preceq \boldsymbol{C}$ implies* $\operatorname{tr}(\boldsymbol{AB}) \leq \operatorname{tr}(\boldsymbol{AC})$.

Proof. This follows a parallel argument to the proof for [4, Lemma 2.2]. □

Recall that $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}})$ is the "optimal" factorization which minimizes the optimization problem in $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$. Let us define the quasi-dimension with respect to this factorization

$$\mathcal{D} := \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^{\top} \boldsymbol{M} \hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^{\top} \boldsymbol{N} \hat{\boldsymbol{Q}}\right).$$

Then, we have that $\mathcal{D} = \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$.

Lemma 14. *For $\tilde{\boldsymbol{U}}$ as defined in Definition 2, we have that,*

$$\operatorname{tr}(\tilde{\boldsymbol{U}}) = \mathcal{D}. \tag{3.18}$$

Proof.

$$\begin{aligned}
\operatorname{tr}(\tilde{\boldsymbol{U}}) = \operatorname{tr}(\boldsymbol{Z}\boldsymbol{Z}^{\top}) &= \operatorname{tr}\left(\begin{pmatrix} \sqrt{\mathcal{R}_{\boldsymbol{M}}}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}} \\ \sqrt{\mathcal{R}_{\boldsymbol{N}}}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}} \end{pmatrix} \begin{pmatrix} \sqrt{\mathcal{R}_{\boldsymbol{M}}}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}} & \sqrt{\mathcal{R}_{\boldsymbol{N}}}\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}} \end{pmatrix}^{\top}\right) \\
&= \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{P}}^{\top}\sqrt{\boldsymbol{M}}^{\top}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\sqrt{\boldsymbol{N}}\hat{\boldsymbol{Q}}\hat{\boldsymbol{Q}}^{\top}\sqrt{\boldsymbol{N}}^{\top}\right) \\
&= \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^{\top}\boldsymbol{M}\hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^{\top}\boldsymbol{N}\hat{\boldsymbol{Q}}\right) \\
&= \mathcal{D}
\end{aligned}$$

□

Lemma 15. *For all trials t,* $\operatorname{tr}(\tilde{\boldsymbol{X}}^t) \leq 1$*, and all eigenvalues of $\tilde{\boldsymbol{X}}^t$ are in* $[0, 1]$.

Proof. Recall that

$$\operatorname{tr}(\tilde{\boldsymbol{X}}^t) = \operatorname{tr}(\boldsymbol{x}^t(\boldsymbol{x}^t)^{\top}) = \left[\frac{\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_{\boldsymbol{M}}}}; \frac{\sqrt{\boldsymbol{N}^+}\boldsymbol{e}_n^{j_t}}{\sqrt{2\mathcal{R}_{\boldsymbol{N}}}}\right]^{\top}\left[\frac{\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_{\boldsymbol{M}}}}; \frac{\sqrt{\boldsymbol{N}^+}\boldsymbol{e}_n^{j_t}}{\sqrt{2\mathcal{R}_{\boldsymbol{N}}}}\right].$$

Hence

$$\left\|\boldsymbol{x}^t\right\|^2 = \left\| \frac{\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_M}} \right\|^2 + \left\| \frac{\sqrt{\boldsymbol{N}^+}\boldsymbol{e}_n^{j_t}}{\sqrt{2\mathcal{R}_N}} \right\|^2$$

and then bounding the first term on the right hand side gives,

$$\left\| \frac{\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t}}{\sqrt{2\mathcal{R}_M}} \right\|^2 = \frac{1}{2\mathcal{R}_M} \left(\boldsymbol{e}_m^{i_t}\right)^\top (\sqrt{\boldsymbol{M}^+})^\top \sqrt{\boldsymbol{M}^+}\boldsymbol{e}_m^{i_t} \leq \frac{1}{2\mathcal{R}_M} \max_{i\in[m]} \left(\boldsymbol{e}_m^{i_t}\right)^\top \boldsymbol{M}^+ \boldsymbol{e}_m^{i_t} = \frac{1}{2}\,.$$

The argument for the second term is parallel. Therefore since it is shown that the trace of $\tilde{\boldsymbol{X}}^t$ is bounded by 1 and that $\tilde{\boldsymbol{X}}^t$ is positive definite, this implies that all eigenvalues of $\tilde{\boldsymbol{X}}^t$ are in $[0, 1]$. □

Proof for regret statement (Equation (3.1))

For the regret statement, we consider non-conservative updates. We recall that $\bar{\boldsymbol{U}} := \gamma\boldsymbol{U}$ as defined in (3.4). The assumption that $\|\boldsymbol{U}\|_{\max} \leq \frac{1}{\gamma}$ guarantees the existence of row-normalized matrices $\hat{\boldsymbol{P}} \in \mathfrak{R}^{m\times d}$ and $\hat{\boldsymbol{Q}} \in \mathfrak{R}^{n\times d}$ that give $\bar{\boldsymbol{U}} = \hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top$.

In the following, we recall the hinge loss as $\mathcal{L}_{\text{hi}}^\gamma(y, \bar{y}) := \frac{1}{\gamma}[\gamma - y\bar{y}]_+$. We define

$$\boldsymbol{H}^t := \nabla_{\tilde{\boldsymbol{W}}^t} \gamma\mathcal{L}_{\text{hi}}^\gamma(y_t, \bar{y}_t) = \nabla_{\tilde{\boldsymbol{W}}^t} \left[\gamma - \left(y_t \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1\right)\right]\,, \tag{3.19}$$

where ∇ denotes the subgradient and where $\bar{y}_t$ is as defined in Algorithm 1. When $y_t\bar{y}_t = \gamma$, we will only consider the specific subgradient $\boldsymbol{H}^t = \boldsymbol{0}$.

Lemma 16. *For all $t \in [T]$,*

$$\boldsymbol{H}^t = -y_t\tilde{\boldsymbol{X}}^t \left[\gamma > y_t\left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1\right)\right]\,.$$

Proof. Recalling the definition of $\boldsymbol{H}^t := \nabla_{\tilde{\boldsymbol{W}}^t}\gamma\mathcal{L}_{\text{hi}}^\gamma(y_t, \bar{y}_t)$, observe that when $\gamma > y_t\left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1\right)$, we have

$$\nabla_{\tilde{\boldsymbol{W}}^t} \gamma\mathcal{L}_{\text{hi}}^\gamma(y_t, \bar{y}_t) = \nabla_{\tilde{\boldsymbol{W}}^t} \left[\gamma - y_t\left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1\right)\right]_+ = -y_t(\tilde{\boldsymbol{X}}^t)^\top = -y_t\tilde{\boldsymbol{X}}^t, \tag{3.20}$$

where we used the fact that $\nabla_{\boldsymbol{A}} \operatorname{tr}(\boldsymbol{AB}) = \boldsymbol{B}^\top$. In the case that $\gamma \leq$

$y_t \left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t\right) - 1\right)$,

$$\nabla_{\tilde{\boldsymbol{W}}^t} \gamma \mathcal{L}_{\text{hi}}^{\gamma}(y_t, \bar{y}_t) = \mathbf{0}.$$

□

Lemma 17. *For all trials $t \in [T]$ in Algorithm 1, we have for $\eta \in (0, 1]$:*

$$\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) \geq \eta \left(\operatorname{tr}((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t) - \eta^2 \operatorname{tr}(\tilde{\boldsymbol{W}}^t (\boldsymbol{H}^t)^2)\right). \tag{3.21}$$

Proof. We have:

$$\begin{aligned}
\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) &= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{W}}^{t+1} - \tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{W}}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\right) - \operatorname{tr}\left(\tilde{\boldsymbol{W}}^{t+1}\right) \\
&= -\eta \operatorname{tr}\left(\tilde{\boldsymbol{U}} \boldsymbol{H}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\right) - \operatorname{tr}\left(e^{\log \tilde{\boldsymbol{W}}^t - \eta \boldsymbol{H}^t}\right) && (3.22) \\
&\geq -\eta \operatorname{tr}\left(\tilde{\boldsymbol{U}} \boldsymbol{H}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\right) - \operatorname{tr}\left(e^{\log \tilde{\boldsymbol{W}}^t} e^{-\eta \boldsymbol{H}^t}\right) && (3.23) \\
&= -\eta \operatorname{tr}\left(\tilde{\boldsymbol{U}} \boldsymbol{H}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t \left(\boldsymbol{I} - e^{-\eta \boldsymbol{H}^t}\right)\right) \\
&\geq -\eta \operatorname{tr}\left(\tilde{\boldsymbol{U}} \boldsymbol{H}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t \left(\eta \boldsymbol{H}^t - \eta^2 (\boldsymbol{H}^t)^2\right)\right) && (3.24)
\end{aligned}$$

where Equation (3.22) comes from substituting Lemma 16 in the update equation of the algorithm, Equation (3.23) comes from Lemma 11 and Equation (3.24) comes from Lemmas 12 and 13. □

Lemma 18. *For Algorithm 1, we have for $\eta \in (0, 1]$:*

$$\sum_{t \in [T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right) \leq \frac{1}{\eta} \left(\operatorname{tr}\left(\tilde{\boldsymbol{U}} \log\left(\frac{\tilde{\boldsymbol{U}}(m+n)}{e\widehat{\mathcal{D}}}\right)\right) + \widehat{\mathcal{D}}\right) + \sum_{t \in [T]} \eta \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t (\boldsymbol{H}^t)^2\right). \tag{3.25}$$

Setting the additional assumptions $\widehat{\mathcal{D}} \geq \mathcal{D} = \operatorname{tr}\left(\tilde{\boldsymbol{U}}\right) \geq 1$ and $m + n \geq 3$ gives

$$\sum_{t \in [T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right) \leq \frac{\widehat{\mathcal{D}}}{\eta} \log(m+n) + \sum_{t \in [T]} \eta \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t (\boldsymbol{H}^t)^2\right). \tag{3.26}$$

Proof. We start by proving Equation (3.25). Rearranging Lemma 17 and summing

over t,

$$\sum_{t\in[T]}\left(\operatorname{tr}((\tilde{\boldsymbol{W}}^t-\tilde{\boldsymbol{U}})\boldsymbol{H}^t)\right)\leq\frac{1}{\eta}\left(\sum_{t\in[T]}\Delta(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{W}}^t)-\Delta(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{W}}^{t+1})+\eta^2\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t(\boldsymbol{H}^t)^2\right)\right)$$
$$\leq\frac{1}{\eta}\left(\Delta(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{W}}^1)-\Delta(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{W}}^{T+1})+\sum_{t\in[T]}\eta^2\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t(\boldsymbol{H}^t)^2\right)\right).$$

Using the fact that $\Delta(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{W}}^{T+1})\geq 0$ and writing out $\Delta(\tilde{\boldsymbol{U}},\tilde{\boldsymbol{W}}^1)$, we then obtain Equation (3.25).

To prove Equation (3.26), we attempt to maximize the term $\operatorname{tr}\left(\tilde{\boldsymbol{U}}\log\left(\frac{\tilde{\boldsymbol{U}}(m+n)}{e\widehat{\mathcal{D}}}\right)\right)+\widehat{\mathcal{D}}$. Noting that $\operatorname{tr}\left(\tilde{\boldsymbol{U}}\log(a\tilde{\boldsymbol{U}})\right)\leq\operatorname{tr}(\tilde{\boldsymbol{U}})\log\left(\operatorname{tr}(a\tilde{\boldsymbol{U}})\right)$ for $a\geq 0$, we then have

$$\operatorname{tr}\left(\tilde{\boldsymbol{U}}\log\left(\frac{\tilde{\boldsymbol{U}}(m+n)}{e\widehat{\mathcal{D}}}\right)\right)+\widehat{\mathcal{D}}\leq\operatorname{tr}(\tilde{\boldsymbol{U}})\log\left(\frac{\operatorname{tr}(\tilde{\boldsymbol{U}})(m+n)}{e\widehat{\mathcal{D}}}\right)+\widehat{\mathcal{D}}.$$

The upper bound in the above equation is convex in $\operatorname{tr}(\tilde{\boldsymbol{U}})$ and hence is maximized at either boundary $\{1,\widehat{\mathcal{D}}\}$. Comparing the terms, we have

$$\operatorname{tr}\left(\tilde{\boldsymbol{U}}\log\left(\frac{\tilde{\boldsymbol{U}}(m+n)}{e\widehat{\mathcal{D}}}\right)\right)+\widehat{\mathcal{D}}\leq\log\left(\frac{m+n}{e\widehat{\mathcal{D}}}\right)+\widehat{\mathcal{D}}$$

for $\operatorname{tr}(\tilde{\boldsymbol{U}})=1$ and

$$\operatorname{tr}\left(\tilde{\boldsymbol{U}}\log\left(\frac{\tilde{\boldsymbol{U}}(m+n)}{e\widehat{\mathcal{D}}}\right)\right)+\widehat{\mathcal{D}}\leq\widehat{\mathcal{D}}\log(m+n)$$

for $\operatorname{tr}(\tilde{\boldsymbol{U}})=\widehat{\mathcal{D}}$. We then observe that given the assumptions, $\widehat{\mathcal{D}}\log(m+n)$ maximizes, therefore giving the upper bound in Equation (3.26).

□

Lemma 19. *For all $t\in[T]$ in Algorithm 1:*

$$\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t(\boldsymbol{H}^t)^2\right)\leq\gamma\mathcal{L}_{hi}^{\gamma}(y_t,\bar{y}_t)+\gamma+1. \tag{3.27}$$

Proof. The proof splits into two cases.

Case 1) $\gamma\leq y_t\bar{y}_t=y_t\left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right)-1\right)$:

Observe that $\boldsymbol{H}^t = \mathbf{0}$ due to Lemma 16, giving $\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t(\boldsymbol{H}^t)^2\right) = 0$ which demonstrates (3.27) in this case.

Case 2) $\gamma > y_t\bar{y}_t = y_t\left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1\right)$:

We have that

$$\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t(\boldsymbol{H}^t)^2\right) = \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t(\tilde{\boldsymbol{X}}^t)^2\right) \leq \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right), \tag{3.28}$$

where the first equality comes from the fact $\boldsymbol{H}^t = -y_t\tilde{\boldsymbol{X}}^t$ from Lemma 16 and the second inequality comes from Lemma 13 and the fact that $(\tilde{\boldsymbol{X}}^t)^2 \preceq \tilde{\boldsymbol{X}}^t$ due to Lemma 15.

We split case 2 into two further subcases.

Sub-case 1) $\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) < \gamma + 1$ (Prediction smaller than margin):

Since the hinge loss is non-negative,

$$\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) < \gamma\mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) + \gamma + 1,$$

lower bounding the L.H.S. by (3.28) demonstrates (3.27).

Sub-case 2) $\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \geq \gamma + 1$ (Prediction larger than margin with mistake):

We have

$$\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \leq \left[\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) + \gamma - 1\right]_+ - (\gamma - 1) \leq \left[\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) + \gamma - 1\right]_+ + (\gamma + 1)\,.$$

By the case 2 and sub-case 2 conditions we have that $y_t = -1$, with

$$\gamma\mathcal{L}^{\gamma}_{\text{hi}}(-1, \bar{y}_t) = \left[\gamma + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1\right]_+.$$

Thus we have

$$\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \leq \gamma\mathcal{L}^{\gamma}_{\text{hi}}(-1, \bar{y}_t) + (\gamma + 1)$$

and by lower bounding L.H.S. by (3.28) we demonstrate (3.27) and thus the lemma.

□

Lemma 20. *For Algorithm 1, assuming that* $\eta \in (0, 1]$, $\widehat{\mathcal{D}} \geq \mathcal{D} = \operatorname{tr}\left(\tilde{\boldsymbol{U}}\right) \geq 1$ *and*

$m + n \geq 3$, we have

$$\sum_{t\in[T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right) \leq \frac{1}{\eta}\widehat{\mathcal{D}}\log(m+n) + \eta\gamma \sum_{t\in[T]} \mathcal{L}^{\gamma}_{hi}(y_t, \bar{y}_t) + \eta(1+\gamma)T\,.$$

Proof. Combining (3.26) and (3.27) gives the lemma. □

Lemma 21. *For $\bar{\boldsymbol{U}} := \gamma\boldsymbol{U}$, where $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$, $\min_{i,j}|U_{ij}| \geq 1$ and $\|\boldsymbol{U}\|_{max} \leq \frac{1}{\gamma}$*

$$\mathcal{L}^{\gamma}_{hi}(y_t, \bar{U}_{i_t j_t}) \leq \left(1 + \max_{i,j}|U_{ij}|\right)[y_t \neq \operatorname{sign}(U_{i_t j_t})] \leq \left(1 + \frac{1}{\gamma}\right)[y_t \neq \operatorname{sign}(U_{i_t j_t})]$$

Proof. We first prove the first inequality. From the definition of the hinge loss and $\bar{\boldsymbol{U}}$,

$$\begin{aligned}\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{U}_{i_t j_t}) &= \mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \gamma U_{i_t j_t}) \\ &= \frac{1}{\gamma}[\gamma(1 - y_t U_{i_t j_t})]_+ \\ &= [1 - y_t U_{i_t j_t}]_+.\end{aligned}$$

In the case that $y_t U_{i_t j_t} \geq 0$, we have $\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{U}_{i_t j_t}) = [1 - |U_{i_t j_t}|]_+ = 0$ since $|U_{i_t j_t}| \geq 1$ by assumption. Otherwise, we have $\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{U}_{i_t j_t}) \leq 1 + |U_{i_t j_t}| \leq 1 + \max_{i,j}|U_{ij}|$. The first inequality follows by combining these two cases.

For the second inequality, we may further evaluate

$$\begin{aligned}\max_{i,j}|U_{ij}| &= \min_{\boldsymbol{P}\boldsymbol{Q}^\top = \boldsymbol{U}} \max_i \max_j |\langle \boldsymbol{P}_i, \boldsymbol{Q}_j\rangle| \\ &\leq \min_{\boldsymbol{P}\boldsymbol{Q}^\top = \boldsymbol{U}} \max_i \|\boldsymbol{P}_i\| \max_j \|\boldsymbol{Q}_j\| \\ &= \|\boldsymbol{U}\|_{\max}.\end{aligned}$$

Using $\|\boldsymbol{U}\|_{\max} \leq \frac{1}{\gamma}$, we then have $\max_{i,j}|U_{ij}| \leq \frac{1}{\gamma}$, from which the second inequality follows.

□

Now we are ready to introduce the regret bound in terms of the hinge loss for the deterministic $\bar{y}_t$.

Lemma 22. *The hinge loss of Algorithm 1 with parameters* $\gamma \in (0, 1]$, $\widehat{\mathcal{D}} \geq \mathcal{D} \geq 1$, $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$, $T \geq 2\widehat{\mathcal{D}}\log(m+n)$ *and* $m + n \geq 3$, *is bounded by*

$$\sum_{t\in[T]} \mathcal{L}^{\gamma}_{hi}(y_t, \bar{y}_t) \leq \sum_{t\in[T]} \mathcal{L}^{\gamma}_{hi}(y_t, \bar{U}_{i_t j_t}) + \frac{4}{\gamma}\sqrt{2\widehat{\mathcal{D}}\log(m+n)T} + \frac{4}{\gamma}\widehat{\mathcal{D}}\log(m+n)\,, \quad (3.29)$$

where we recall that $\bar{\boldsymbol{U}} := \gamma \boldsymbol{U}$ *and* $\bar{\boldsymbol{U}}_{i_t j_t} = \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) - 1$ *(see page 35).*

Proof. First, we aim to prove the bound

$$\sum_{t\in[T]} \left(\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{y}_t) - \mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{U}_{i_t j_t})\right) \leq \sum_{t\in[T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right), \quad (3.30)$$

so that we can apply Lemma 20. Recall $\boldsymbol{H}^t$ is defined as the subgradient of $\gamma\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{y}_t)$. Substituting for $\bar{y}_t$ gives,

$$\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{y}_t) = \frac{1}{\gamma}[\gamma - y_t(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) - 1)]_+ .$$

Lemma 3 gives,

$$\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{U}_{i_t j_t}) = \frac{1}{\gamma}[\gamma - y_t(\operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) - 1)]_+ .$$

Define

$$f_t(\boldsymbol{Z}) := \frac{1}{\gamma}[\gamma - y_t(\operatorname{tr}\left(\boldsymbol{Z}\tilde{\boldsymbol{X}}^t\right) - 1)]_+$$

Since $\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \cdot)$ is convex and the fact that a convex function applied to a linear function is again convex, we have that $f(\cdot)$ is convex. We have

$$\begin{aligned}
\sum_{t\in[T]} \left(\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{y}_t) - \mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{U}_{i_t j_t})\right) &= \sum_{t\in[T]} \left(f_t(\tilde{\boldsymbol{W}}^t) - f_t(\tilde{\boldsymbol{U}})\right) \\
&\leq \sum_{t\in[T]} \operatorname{tr}\left(\left(\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}}\right)\nabla f_t(\tilde{\boldsymbol{W}})\right) \quad (3.31) \\
&= \sum_{t\in[T]} \operatorname{tr}\left(\left(\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}}\right)\nabla_{\tilde{\boldsymbol{W}}^t}\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t, \bar{y}_t)\right) \\
&= \frac{1}{\gamma}\sum_{t\in[T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right), \quad (3.32)
\end{aligned}$$

where (3.31) follows from the fact that $f(\boldsymbol{A}) - f(\boldsymbol{B}) \leq \operatorname{tr}((\boldsymbol{A} - \boldsymbol{B})\nabla f(\boldsymbol{A}))$ for a con-

vex function f and symmetric matrices $\boldsymbol{A}, \boldsymbol{B}$ and (3.32) comes from the definition of $\boldsymbol{H}^t = \nabla_{\tilde{\boldsymbol{W}}} \gamma \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t)$ (see (3.19)).

Hence we can apply Lemma 20, which gives the following upper bound for $\sum_{t\in[T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right)$:

$$\sum_{t\in[T]} \operatorname{tr}\left((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t\right) \leq \frac{1}{\eta}\widehat{\mathcal{D}}\log(m+n) + \eta\gamma \sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) + \eta(1+\gamma)T\,.$$

Substituting the above into (3.32) gives,

$$\begin{aligned}
\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) &\leq \sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) + \frac{1}{\gamma}\left(\frac{1}{\eta}\widehat{\mathcal{D}}\log(m+n) + \eta\gamma \sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) + \eta(1+\gamma)T\right) \\
&= \left(\frac{1}{1-\eta}\right)\left(\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) + \frac{1}{\eta\gamma}\widehat{\mathcal{D}}\log(m+n) + \frac{\eta}{\gamma}(1+\gamma)T\right) \\
&\leq \left(\frac{1}{1-\eta}\right)\left(\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) + \frac{1}{\eta\gamma}\widehat{\mathcal{D}}\log(m+n) + \frac{2\eta}{\gamma}T\right),
\end{aligned}$$

where the final inequality follows since $\gamma \in (0, 1]$.

We observe that $\eta \in \left(0, \frac{1}{2}\right]$ due to the definition of η and the assumption on T. We apply $(1/(1-x)) \leq 1 + 2x$ for $x \in [0, 1/2]$ to obtain

$$\begin{aligned}
\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) &\leq (1+2\eta)\left(\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) + \frac{1}{\eta\gamma}\left(\widehat{\mathcal{D}}\log(m+n)\right) + \frac{2\eta}{\gamma}T\right) \\
&= \overbrace{(1+2\eta)\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t})}^{(1)} + \overbrace{\frac{1}{\eta\gamma}\widehat{\mathcal{D}}\log(m+n)}^{(2)} + \overbrace{\frac{2\eta}{\gamma}T}^{(3)} + \\
&\qquad \overbrace{\frac{2}{\gamma}\widehat{\mathcal{D}}\log(m+n)}^{(4)} + \overbrace{\frac{4\eta^2}{\gamma}T}^{(5)}
\end{aligned}$$

By substituting $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$,

$$\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) \leq \overbrace{\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t})}^{(a)} + \overbrace{\sqrt{\frac{2\widehat{\mathcal{D}}\log(m+n)}{T}} \cdot \sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t})}^{(b)} + \overbrace{\frac{4}{\gamma}\widehat{\mathcal{D}}\log(m+n)}^{(c)} + \overbrace{\frac{2}{\gamma}\sqrt{2\widehat{\mathcal{D}}\log(m+n)T}}^{(d)}$$

where (1) = (a) + (b), (2) + (3) = (d), and (4) + (5) = (c). From Lemma 21, we then have $\mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) \leq 1 + \frac{1}{\gamma} \leq \frac{2}{\gamma}$ for $t \in [T]$, giving

$$\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) - \sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) \leq \frac{4}{\gamma}\sqrt{2\widehat{\mathcal{D}}\log(m+n)T} + \frac{4}{\gamma}\widehat{\mathcal{D}}\log(m+n).$$

□

Lemma 23. *For $y_t \in \{-1, 1\}$, $\bar{y}_t \in \mathfrak{R}$, $Y_t \sim$ Uniform$(-\gamma, \gamma)$, $\gamma \in (0, 1]$ and $\hat{y}_t :=$ sign$(\bar{y}_t - Y_t)$,*

$$2\mathbb{E}[y_t \neq \hat{y}_t] \leq \mathcal{L}^{\gamma}_{hi}(y_t, \bar{y}_t).$$

Proof. We have

$$p(\hat{y}_t = 1) = \begin{cases} 0 & \text{if } \bar{y}_t \leq -\gamma \\ \frac{1}{2} + \frac{\bar{y}_t}{2\gamma} & \text{if } -\gamma < \bar{y}_t \leq \gamma \\ 1 & \text{if } \bar{y}_t > \gamma \end{cases}$$

and

$$p(\hat{y}_t = -1) = \begin{cases} 1 & \text{if } \bar{y}_t \leq -\gamma \\ \frac{1}{2} - \frac{\bar{y}_t}{2\gamma} & \text{if } -\gamma < \bar{y}_t \leq \gamma \\ 0 & \text{if } \bar{y}_t > \gamma. \end{cases}$$

The possible cases are as follows.

1. If $|\bar{y}_t| < \gamma$, $2\mathbb{E}[y_t \neq \hat{y}_t] = \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t)$. This is since if $y_t = 1$, $\mathbb{E}[y_t \neq \hat{y}_t] = \frac{1}{2} - \frac{\bar{y}_t}{2\gamma}$ and $\mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) = \frac{1}{\gamma}(\gamma - \bar{y}_t)$. Similarly if $y_t = -1$, $\mathbb{E}[y_t \neq \hat{y}_t] = \frac{1}{2} + \frac{\bar{y}_t}{2\gamma}$ and $\mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) = \frac{1}{\gamma}(\gamma + \bar{y}_t)$.

2. If $|\bar{y}_t| \geq \gamma$ and $\mathbb{E}[y_t \neq \hat{y}_t] = 0$, then $\mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) = \frac{1}{\gamma}[\gamma - |\bar{y}_t|]_+ = 0$.

3. If $|\bar{y}_t| \geq \gamma$ and $\mathbb{E}[y_t \neq \hat{y}_t] = 1$, $\mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{y}_t) = \frac{1}{\gamma}[\gamma + |\bar{y}_t|]_+ \geq \frac{2\gamma}{\gamma} = 2\mathbb{E}[y_t \neq \hat{y}_t]$.

□

Lemma 24. *The expected mistakes of Algorithm 1 with parameters $\gamma \in (0, 1]$, $\widehat{\mathcal{D}} \geq \mathcal{D} \geq 1$, $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$, and $m + n \geq 3$, is bounded by*

$$\mathbb{E}[|\mathbb{M}|] \leq \frac{1}{2}\sum_{t\in[T]} \mathcal{L}^{\gamma}_{hi}(y_t, \bar{U}_{i_t j_t}) + \frac{3.5}{\gamma}\sqrt{\widehat{\mathcal{D}}\log(m+n)T}\,, \tag{3.33}$$

where we recall that $\bar{\boldsymbol{U}} := \gamma \boldsymbol{U}$ and $\bar{\boldsymbol{U}}_{i_t j_t} = \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) - 1$ (see page 35).

Proof. Let us consider the following two cases.

1. $T \leq 9\widehat{\mathcal{D}}\log(m+n)$.

$$\begin{aligned}
\mathbb{E}[|\mathbb{M}|] &\leq T \\
&\leq \min(T, 9\widehat{\mathcal{D}}\log(m+n)) \\
&= \sqrt{\min(T, 9\widehat{\mathcal{D}}\log(m+n))^2} \\
&\leq \sqrt{9\widehat{\mathcal{D}}\log(m+n))T} \\
&\leq \frac{3}{\gamma}\sqrt{\widehat{\mathcal{D}}\log(m+n))T}
\end{aligned}$$

   where the last inequality holds since $\gamma \in (0, 1]$.

2. $T > 9\widehat{\mathcal{D}}\log(m+n)$. We apply Lemma 23 to Lemma 22, and obtain

$$\begin{aligned}
\mathbb{E}[|\mathbb{M}|] - \frac{1}{2}\sum_{t\in[T]} \mathcal{L}^{\gamma}_{\text{hi}}(y_t, \bar{U}_{i_t j_t}) &\leq \frac{2}{\gamma}\sqrt{2\widehat{\mathcal{D}}\log(m+n)T} + \frac{2}{\gamma}\widehat{\mathcal{D}}\log(m+n) \\
&\leq \frac{2}{\gamma}\sqrt{2\widehat{\mathcal{D}}\log(m+n)T} + \frac{2}{\gamma}\sqrt{(\widehat{\mathcal{D}}\log(m+n))^2} \\
&< \frac{2}{\gamma}\sqrt{2\widehat{\mathcal{D}}\log(m+n)T} + \frac{2}{3\gamma}\sqrt{\widehat{\mathcal{D}}\log(m+n)T} \\
&\leq \frac{3.5}{\gamma}\sqrt{\widehat{\mathcal{D}}\log(m+n)T}
\end{aligned} \tag{3.34}$$

where (3.34) applies due to the assumption on T.

Combining the two cases gives the lemma. □

The regret statement of the theorem then follows from Lemma 24 by applying the first inequality in Lemma 21.

Proof for the mistake bound (Equation (3.2))

In this subsection, we prove the second part of Theorem 1 (Equation (3.2)). To do so, we prove an intermediate result which implies the theorem:

Lemma 25. *The mistakes in the* **realizable** *case with* **conservative** *updates (*[NON-CONSERVATIVE] $= 0$*) and parameters* $\eta = \gamma$, $\widehat{\mathcal{D}} \geq \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ *are bounded by,*

$$|\mathbb{M}| \leq 3.6(\widehat{\mathcal{D}}/\gamma^2)\log(m+n)\,, \tag{3.35}$$

for all $\boldsymbol{U} \in ((-\infty,-1] \cup [1,\infty))^{m\times n}$ *with* $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$ *and* $y_t = \text{sign}(U_{i_t j_t})$ *for all* $t \in \mathbb{M}$.

We now recall the second part of the theorem and show that Lemma 25 implies this.

> The mistakes in the **realizable** case with **conservative** updates ([NON-CONSERVATIVE] $= 0$) and parameters $\eta = \gamma$, $\widehat{\mathcal{D}} \geq \min_{\boldsymbol{V}\in \text{SP}^1(\boldsymbol{U})} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V})$ are bounded by,
>
> $$|\mathbb{M}| \leq 3.6(\widehat{\mathcal{D}}/\gamma^2)\log(m+n)\,,$$
>
> for all $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$ with $\text{mc}(\boldsymbol{U}) \leq 1/\gamma$ and $y_t = \text{sign}(U_{i_t j_t})$ for all $t \in \mathbb{M}$.

Since Lemma 25 holds for all $\boldsymbol{U} \in ((-\infty,-1] \cup [1,\infty))^{m\times n}$, it also holds for the comparator matrix $\boldsymbol{V}^* \in \text{argmin}_{\boldsymbol{V}\in \text{SP}^1(\boldsymbol{U}')} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V})$ where $\boldsymbol{U}' \in \mathfrak{R}^{m\times n}$. We have $y_t = \text{sign}(V^*_{i_t j_t}) = \text{sign}(U'_{i_t j_t})$ for all t. It now remains to be shown that $\|\boldsymbol{V}^*\|_{\max} \leq 1/\gamma$ is equivalent to $\text{mc}(\boldsymbol{U}') \leq 1/\gamma$. It is easy to show that $\|\boldsymbol{V}^*\|_{\max} \leq 1/\gamma$ implies $\text{mc}(\boldsymbol{U}') \leq 1/\gamma$ since we have $\text{mc}(\boldsymbol{U}') \leq \|\boldsymbol{V}^*\|_{\max} \leq 1/\gamma$. To show that $\text{mc}(\boldsymbol{U}') \leq 1/\gamma$ implies $\|\boldsymbol{V}^*\|_{\max} \leq 1/\gamma$, we observe that $\|\boldsymbol{V}^*\|_{\max} \leq 1/\gamma$ iff there

exist $\hat{\boldsymbol{P}}$ and $\hat{\boldsymbol{Q}}$ such that $\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \frac{1}{\gamma}\boldsymbol{V}^*$. We now show that $\mathrm{mc}(\boldsymbol{U}') \leq 1/\gamma$ implies the existence of such a decomposition for $\boldsymbol{V}^*$. The condition $\mathrm{mc}(\boldsymbol{U}') \leq 1/\gamma$ implies that there exists a $\boldsymbol{V}' \in \mathrm{SP}^1(\boldsymbol{U}')$ such that $\|\boldsymbol{V}'\|_{\max} \leq 1/\gamma$. Therefore, there exist $\hat{\boldsymbol{P}}$ and $\hat{\boldsymbol{Q}}$ such that $\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \frac{1}{\gamma}\boldsymbol{V}'$ and $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}')$ is finite. Since $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}^*) := \min_{\boldsymbol{V}\in\mathrm{SP}^1(\boldsymbol{U})} \mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}) \leq \mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}')$, we have that $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}^*)$ must be finite as well, thus implying the existence of a decomposition.

In the remainder of the section, we provide the proof for Lemma 25.

Proof of Lemma 25

Lemma 26. *[4, Lemma 2.1] If $\boldsymbol{A} \in \boldsymbol{S}^d_+$ with eigenvalues in $[0,1]$ and $a \in \mathfrak{R}$ then:*

$$(1 - e^a)\,\boldsymbol{A} \preceq \boldsymbol{I} - \exp(a\boldsymbol{A})$$

Lemma 27. *For all trials $t \in \mathbb{M}$, we have:*

$$\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) \geq \eta y_t \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + (1 - e^{\eta y_t}) \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right). \tag{3.36}$$

Proof. We have:

$$\begin{aligned}
\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) &= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{W}}^{t+1} - \tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{W}}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\right) - \operatorname{tr}\left(\tilde{\boldsymbol{W}}^{t+1}\right) \\
&= \eta y_t \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\right) - \operatorname{tr}\left(e^{\log \tilde{\boldsymbol{W}}^t + \eta y_t \tilde{\boldsymbol{X}}^t}\right) && (3.37) \\
&\geq \eta y_t \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\right) - \operatorname{tr}\left(e^{\log \tilde{\boldsymbol{W}}^t} e^{\eta y_t \tilde{\boldsymbol{X}}^t}\right) && (3.38) \\
&= \eta y_t \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\left(\boldsymbol{I} - e^{\eta y_t \tilde{\boldsymbol{X}}^t}\right)\right) \\
&\geq \eta y_t \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + (1 - e^{\eta y_t}) \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right), && (3.39)
\end{aligned}$$

where Equation (3.37) comes from the update of the algorithm, Equation (3.38) comes from Lemma 11 and Equation (3.39) comes from Lemma 26 which applies since, by Lemma 15 all eigenvalues of $\tilde{\boldsymbol{X}}^t$ are in $[0,1]$. □

Lemma 28. *[34, Lemma A.5] For $x \in [-1,1]$,*

$$x^2 + x + 1 - e^x \geq (3 - e)x^2\,.$$

We proceed by showing that the "progress" $\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1})$ of $\tilde{\boldsymbol{W}}^t$ towards $\tilde{\boldsymbol{U}}$ may be further lower bounded by $c\gamma$ (see Lemma 29).

Lemma 29. *Let $c := 3 - e$. For all trials t with $t \in \mathbb{M}$ (under the conditions of Lemma 32) we have:*

$$\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) \geq c\gamma^2$$

Proof. By Lemma 3, $\bar{U}_{i_t j_t} = \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) - 1$ so since $y_t = \operatorname{sign}(U_{i_t,j_t}) = \operatorname{sign}(\bar{U}_{i_t j_t})$, and $\gamma \leq \min_{i_t,j_t} |\bar{U}_{i_t j_t}|$ by the constraints on $\boldsymbol{U}$, we have $\gamma \leq y_t\left(\operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) - 1\right)$. So when $y_t = 1$ we have $\operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) \geq 1 + \gamma$ and when $y_t = -1$ we have $\operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) \leq 1 - \gamma$. We use these inequalities as follows.

First suppose that $y_t = 1$. By Lemma 27 we have:

$$\begin{aligned}
\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) &\geq \gamma \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + (1 - e^{\gamma}) \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \\
&\geq \gamma(1 + \gamma) + (1 - e^{\gamma}) \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \\
&\geq \gamma(1 + \gamma) + (1 - e^{\gamma}) \qquad (3.40) \\
&= (\gamma + \gamma^2) + 1 - e^{\gamma} \\
&\geq c\gamma^2, \qquad (3.41)
\end{aligned}$$

where Equation (3.41) comes from Lemma 28 and Equation (3.40) comes from the fact that $\hat{y}_t = -1$ and hence, by the algorithm, $\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \leq 1$.

Now suppose that $y_t = -1$. By Lemma 27 we have:

$$\begin{aligned}
\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1}) &\geq -\gamma \operatorname{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right) + (1 - e^{-\gamma}) \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \\
&\geq -\gamma(1 - \gamma) + (1 - e^{-\gamma}) \operatorname{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right) \\
&\geq -\gamma(1 - \gamma) + (1 - e^{-\gamma}) \qquad (3.42) \\
&= -\gamma + \gamma^2 + 1 - e^{-\gamma} \\
&\geq c\gamma^2, \qquad (3.43)
\end{aligned}$$

where Equation (3.43) comes from Lemma 28 and Equation (3.42) comes from the

fact that $\hat{y}_t = 1$ and hence, by the algorithm, $\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t\right) \geq 1$. □

Lemma 30. *We have,*

$$c\gamma^2|\mathbb{M}| \leq \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^1)\,.$$

Proof. Suppose that we have T trials. Then we have:

$$\begin{aligned}
\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^1) &\geq \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^1) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{T+1}) \\
&= \sum_{t\in[T]} \left(\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1})\right) \\
&= \sum_{t\in\mathbb{M}} \left(\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^t) - \Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^{t+1})\right) \qquad (3.44) \\
&\geq \sum_{t\in\mathbb{M}} c\gamma^2 \qquad (3.45) \\
&= c\gamma^2|\mathbb{M}|,
\end{aligned}$$

where (3.45) follows from (3.44) using Lemma 29. □

Lemma 31. *Given that* $\tilde{\boldsymbol{W}}^1 = \widehat{\mathcal{D}}\frac{\boldsymbol{I}}{m+n}$ *we have*

$$\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^1) \leq \operatorname{tr}\left(\tilde{\boldsymbol{U}}\right)\log(m+n) + \operatorname{tr}(\tilde{\boldsymbol{U}})\log\frac{\operatorname{tr}(\tilde{\boldsymbol{U}})}{\widehat{\mathcal{D}}} + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}})$$

Proof. We have:

$$\begin{aligned}
\Delta(\tilde{\boldsymbol{U}}, \tilde{\boldsymbol{W}}^1) &= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{U}}\right) - \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{W}}^1\right) + \operatorname{tr}\left(\tilde{\boldsymbol{W}}^1\right) - \operatorname{tr}(\tilde{\boldsymbol{U}}) \\
&= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{U}}\right) - \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}\right)\right) + \operatorname{tr}\left(\frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}\right) - \operatorname{tr}(\tilde{\boldsymbol{U}}) \\
&= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{U}}\right) - \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}\right)\right) + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}}) \\
&= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{U}}\right) - \operatorname{tr}\left(\tilde{\boldsymbol{U}} \left(\boldsymbol{I} \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\right)\right)\right) + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}}) \\
&= \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{U}}\right) - \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\right)\right) + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}}) \qquad (3.46) \\
&\leq \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log(\operatorname{tr}(\tilde{\boldsymbol{U}}))\right) - \operatorname{tr}\left(\tilde{\boldsymbol{U}} \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\right)\right) + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}}) \qquad (3.47) \\
&= \left(\log(\operatorname{tr}(\tilde{\boldsymbol{U}})) - \log\left(\frac{\widehat{\mathcal{D}}}{m+n}\right)\right)\operatorname{tr}(\tilde{\boldsymbol{U}}) + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}}) \\
&= \log\left(\frac{\operatorname{tr}(\tilde{\boldsymbol{U}})(m+n)}{\widehat{\mathcal{D}}}\right)\operatorname{tr}(\tilde{\boldsymbol{U}}) + \widehat{\mathcal{D}} - \operatorname{tr}(\tilde{\boldsymbol{U}}),
\end{aligned}$$

where (3.47) follows from (3.46), since $\tilde{\boldsymbol{U}} := \boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}$ where $\boldsymbol{\Lambda}$ is a diagonal matrix of the eigenvalues of $\tilde{\boldsymbol{U}}$. This holds since,

$$\begin{aligned}
\operatorname{tr}(\tilde{\boldsymbol{U}} \log \tilde{\boldsymbol{U}}) &= \operatorname{tr}(\boldsymbol{V}\boldsymbol{\Lambda}\boldsymbol{V}^{-1}\boldsymbol{V} \log \boldsymbol{\Lambda}\boldsymbol{V}^{-1}) \\
&= \operatorname{tr}(\boldsymbol{V}\boldsymbol{\Lambda} \log \boldsymbol{\Lambda}\boldsymbol{V}^{-1}) \\
&= \operatorname{tr}(\boldsymbol{\Lambda} \log \boldsymbol{\Lambda}) \\
&= \sum_{i=1}^{m+n} \lambda_i \log(\lambda_i) \\
&\leq (\sum_{i=1}^{m+n} \lambda_i) \log(\sum_{i=1}^{m+n} \lambda_i) \\
&= \operatorname{tr}(\tilde{\boldsymbol{U}} \log(\operatorname{tr}(\tilde{\boldsymbol{U}}))).
\end{aligned}$$

□

Lemma 32. *The mistakes, $|\mathbb{M}|$, of Algorithm 1 with the assumption that $y_t = \operatorname{sign}(U_{i_t j_t})$ for all $t \in \mathbb{M}$ and with parameters $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$, $1 \leq \widehat{\mathcal{D}}$ and $\eta = \gamma$*

and **conservative** *updates, is bounded above by:*

$$|\mathbb{M}| \leq 3.6\frac{1}{\gamma^2}\left(\mathcal{D}\left(\log(m+n) + \log\frac{\mathcal{D}}{\widehat{\mathcal{D}}}\right) + \widehat{\mathcal{D}} - \mathcal{D}\right) \tag{3.48}$$

Proof. Combining Lemmas 30 and 31 gives us

$$|\mathbb{M}| \leq \frac{1}{c}\frac{1}{\gamma^2}\left(\mathrm{tr}\left(\tilde{\boldsymbol{U}}\right)\log(m+n) + \mathrm{tr}(\tilde{\boldsymbol{U}})\log\frac{\mathrm{tr}(\tilde{\boldsymbol{U}})}{\widehat{\mathcal{D}}} + \widehat{\mathcal{D}} - \mathrm{tr}(\tilde{\boldsymbol{U}})\right)$$

Using Lemma 14 and upper bounding $1/c$ by 3.6 then gives the result. □

The theorem statement for the realizable case then follows by setting $\widehat{\mathcal{D}} \geq \mathcal{D}$.

■

3.6.2 Proof of Theorem 4

In this subsection we provide the proofs for the regret and mistake bound statements for the MGD algorithm. For the mistake bound, we prove the following lemma in the remainder of the subsection, which implies the bound (see discussion following Lemma 25, where we can replace the quasi-dimension by the quasi-area).

Lemma 33. *The mistakes in the* **realizable** *case with* ***conservative*** *updates ([*NON-CONSERVATIVE*]=0) and learning rate* $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{|\mathbb{M}|}}$*, where* $\widehat{\mathcal{D}}^2 \geq \mathcal{A}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$*, are bounded by,*

$$|\mathbb{M}| \leq \frac{\widehat{\mathcal{D}}^2}{\gamma^2}\,,$$

for all $\boldsymbol{U} \in ((-\infty,-1]\cup[1,\infty))^{m\times n}$ *with* $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$ *and* $y_t = \mathrm{sign}(U_{i_t j_t})$ *for all* $t \in \mathbb{M}$*.*

For all $t \in [T]$, we define

$$\boldsymbol{H}^t := \gamma\,\nabla_{\tilde{\boldsymbol{W}}^t}\mathcal{L}^{\gamma}_{\mathrm{hi}}(y_t,\bar{y}_t)\,[\gamma \times [\text{NON-CONSERVATIVE}] > y_t\bar{y}_t]\,, \tag{3.49}$$

where ∇ denotes the subgradient and we recall that $\bar{y}_t = \mathrm{tr}\left(\tilde{\boldsymbol{W}}^t(\tilde{\boldsymbol{X}}^t)^\top\right)$, as defined in Algorithm 2. We also define $\bar{\boldsymbol{U}} := \gamma\boldsymbol{U}$ for $\boldsymbol{U} \in ((-\infty,-1]\cup[1,\infty))^{m\times n}$ and recall the following lemmas from Section 3.6.1.

Lemma 21. *For* $\bar{\boldsymbol{U}} := \gamma \boldsymbol{U}$*, where* $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$*,* $\min_{i,j} |U_{ij}| \geq 1$ *and* $\|\boldsymbol{U}\|_{max} \leq \frac{1}{\gamma}$

$$\mathcal{L}_{hi}^{\gamma}(y_t, \bar{U}_{i_t j_t}) \leq \left(1 + \max_{i,j} |U_{ij}|\right) [y_t \neq \operatorname{sign}(U_{i_t j_t})] \leq \left(1 + \frac{1}{\gamma}\right) [y_t \neq \operatorname{sign}(U_{i_t j_t})]$$

Lemma 23. *For* $y_t \in \{-1, 1\}$*,* $\bar{y}_t \in \mathfrak{R}$*,* $Y_t \sim \textsc{Uniform}(-\gamma, \gamma)$*,* $\gamma \in (0, 1]$ *and* $\hat{y}_t := \operatorname{sign}(\bar{y}_t - Y_t)$*,*

$$2\mathbb{E}[y_t \neq \hat{y}_t] \leq \mathcal{L}_{hi}^{\gamma}(y_t, \bar{y}_t).$$

Lemma 34. *For all* $t \in [T]$*,*

$$\boldsymbol{H}^t = -y_t \tilde{\boldsymbol{X}}^t \left[\gamma \times [\textsc{non-conservative}] > y_t \bar{y}_t\right].$$

Proof. When $\gamma \times [\textsc{non-conservative}] > y_t \bar{y}_t$, we have that $\boldsymbol{H}^t = \nabla_{\tilde{\boldsymbol{W}}^t} \gamma \mathcal{L}_{\text{hi}}^{\gamma}(y_t, \bar{y}_t)$ and hence

$$\boldsymbol{H}^t = \nabla_{\tilde{\boldsymbol{W}}^t} \left[\gamma - y_t \left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t\right) - 1\right)\right]_+ = \nabla_{\tilde{\boldsymbol{W}}^t} \left(\gamma - y_t \left(\operatorname{tr}\left(\tilde{\boldsymbol{W}}^t \tilde{\boldsymbol{X}}^t\right) - 1\right)\right) = -y_t \tilde{\boldsymbol{X}}^t.$$

When $\gamma \times [\textsc{non-conservative}] \leq y_t \bar{y}_t$, $\boldsymbol{H}^t = \boldsymbol{0}$ by definition. □

Recall that $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}})$ is the "optimal" factorization which minimizes the optimization problem in the definition of $\mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma}(\boldsymbol{U})$. Let us define the quasi-area with respect to this factorization as

$$\mathcal{A} := \mathcal{R}_{\boldsymbol{M}} \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M} \hat{\boldsymbol{P}}\right) \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N} \hat{\boldsymbol{Q}}\right).$$

Then, we have that $\mathcal{A} = \mathcal{A}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma}(\boldsymbol{U})$.

Lemma 35.

$$\operatorname{tr}(\tilde{\boldsymbol{U}} \tilde{\boldsymbol{U}}^\top) \leq \mathcal{A}$$

Proof. We start by proving the equality. Recall that $\tilde{\boldsymbol{U}} = \sqrt{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top\sqrt{\boldsymbol{N}}$.

$$\operatorname{tr}(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{U}}^\top) = \mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}\operatorname{tr}(\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top\boldsymbol{N}\hat{\boldsymbol{Q}}\hat{\boldsymbol{P}}^\top\sqrt{\boldsymbol{M}})$$

from which the equality follows by the cyclic property of the trace. For the inequality, we bound

$$\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}\operatorname{tr}(\hat{\boldsymbol{P}}^\top\boldsymbol{M}\hat{\boldsymbol{P}}\cdot\hat{\boldsymbol{Q}}^\top\boldsymbol{N}\hat{\boldsymbol{Q}}) \le \mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}\operatorname{tr}(\hat{\boldsymbol{P}}^\top\boldsymbol{M}\hat{\boldsymbol{P}})\operatorname{tr}(\hat{\boldsymbol{Q}}^\top\boldsymbol{N}\hat{\boldsymbol{Q}})$$

which follows since $\operatorname{tr}(\boldsymbol{AB}) \le \operatorname{tr}(\boldsymbol{A})\operatorname{tr}(\boldsymbol{B})$ for $\boldsymbol{A}, \boldsymbol{B} \in \boldsymbol{S}_+$. □

Lemma 36. *For all $t \in [T]$,*

$$\operatorname{tr}(\tilde{\boldsymbol{X}}^t(\tilde{\boldsymbol{X}}^t)^\top) \le 1.$$

Proof. Recall that

$$\tilde{\boldsymbol{X}}^t = \frac{1}{\sqrt{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}}\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_{i_t}(\boldsymbol{e}_{j_t})^\top\sqrt{\boldsymbol{N}^+}.$$

Evaluating

$$\begin{aligned}
\operatorname{tr}(\tilde{\boldsymbol{X}}^t(\tilde{\boldsymbol{X}}^t)^\top) &= \frac{1}{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}\operatorname{tr}(\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_{i_t}(\boldsymbol{e}_{j_t})^\top\boldsymbol{N}^+\boldsymbol{e}_{j_t}(\boldsymbol{e}_{i_t})^\top\sqrt{\boldsymbol{N}^+}) \\
&= \frac{1}{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}(\boldsymbol{e}_{i_t})^\top\boldsymbol{M}^+\boldsymbol{e}_{i_t}(\boldsymbol{e}_{j_t})^\top\boldsymbol{N}^+\boldsymbol{e}_{j_t} \\
&= \frac{1}{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}M^+_{i_t,i_t}N^+_{j_t,j_t} \\
&\le 1,
\end{aligned}$$

where the inequality comes from the definition of the squared radius: $\mathcal{R}_{\boldsymbol{M}} := \max_{i\in[m]} M^+_{ii}$. □

Lemma 37. *For all $t \in [T]$,*

$$\operatorname{tr}(\tilde{\boldsymbol{U}}(\tilde{\boldsymbol{X}}^t)^\top) = \bar{U}_{i_t,j_t}.$$

Proof. Recall that $\tilde{\boldsymbol{U}} = \sqrt{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top\sqrt{\boldsymbol{N}}$ and $\tilde{\boldsymbol{X}}^t = \frac{1}{\sqrt{\mathcal{R}_{\boldsymbol{M}}\mathcal{R}_{\boldsymbol{N}}}}\sqrt{\boldsymbol{M}^+}\boldsymbol{e}_{i_t}(\boldsymbol{e}_{j_t})^\top\sqrt{\boldsymbol{N}^+}$.

Then we have

$$\begin{aligned}\operatorname{tr}(\tilde{\boldsymbol{U}}(\tilde{\boldsymbol{X}}^t)^\top) &= \operatorname{tr}(\sqrt{\boldsymbol{M}}\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top \boldsymbol{e}_{j_t}(\boldsymbol{e}_{i_t})^\top \sqrt{\boldsymbol{M}^+}) \\ &= \operatorname{tr}(\boldsymbol{e}_{i_t}^\top \hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top \boldsymbol{e}_{j_t}) \\ &= (\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top)_{i_t,j_t}\end{aligned}$$

□

We proceed with the typical progress inequality for MGD. For a matrix $\boldsymbol{A}$, we define the $\|\boldsymbol{A}\|_F$ as the Frobenius norm, given by $\sqrt{\operatorname{tr}(\boldsymbol{A}\boldsymbol{A}^\top)}$.

Lemma 38. *For all $t \in [T]$,*

$$\|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^t\|_F^2 - \|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^{t+1}\|_F^2 = -\eta^2 \operatorname{tr}((\boldsymbol{H}^t)^2) + 2\eta \operatorname{tr}((\tilde{\boldsymbol{W}}^t - \boldsymbol{U})^\top \boldsymbol{H}^t).$$

Proof. Recalling the update rule and by Lemma 34, we have that for both conservative and non-conservative updates, $\tilde{\boldsymbol{W}}^{t+1} = \tilde{\boldsymbol{W}}^t - \eta\boldsymbol{H}^t$. Hence,

$$\begin{aligned}\|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^{t+1}\|_F^2 &= \langle \tilde{\boldsymbol{U}} - (\tilde{\boldsymbol{W}}^t - \eta\boldsymbol{H}^t), \tilde{\boldsymbol{U}} - (\tilde{\boldsymbol{W}}^t - \eta\boldsymbol{H}^t)\rangle \\ &= \|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^t\|_F^2 + \eta^2 \operatorname{tr}(\boldsymbol{H}^t(\boldsymbol{H}^t)^\top) - 2\eta \operatorname{tr}((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})^\top \boldsymbol{H}^t). \quad (3.50)\end{aligned}$$

The lemma then follows by rearranging Equation (3.50). □

Lemma 39. *For Algorithm 2 with non-conservative updates, assuming that $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{T}}$ and $\widehat{\mathcal{D}}^2 \geq \mathcal{A} \geq 1$,*

$$\sum_{t\in[T]} \operatorname{tr}((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})^\top \boldsymbol{H}^t) \leq \sqrt{\widehat{\mathcal{D}}^2 T}$$

In the case of conservative updates, assuming that $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{|\mathbb{M}|}}$ and $\widehat{\mathcal{D}}^2 \geq \mathcal{A} \geq 1$,

$$\sum_{t\in\mathbb{M}} \operatorname{tr}((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})^\top \boldsymbol{H}^t) \leq \sqrt{\widehat{\mathcal{D}}^2 |\mathbb{M}|}$$

Proof. We prove for the case of conservative updates. The case of non-conservative updates can be proven in the same manner by summing over $[T]$ instead of $\mathbb{M}$. Rearranging the equation in Lemma 38 and summing over $t \in \mathbb{M}$ (while noting that for conservative updates $\boldsymbol{H}^t = \mathbf{0}$ when $t \neq |\mathbb{M}|$) gives

$$\begin{aligned}
\sum_{t\in\mathbb{M}} \operatorname{tr}((\tilde{\boldsymbol{W}}^t - \tilde{\boldsymbol{U}})\boldsymbol{H}^t) &= \frac{1}{2\eta}\left(\|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^1\|_F^2 - \|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^{|\mathbb{M}|+1}\|_F^2\right) + \eta^2 \sum_{t\in\mathbb{M}} \operatorname{tr}(\boldsymbol{H}^t(\boldsymbol{H}^t)^\top) \\
&\leq \frac{1}{2\eta}\left(\|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^1\|_F^2 + \eta^2 \sum_{t\in\mathbb{M}} \operatorname{tr}(\boldsymbol{H}^t(\boldsymbol{H}^t)^\top)\right) && (3.51) \\
&= \frac{1}{2\eta}\left(\operatorname{tr}(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{U}}^\top) + \eta^2 \sum_{t\in\mathbb{M}} \operatorname{tr}(\boldsymbol{H}^t(\boldsymbol{H}^t)^\top)\right) && (3.52) \\
&= \frac{1}{2\eta}\left(\operatorname{tr}(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{U}}^\top) + \eta^2 \sum_{t\in\mathbb{M}} \operatorname{tr}(\tilde{\boldsymbol{X}}^t(\tilde{\boldsymbol{X}}^t)^\top)\right) && (3.53) \\
&\leq \frac{1}{2\eta}\left(\widehat{\mathcal{D}}^2 + \eta^2|\mathbb{M}|\right) && (3.54) \\
&= \frac{1}{2\eta}\widehat{\mathcal{D}}^2 + \frac{\eta}{2}|\mathbb{M}|
\end{aligned}$$

where (3.51) comes from the fact that $\|\tilde{\boldsymbol{U}} - \tilde{\boldsymbol{W}}^{|\mathbb{M}|+1}\|_F^2 \geq 0$, (3.52) follows from $\tilde{\boldsymbol{W}}^1 := \mathbf{0}$, (3.53) comes from Lemma 16 and (3.54) comes from Lemmas 35 and 36. The lemma follows by setting the η from the assumption. □

Lemma 40. *For Algorithm 2 with non-conservative updates, assuming that* $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{T}}$ *and* $\widehat{\mathcal{D}}^2 \geq \mathcal{A} \geq 1$

$$\sum_{t\in[T]} \mathcal{L}_{hi}^{\gamma}(y_t, \bar{y}_t) - \sum_{t\in[T]} \mathcal{L}_{hi}^{\gamma}(y_t, \bar{U}_{i_t j_t}) \leq \frac{1}{\gamma}\sqrt{\widehat{\mathcal{D}}^2 T} \tag{3.55}$$

where we recall that $\bar{\boldsymbol{U}} := \gamma \boldsymbol{U}$. *In the case of conservative updates,* $\eta = \sqrt{\frac{\widehat{\mathcal{D}}^2}{|\mathbb{M}|}}$ *and* $\widehat{\mathcal{D}}^2 \geq \mathcal{A} \geq 1$

$$\sum_{t\in\mathbb{M}} \mathcal{L}_{hi}^{\gamma}(y_t, \bar{y}_t) - \sum_{t\in\mathbb{M}} \mathcal{L}_{hi}^{\gamma}(y_t, \bar{U}_{i_t j_t}) \leq \frac{1}{\gamma}\sqrt{\widehat{\mathcal{D}}^2|\mathbb{M}|}. \tag{3.56}$$

Proof. We prove the case for the conservative updates. The case for non-conservative updates can be proven in a similar manner by using $[T]$ instead of

$\mathbb{M}$. We aim to prove

$$\sum_{t\in\mathbb{M}}\left(\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)-\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{U}_{i_tj_t})\right)\leq\frac{1}{\gamma}\sum_{t\in[T]}\text{tr}\left((\tilde{\boldsymbol{W}}^t-\tilde{\boldsymbol{U}})\boldsymbol{H}^t\right)$$

so that the lemma follows from Lemma 39. For $t\in\mathbb{M}$, we have that $\boldsymbol{H}^t=\nabla_{\tilde{\boldsymbol{W}}}\gamma\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)$ since $y_t\bar{y}_t<0$. Substituting for $\bar{y}_t$ gives,

$$\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)=\frac{1}{\gamma}[\gamma-y_t(\text{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t\right)-1)]_+\,.$$

Lemma 3 gives,

$$\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{U}_{i_tj_t})=\frac{1}{\gamma}[\gamma-y_t(\text{tr}\left(\tilde{\boldsymbol{U}}\tilde{\boldsymbol{X}}^t\right)-1)]_+\,.$$

Define

$$f_t(\boldsymbol{Z}):=\frac{1}{\gamma}[\gamma-y_t(\text{tr}\left(\boldsymbol{Z}\tilde{\boldsymbol{X}}^t\right)-1)]_+$$

Since $\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\cdot)$ is convex and the fact that a convex function applied to a linear function is again convex, we have that $f(\cdot)$ is convex. We have

$$\begin{aligned}
\sum_{t\in\mathbb{M}}\left(\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)-\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{U}_{i_tj_t})\right)&=\sum_{t\in\mathbb{M}}\left(f_t(\tilde{\boldsymbol{W}}^t)-f_t(\tilde{\boldsymbol{U}})\right)\\
&\leq\sum_{t\in\mathbb{M}}\text{tr}\left(\left(\tilde{\boldsymbol{W}}^t-\tilde{\boldsymbol{U}}\right)^{\top}\nabla f_t(\tilde{\boldsymbol{W}})\right) \qquad (3.57)\\
&=\sum_{t\in\mathbb{M}}\text{tr}\left(\left(\tilde{\boldsymbol{W}}^t-\tilde{\boldsymbol{U}}\right)^{\top}\nabla_{\tilde{\boldsymbol{W}}}\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)\right)\\
&=\frac{1}{\gamma}\sum_{t\in\mathbb{M}}\text{tr}\left((\tilde{\boldsymbol{W}}^t-\tilde{\boldsymbol{U}})^{\top}\boldsymbol{H}^t\right), \qquad (3.58)
\end{aligned}$$

where (3.57) follows from the fact that $f(\boldsymbol{A})-f(\boldsymbol{B})\leq\text{tr}((\boldsymbol{A}-\boldsymbol{B})^{\top}\nabla f(\boldsymbol{A}))$ for a convex function f and (3.58) comes from the definition of $\boldsymbol{H}^t=\nabla_{\tilde{\boldsymbol{W}}}\gamma\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)$ (see (3.49)). □

The regret statement of the theorem follows from Equation (3.55) in Lemma 40 by bounding $\sum_{t\in[T]}\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{y}_t)$ and $\sum_{t\in[T]}\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{U}_{i_tj_t})$ through Lemma 23 and the first inequality of Lemma 21 respectively. For the mistake bound in the realizable case, we use Equation (3.56) in Lemma 40 and observe that $\sum_{t\in\mathbb{M}}\mathcal{L}^{\gamma}_{\text{hi}}(y_t,\bar{U}_{i_tj_t})=0$, from

which Lemma 33 follows. ■

3.6.3 Proof of Proposition 6

We now prove Proposition 6, that the transductive and inductive algorithms are equivalent. Recall by assumption that $\mathcal{R}_M = \mathcal{R}_M$ and $\mathcal{R}_N = \mathcal{R}_N$.

Equivalence of Traces

Suppose, in this subsection, that we have some given trial t. In this subsection we analyse the inductive algorithm. We make the following definitions:

Definition 41. *For all* $s \in \mathbb{U} \cap [t]$

$$\boldsymbol{v}(s) := \left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\boldsymbol{e}^{i_s}}{\sqrt{2\mathcal{R}_M}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\boldsymbol{e}^{j_s}}{\sqrt{2\mathcal{R}_N}} \right]$$

$$\bar{\boldsymbol{v}}(s) := \left[\frac{(\sqrt{(\boldsymbol{M}^{T+1})^+})\boldsymbol{e}^{i_s}}{\sqrt{2\mathcal{R}_M}}; \frac{(\sqrt{(\boldsymbol{N}^{T+1})^+})\boldsymbol{e}^{j_s}}{\sqrt{2\mathcal{R}_N}} \right]$$

Note that $\tilde{\boldsymbol{X}}^t(s) = \boldsymbol{v}(s)\boldsymbol{v}(s)^\top$ and $\tilde{\boldsymbol{X}}^{T+1}(s) = \bar{\boldsymbol{v}}(s)\bar{\boldsymbol{v}}(s)^\top$ for $s \in \mathbb{U} \cap [t]$.

Lemma 42. *For all* $l \in \mathbb{N}$ *and for all* $a_1, a_2, ..., a_l \in \mathbb{U} \cap [t-1]$ *there exists some* $\alpha \in \mathfrak{R}$ *such that:*

$$\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_l) = \alpha\boldsymbol{v}(a_1)\boldsymbol{v}(a_l)^\top$$

and

$$\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_l) = \alpha\bar{\boldsymbol{v}}(a_1)\bar{\boldsymbol{v}}(a_l)^\top$$

Proof. We prove by induction on l. In the case $l := 1$ the result is clear with $\alpha := 1$.

Now suppose the result holds with $l := q$ for some $q \in \mathbb{N}$. We now show that it holds for $l := q+1$. Since it holds for $l := q$, choose α' such that $\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_q) = \alpha'\boldsymbol{v}(a_1)\boldsymbol{v}(a_q)^\top$ and $\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_q) =$

$\alpha' \bar{\boldsymbol{v}}(a_1)\bar{\boldsymbol{v}}(a_q)^\top$. Note that we now have:

$$\begin{aligned}
\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_l) &= \tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_q)\tilde{\boldsymbol{X}}^t(a_l) \\
&= \alpha' \boldsymbol{v}(a_1)\boldsymbol{v}(a_q)^\top \tilde{\boldsymbol{X}}^t(a_l) \\
&= \alpha' \boldsymbol{v}(a_1)\boldsymbol{v}(a_q)^\top \boldsymbol{v}(a_l)\boldsymbol{v}(a_l)^\top \\
&= \left(\boldsymbol{v}(a_q)^\top \boldsymbol{v}(a_l)\right) \alpha' \boldsymbol{v}(a_1)\boldsymbol{v}(a_l)^\top \\
&= \left(\frac{\mathcal{M}^+(i_{a_q}, i_{a_l})}{2\mathcal{R}_{\boldsymbol{M}}} + \frac{\mathcal{N}^+(j_{a_q}, j_{a_l})}{2\mathcal{R}_{\boldsymbol{N}}}\right) \alpha' \boldsymbol{v}(a_1)\boldsymbol{v}(a_l)^\top
\end{aligned}$$

Similarly we have:

$$\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_l) = \left(\frac{\mathcal{M}^+(i_{a_q}, i_{a_l})}{2\mathcal{R}_{\boldsymbol{M}}} + \frac{\mathcal{N}^+(j_{a_q}, j_{a_l})}{2\mathcal{R}_{\boldsymbol{N}}}\right) \alpha' \bar{\boldsymbol{v}}(a_1)\bar{\boldsymbol{v}}(a_l)^\top,$$

from which the result follows. □

Lemma 43. *For all $l \in \mathbb{N}$ and for all $a_1, a_2, ..., a_l \in \mathbb{U} \cap [t-1]$ we have:*

$$\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_l)\right) = \operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_l)\right)$$

Proof. By Lemma 42, let α be such that

$$\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_l) = \alpha \boldsymbol{v}(a_1)\boldsymbol{v}(a_l)^\top$$

and

$$\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_l) = \alpha \bar{\boldsymbol{v}}(a_1)\bar{\boldsymbol{v}}(a_l)^\top.$$

Note that:

$$\begin{aligned}
&\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_l)\right)\\
&=\alpha\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\boldsymbol{v}(a_1)\boldsymbol{v}(a_l)^\top\right)\\
&=\alpha\operatorname{tr}\left(\boldsymbol{v}(t)\boldsymbol{v}(t)^\top\boldsymbol{v}(a_1)\boldsymbol{v}(a_l)^\top\right)\\
&=\alpha\operatorname{tr}\left(\boldsymbol{v}(a_l)^\top\boldsymbol{v}(t)\boldsymbol{v}(t)^\top\boldsymbol{v}(a_1)\right)\\
&=\alpha\left(\boldsymbol{v}(a_l)^\top\boldsymbol{v}(t)\right)\left(\boldsymbol{v}(t)^\top\boldsymbol{v}(a_1)\right)\\
&=\alpha\left(\frac{\mathcal{M}^+(i_{a_l},i_t)}{2\mathcal{R}_{\boldsymbol{M}}}+\frac{\mathcal{N}^+(j_{a_l},j_t)}{2\mathcal{R}_{\boldsymbol{N}}}\right)\left(\frac{\mathcal{M}^+(i_t,i_{a_1})}{2\mathcal{R}_{\boldsymbol{M}}}+\frac{\mathcal{N}^+(j_t,j_{a_1})}{2\mathcal{R}_{\boldsymbol{N}}}\right)
\end{aligned}$$

Similarly we have:

$$\begin{aligned}
&\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_l)\right)\\
=&\alpha\left(\frac{\mathcal{M}^+(i_{a_l},i_t)}{2\mathcal{R}_{\boldsymbol{M}}}+\frac{\mathcal{N}^+(j_{a_l},j_t)}{2\mathcal{R}_{\boldsymbol{N}}}\right)\left(\frac{\mathcal{M}^+(i_t,i_{a_1})}{2\mathcal{R}_{\boldsymbol{M}}}+\frac{\mathcal{N}^+(j_t,j_{a_1})}{2\mathcal{R}_{\boldsymbol{N}}}\right)
\end{aligned}$$

The result follows. □

Lemma 44. *For any $q\in\mathbb{N}$, any $\kappa\in\mathfrak{R}^+$ and any $b_1,b_2,\cdots b_{t-1}\in\mathfrak{R}$ we have:*

$$\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\left(\sum_{s\in\mathbb{U}\cap[t-1]}b_s\tilde{\boldsymbol{X}}^t(s)\right)^q\right)=\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\left(\sum_{s\in\mathbb{U}\cap[t-1]}b_s\tilde{\boldsymbol{X}}^{T+1}(s)\right)^q\right)$$

Proof. We have:

$$\begin{aligned}
&\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\left(\sum_{s\in\mathbb{U}\cap[t-1]}b_s\tilde{\boldsymbol{X}}^t(s)\right)^q\right)\\
&=\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\sum_{a_1\in\mathbb{U}\cap[t-1]}\sum_{a_2\in\mathbb{U}\cap[t-1]}\cdots\sum_{a_q\in\mathbb{U}\cap[t-1]}\left(\prod_{i=1}^q b_{a_i}\right)\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_q)\right)\\
&=\sum_{a_1\in\mathbb{U}\cap[t-1]}\sum_{a_2\in\mathbb{U}\cap[t-1]}\cdots\sum_{a_q\in\mathbb{U}\cap[t-1]}\left(\prod_{i=1}^q b_{a_i}\right)\operatorname{tr}\left(\tilde{\boldsymbol{X}}^t(t)\tilde{\boldsymbol{X}}^t(a_1)\tilde{\boldsymbol{X}}^t(a_2)\cdots\tilde{\boldsymbol{X}}^t(a_q)\right)
\end{aligned}$$

and similarly,

$$\begin{aligned}
&\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\left(\sum_{s=1}^{t-1} b_s \tilde{\boldsymbol{X}}^{T+1}(s)\right)^q\right) \\
&= \operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t) \sum_{a_1 \in \mathbb{U}\cap[t-1]} \sum_{a_2 \in \mathbb{U}\cap[t-1]} \cdots \sum_{a_q \in \mathbb{U}\cap[t-1]} \left(\prod_{i=1}^{q} b_{a_i}\right) \tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_q)\right) \\
&= \sum_{a_1 \in \mathbb{U}\cap[t-1]} \sum_{a_2 \in \mathbb{U}\cap[t-1]} \cdots \sum_{a_q \in \mathbb{U}\cap[t-1]} \left(\prod_{i=1}^{q} b_{a_i}\right) \operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\tilde{\boldsymbol{X}}^{T+1}(a_1)\tilde{\boldsymbol{X}}^{T+1}(a_2)\cdots\tilde{\boldsymbol{X}}^{T+1}(a_q)\right).
\end{aligned}$$

The result follows by Lemma 43. □

Lemma 45. *For any $\kappa \in \mathfrak{R}^+$ and any $b_1, b_2, \cdots b_{t-1} \in \mathfrak{R}$ we have:*

$$\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\exp\left(\kappa\boldsymbol{I} + \sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{t}(s)\right)\right) = \operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\exp\left(\kappa\boldsymbol{I} + \sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{T+1}(s)\right)\right)$$

Proof. Using the fact that $\exp(\boldsymbol{A}+\boldsymbol{B}) = \exp(\boldsymbol{A})\exp(\boldsymbol{B})$ for commuting matrices $\boldsymbol{A}$ and $\boldsymbol{B}$, and noting that the multiple of the identity matrix commutes with any matrix, we have that

$$\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\exp\left(\kappa\boldsymbol{I} + \sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{t}(s)\right)\right) = \operatorname{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\exp(\kappa\boldsymbol{I})\exp\left(\sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{t}(s)\right)\right).$$

By the Taylors series expansion we have:

$$\begin{aligned}
\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\exp(\kappa\boldsymbol{I})\exp\left(\sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{t}(s)\right)\right) &= e^{\kappa}\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\sum_{q=0}^{\infty}\frac{1}{q!}\left(\sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{t}(s)\right)^q\right) \\
&= e^{\kappa}\sum_{q=0}^{\infty}\frac{1}{q!}\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\left(\sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{t}(s)\right)^q\right)
\end{aligned}$$

Similarly, we have

$$\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\exp\left(\kappa\boldsymbol{I} + \sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{T+1}(s)\right)\right) = e^{\kappa}\sum_{q=0}^{\infty}\frac{1}{q!}\operatorname{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\left(\sum_{s\in\mathbb{U}\cap[t-1]} b_s \tilde{\boldsymbol{X}}^{T+1}(s)\right)^q\right).$$

The result then follows from Lemma 44. □

Equivalence of Algorithms

On a trial t, let $\bar{z}_t$ be the prediction ($\bar{y}_t$) of the inductive algorithm and let $\bar{y}_t$ remain the prediction of the transductive algorithm. We fix $\kappa := \log\left(\widehat{\mathcal{D}}/(m+n)\right)$.

Lemma 46. *On a trial t the prediction, $\bar{y}_t$, of the transductive algorithm is given by:*

$$\bar{y}_t = \mathrm{tr}\left(\tilde{\boldsymbol{X}}^{T+1}(t)\exp\left(\kappa\boldsymbol{I} + \sum_{s=1}^{t-1} f_s(\bar{y}_s)\tilde{\boldsymbol{X}}^{T+1}(s)\right)\right)$$

and the prediction, $\bar{z}_t$, of the inductive algorithm is given by:

$$\bar{z}_t = \mathrm{tr}\left(\tilde{\boldsymbol{X}}^{t}(t)\exp\left(\kappa\boldsymbol{I} + \sum_{s=1}^{t-1} f_s(\bar{z}_s)\tilde{\boldsymbol{X}}^{t}(s)\right)\right)$$

where $f_s(x) := \eta y_s$ if $y_s x \leq \gamma \times$ [NON-CONSERVATIVE] and $f_s(x) := 0$ otherwise.

Proof. Direct from algorithms, noting that if $s \notin \mathbb{U} \cap [t-1]$ then $f_s(\bar{z}_s) = 0$. □

Lemma 47. *Given a trial t, if $\bar{y}_s = \bar{z}_s$ for all $s < t$, then $\bar{y}_t = \bar{z}_t$.*

Proof. Direct from Lemmas 46 and 45 (with $b_s := f_s(\bar{y}_t) = f_s(\bar{z}_t)$), noting that if $s \notin \mathbb{U} \cap [t-1]$ then $f_s(\bar{z}_s) = 0$. □

Proposition 6 follows by induction over Lemma 47. ■

3.6.4 Proof of Proposition 7

Before proving Proposition 7, we first present an intermediate lemma, Lemma 48.

Lemma 48. *Giving a matrix $\boldsymbol{A} \in \Re^{n\times k}$, scalar $a \in \Re$, we have that*

$$\Delta(\boldsymbol{A}\boldsymbol{A}^\top, a\boldsymbol{I}^n) = \Delta(\boldsymbol{A}^\top\boldsymbol{A}, a\boldsymbol{I}^k) + a(n-k).$$

Proof. Denote the $n \times n$ eigendecomposition diagonal matrix of $\boldsymbol{A}^\top\boldsymbol{A}$ as $\boldsymbol{\Lambda}_n$ and that of $\boldsymbol{A}\boldsymbol{A}^\top$ as $\boldsymbol{\Lambda}_k$. Observe that the non-zero eigenvalues of $\boldsymbol{A}^\top\boldsymbol{A}$ is identical to that of $\boldsymbol{A}\boldsymbol{A}^\top$. In both cases, these are given by the square of the singular values of $\boldsymbol{A}$.

This fact then allows us to write the following,

$$\begin{aligned}
\Delta(\boldsymbol{A}\boldsymbol{A}^\top, a\boldsymbol{I}^n) &= \operatorname{tr}(\boldsymbol{A}\boldsymbol{A}^\top \log(\boldsymbol{A}\boldsymbol{A}^\top)) - \operatorname{tr}(\boldsymbol{A}\boldsymbol{A}^\top \log(a\boldsymbol{I}^n)) + \operatorname{tr}(a\boldsymbol{I}^n) - \operatorname{tr}(\boldsymbol{A}\boldsymbol{A}^\top) \\
&= \operatorname{tr}(\boldsymbol{\Lambda}_n \log(\boldsymbol{\Lambda}_n)) - \operatorname{tr}(\boldsymbol{\Lambda}_n \log(a\boldsymbol{I}^n)) + \operatorname{tr}(a\boldsymbol{I}^n) + \operatorname{tr}(\boldsymbol{\Lambda}_n) \\
&= \operatorname{tr}(\boldsymbol{\Lambda}_k \log(\boldsymbol{\Lambda}_k)) - \operatorname{tr}(\boldsymbol{\Lambda}_k \log(a\boldsymbol{I}^k)) + \operatorname{tr}(a\boldsymbol{I}^n) + \operatorname{tr}(\boldsymbol{\Lambda}_k) \\
&= \operatorname{tr}(\boldsymbol{\Lambda}_k \log(\boldsymbol{\Lambda}_k)) - \operatorname{tr}(\boldsymbol{\Lambda}_k \log(a\boldsymbol{I}^k)) + \operatorname{tr}(a\boldsymbol{I}^k) + a(n-k) + \operatorname{tr}(\boldsymbol{\Lambda}_k) \\
&= \operatorname{tr}(\boldsymbol{A}^\top\boldsymbol{A} \log(\boldsymbol{A}^\top\boldsymbol{A})) - \operatorname{tr}(\boldsymbol{A}^\top\boldsymbol{A} \log(a\boldsymbol{I}^k)) + \operatorname{tr}(a\boldsymbol{I}^k) + a(n-k) + \operatorname{tr}(\boldsymbol{\Lambda}_k) \\
&= \Delta(\boldsymbol{A}^\top\boldsymbol{A}, a\boldsymbol{I}^k) + a(n-k).
\end{aligned}$$

□

We now recall the proposition and provide a proof.

Proposition 7. *For all t*

$$\boldsymbol{W}_1^{*,t} = \boldsymbol{W}_2^{*,t}$$

where $\boldsymbol{W}_1^{*,t}$ *is defined as in* (3.15) *and* $\boldsymbol{W}_2^{*,t}$ *is defined as in* (3.16).

Proof. We recall that

$$\boldsymbol{W}_2^{*,t} = \operatorname*{argmin}_{\boldsymbol{W}:\boldsymbol{W}\in\boldsymbol{S}_+^{m+n}} \Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}^{m+n}) + \eta\gamma \sum_{s\in[t]} f_\gamma^s(\boldsymbol{W}).$$

Then, we write

$$\begin{aligned}
\boldsymbol{W}_2^{*,t} &= \operatorname*{argmin}_{\boldsymbol{W}:\boldsymbol{W}\in\boldsymbol{S}_+^{m+n},\tilde{\boldsymbol{W}}=\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}} \Delta(\tilde{\boldsymbol{W}}, \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}^{m+n}) + \eta\gamma \sum_{s=1}^{t-1} f_\gamma^s(\boldsymbol{W}) \\
&= \operatorname*{argmin}_{\boldsymbol{W}:\boldsymbol{W}=\boldsymbol{Z}\boldsymbol{Z}^\top,\bar{\boldsymbol{Z}}=\boldsymbol{D}\boldsymbol{Z}} \Delta(\bar{\boldsymbol{Z}}\bar{\boldsymbol{Z}}^\top, \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}^{m+n}) + \eta\gamma \sum_{s=1}^{t-1} f_\gamma^s(\boldsymbol{W})
\end{aligned}$$

where $\boldsymbol{Z} \in \Re^{(m+n)\times k}$, the last optimisation is over all $\boldsymbol{Z}$ and k, and we have made use of the fact that requiring $\boldsymbol{W} \in \boldsymbol{S}_+^{m+n}$ is equivalent to requiring that $\boldsymbol{W} = \boldsymbol{Z}\boldsymbol{Z}^\top$. This is because any symmetric matrix $\boldsymbol{A}$ has an eigendecomposition $\boldsymbol{A} = \boldsymbol{V}\boldsymbol{\Sigma}\boldsymbol{V}^\top$. Since all the eigenvalues of $\boldsymbol{W}$ are non-negative by virtue of it being positive semi-definite, we can set $\boldsymbol{Z} = \boldsymbol{V}\sqrt{\boldsymbol{\Sigma}}$. Similarly, any matrix $\boldsymbol{Z}$ has a SVD decomposition

$\boldsymbol{Z} = \boldsymbol{S}\boldsymbol{\Omega}\boldsymbol{T}^\top$, so that $\boldsymbol{Z}\boldsymbol{Z}^\top = \boldsymbol{S}\boldsymbol{\Omega}^2\boldsymbol{S}^\top$, which gives a matrix with non-negative eigenvalues $\{\Omega_{i,i}^2 : i \in [m+n]\}$. Then using Lemma 48,

$$\begin{aligned}\boldsymbol{W}_2^{*,t} &= \operatorname*{argmin}_{\boldsymbol{W}:\boldsymbol{W}=\boldsymbol{Z}\boldsymbol{Z}^\top,\bar{\boldsymbol{Z}}=\boldsymbol{D}\boldsymbol{Z}} \Delta(\bar{\boldsymbol{Z}}^\top\bar{\boldsymbol{Z}}, \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}^k) + \eta\gamma \sum_{s=1}^{t-1} f_\gamma^t(\boldsymbol{W}) + \frac{\widehat{\mathcal{D}}(m+n-k)}{m+n} \\ &= \operatorname*{argmin}_{\boldsymbol{W}:\boldsymbol{W}=\boldsymbol{Z}\boldsymbol{Z}^\top} \Delta(\boldsymbol{Z}^\top\boldsymbol{D}^2\boldsymbol{Z}, \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{I}^k) + \eta\gamma \sum_{s=1}^{t-1} f_\gamma^t(\boldsymbol{W}) + \frac{\widehat{\mathcal{D}}(m+n-k)}{m+n}\end{aligned}$$

Finally by considering wlog $\boldsymbol{Z} = [\boldsymbol{P}; \boldsymbol{Q}]$ for $\boldsymbol{P} \in \Re^{n\times k}$, $\boldsymbol{Q} \in \Re^{m\times k}$, we then have that this equals $\boldsymbol{W}_1^{*,t}$.

□

3.6.5 Proof of Proposition 8

In the following, we prove Proposition 8. We compute $\boldsymbol{W}^{*,t+1}$ as described in (3.17), and show that this gives rise to the prediction in Algorithm 1. We note that (3.17) is convex in $\boldsymbol{W}$ due to the following lemma.

Lemma 49. *The function*

$$\Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \boldsymbol{A}) + f_\gamma^t(\boldsymbol{W})$$

is convex in $\boldsymbol{W} \in \boldsymbol{S}_+^{m+n}$, where $\boldsymbol{A} \in \boldsymbol{S}_+^{m+n}$ and $\boldsymbol{D} \in \boldsymbol{S}_+^{m+n}$.

Proof. First, we note that the sum of convex functions gives a convex function. The hinge loss $f_\gamma^t(\boldsymbol{W})$ is a convex function in $\bar{y}_t$, where we recall that $\bar{y}_t := \operatorname{tr}(\boldsymbol{W}\boldsymbol{X}) - 1$. Since the composition of a convex and linear function is a convex function, we have that the hinge loss is convex in $\boldsymbol{W}$.

Hence, we aim to show the convexity of $F(\boldsymbol{W}) := \Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \boldsymbol{A})$. We define the function

$$\boldsymbol{G}(\boldsymbol{W}) = \boldsymbol{D}\boldsymbol{W}\boldsymbol{D}$$

and the function

$$H(\boldsymbol{W}) = \Delta(\boldsymbol{W}, \boldsymbol{A}).$$

We then have that $F(\boldsymbol{W}) = H(\boldsymbol{G}(\boldsymbol{W}))$. It is clear that $\boldsymbol{G}$ is linear in $\boldsymbol{W}$ since

$$\begin{aligned}\boldsymbol{G}(\alpha\boldsymbol{W}_1 + (1-\alpha)\boldsymbol{W}_2) &= \boldsymbol{D}(\alpha\boldsymbol{W}_1 + (1-\alpha\boldsymbol{W}_2))\boldsymbol{D} \\ &= \alpha\boldsymbol{D}\boldsymbol{W}_1\boldsymbol{D} + (1-\alpha)\boldsymbol{D}\boldsymbol{W}_2\boldsymbol{D} \\ &= \alpha\boldsymbol{G}(\boldsymbol{W}_1) + (1-\alpha)\boldsymbol{G}(\boldsymbol{W}_2).\end{aligned}$$

We also know that H is convex in $\boldsymbol{W}$ due to the convexity of the quantum relative entropy with respect to its first argument. By the convexity of H and the linearity of $\boldsymbol{G}$, we have that their composition F is again convex.

□

Since the function in (3.17) is convex, we can use the first derivative to find the minimizer. For a function $F(\boldsymbol{W}) : \mathfrak{R}^{(m+n)\times(m+n)} \to \mathfrak{R}$, we will denote its derivative with respect to $\boldsymbol{W}$ as $\nabla_{\boldsymbol{W}} F(\boldsymbol{W})$.

Lemma 50. *We have*

$$\nabla_{\boldsymbol{W}} f_\gamma^t(\boldsymbol{W}) = -\frac{1}{\gamma} y_t \boldsymbol{X}^t [\gamma > y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1)].$$

Proof. Clearly if $\gamma \leq y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1)$, then $\mathcal{L}_{\mathrm{hi}}^\gamma(\bar{y}_t, y_t) = f_\gamma^t(\boldsymbol{W}) = 0$, resulting in a derivative of $\boldsymbol{0}^{(m+n)\times(m+n)}$. For $\gamma > y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1)$, we have that $f_\gamma^t(\boldsymbol{W}) = \frac{1}{\gamma}[\gamma - y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1)]$ which gives the lemma since $\nabla_{\boldsymbol{A}} \mathrm{tr}(\boldsymbol{A}\boldsymbol{B}) = \boldsymbol{B}^\top$ and $\boldsymbol{X}^t$ is symmetric for all t. □

Lemma 51. *For $F(\boldsymbol{W}) := \Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D})$, we have*

$$\nabla_{\boldsymbol{W}} F(\boldsymbol{W}) = \boldsymbol{D}(\log(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}) - \log(\boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}))\boldsymbol{D}.$$

Proof. Recall that $(\nabla_{\boldsymbol{W}} F(\boldsymbol{W}))_{ij} = \frac{\partial F}{\partial W_{ij}}$ for all $i, j \in [m+n]$. Defining $\boldsymbol{G}(\boldsymbol{W}) := \boldsymbol{D}\boldsymbol{W}\boldsymbol{D}$, we have that

$$\frac{\partial F}{\partial W_{ij}} = \sum_{k,l} \frac{\partial F}{\partial G_{kl}} \frac{\partial G_{kl}}{W_{ij}}. \tag{3.59}$$

Since $G_{kl} = \sum_{s,t} D_{ks} W_{st} D_{tl}$,

$$\frac{\partial G_{kl}}{\partial W_{ij}} = D_{ki} D_{jl}. \tag{3.60}$$

We also have

$$\frac{\partial F}{\partial G_{kl}} = (\log(\boldsymbol{G}) - \log(\boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}))_{kl}, \tag{3.61}$$

since $\nabla_{\boldsymbol{A}}(\Delta(\boldsymbol{A}, \boldsymbol{B})) = \log(\boldsymbol{A}) - \log(\boldsymbol{B})$ for symmetric matrices $\boldsymbol{A}$, $\boldsymbol{B}$ (see e.g. [4]). Combining (3.59), (3.60) and (3.61),

$$\frac{\partial F}{\partial W_{ij}} = \sum_{k,\ell} (\log(\boldsymbol{G}) - \log(\boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}))_{kl} D_{ki} D_{jl}.$$

The lemma then follows since $\boldsymbol{D}$ is symmetric.

□

Lemma 52.

$$\nabla_{\boldsymbol{W}}(\Delta(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}, \boldsymbol{D}\boldsymbol{W}^t\boldsymbol{D}) + \eta\gamma f_\gamma^t(\boldsymbol{W}) = \boldsymbol{D}(\log(\boldsymbol{D}\boldsymbol{W}\boldsymbol{D}) - \log(\boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}))\boldsymbol{D} - \eta y_t \boldsymbol{X}^t [\gamma > y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1)]$$

Proof. This follows directly from Lemmas 50 and 51. □

Lemma 53. *We have*

$$\boldsymbol{X}^t = \boldsymbol{D}\tilde{\boldsymbol{X}}^t\boldsymbol{D}.$$

Proof. Recall that

$$\boldsymbol{X}^t := \frac{1}{2}\left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]\left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]^\top.$$

and

$$\tilde{\boldsymbol{X}}^t := \frac{1}{2}\boldsymbol{D}^{-1}\left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]\left[\boldsymbol{e}_m^{i_t}; \boldsymbol{e}_n^{j_t}\right]^\top \boldsymbol{D}^{-1}.$$

□

Lemma 54. *For all $t \in [T]$,*

$$\boldsymbol{W}^{*,t+1} = \begin{cases} \boldsymbol{D}^{-1}\exp(\log(\boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}) + \eta y_t \tilde{\boldsymbol{X}}^t)\boldsymbol{D}^{-1} & \text{if } \gamma > y_t(\mathrm{tr}(\boldsymbol{W}\boldsymbol{X}^t) - 1) \\ \boldsymbol{W}^{*,t} & \text{otherwise.} \end{cases} \tag{3.62}$$

Proof. We recall that $\boldsymbol{W}^{*,t+1} = \operatorname{argmin}_{\boldsymbol{W}} \Delta(\boldsymbol{DWD}, \boldsymbol{DW}^{*,t}\boldsymbol{D}) + \eta\gamma f_\gamma^t(\boldsymbol{W})$. We first consider the case when $\gamma > y_t(\operatorname{tr}(\boldsymbol{WX}^t) - 1)$. From Lemma 52, we have that the minimum $\boldsymbol{W}^*$ satisfies

$$\boldsymbol{D}(\log(\boldsymbol{DW}^*\boldsymbol{D}) - \log(\boldsymbol{DW}^{*,t}\boldsymbol{D}))\boldsymbol{D} - \eta y_t \boldsymbol{X}^t = \boldsymbol{0}.$$

Rearranging,

$$\log(\boldsymbol{DW}^*\boldsymbol{D}) - \log(\boldsymbol{DW}^{*,t}\boldsymbol{D}) = \eta y_t \boldsymbol{D}^{-1}\boldsymbol{X}^t\boldsymbol{D}^{-1}$$

$$\boldsymbol{W}^* = \boldsymbol{D}^{-1}\exp(\log(\boldsymbol{DW}^{*,t}\boldsymbol{D}) + \eta y_t \boldsymbol{D}^{-1}\boldsymbol{X}^t\boldsymbol{D}^{-1})\boldsymbol{D}^{-1}.$$

From Lemma 53, we then obtain the first case. For the other case, we have that the minimum $\boldsymbol{W}^*$ satisfies

$$\boldsymbol{D}(\log(\boldsymbol{DW}^*\boldsymbol{D}) - \log(\boldsymbol{DW}^{*,t}\boldsymbol{D}))\boldsymbol{D} = \boldsymbol{0},$$

from which the second case follows. □

Lemma 55. *For all $t \in [T]$,*

$$\boldsymbol{W}^{*,t} = \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^t\boldsymbol{D}^{-1},$$

where $\tilde{\boldsymbol{W}}^t$ is as defined in Algorithm 1.

Proof. We prove by induction over t. For $t = 1$, we have that $\boldsymbol{W}^{*,1} = \frac{\widehat{\mathcal{D}}}{m+n}\boldsymbol{D}^{-2} = \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^1\boldsymbol{D}^{-1}$. Now we show that if $\boldsymbol{W}^{*,t} = \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^t\boldsymbol{D}^{-1}$, we also have $\boldsymbol{W}^{*,t+1} = \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^{t+1}\boldsymbol{D}^{-1}$. From Lemma 54, we have for the case $\gamma > y_t(\operatorname{tr}(\boldsymbol{WX}^t) - 1)$:

$$\begin{aligned}
\boldsymbol{W}^{*,t+1} &= \boldsymbol{D}^{-1}\exp(\log(\boldsymbol{DW}^{*,t}\boldsymbol{D}) + \eta y_t \tilde{\boldsymbol{X}}^t)\boldsymbol{D}^{-1} \\
&= \boldsymbol{D}^{-1}\exp(\log(\tilde{\boldsymbol{W}}^t) + \eta y_t \tilde{\boldsymbol{X}}^t)\boldsymbol{D}^{-1} && (3.63) \\
&= \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^{t+1}\boldsymbol{D}^{-1}. && (3.64)
\end{aligned}$$

where (3.63) comes from the assumption that $\boldsymbol{W}^{*,t} = \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^t\boldsymbol{D}^{-1}$, and (3.64) comes from the update rule in Algorithm 1. Otherwise, there is no update and we have

$$\begin{aligned}\boldsymbol{W}^{*,t+1} &= \boldsymbol{W}^{*,t} \\ &= \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^t\boldsymbol{D}^{-1} \\ &= \boldsymbol{D}^{-1}\tilde{\boldsymbol{W}}^{t+1}\boldsymbol{D}^{-1}.\end{aligned}$$

□

By Lemmas 55 and 53,

$$\begin{aligned}\bar{y}_t + 1 &= \mathrm{tr}(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t) \\ &= \mathrm{tr}(\boldsymbol{D}\boldsymbol{W}^{*,t}\boldsymbol{D}\tilde{\boldsymbol{X}}^t) \\ &= \mathrm{tr}(\boldsymbol{W}^{*,t}\boldsymbol{D}\tilde{\boldsymbol{X}}^t\boldsymbol{D}) \\ &= \mathrm{tr}(\boldsymbol{W}^{*,t}\boldsymbol{X}^t).\end{aligned}$$

■

Chapter 4

Latent Block Structure

In the previous chapter, we gave mistake and regret bounds for online matrix completion with side information for a general matrix. In this chapter, we will apply these bounds to binary matrices with a $k \times \ell$ latent block structure, where k and ℓ are respectively the number of distinct row and column factors. In the transductive setting, if the side information is predictive of the underlying factorization of the matrix, then in an ideal case, the mistakes are bounded by $\tilde{O}(k\ell)$. In the inductive setting, we provide an example where the mistakes are bounded by $\tilde{O}(\max(k,\ell)^2 \min(k,\ell))$.

4.1 Introduction

In this chapter, we aim to apply the bounds that we developed in the last chapter for the MEG and MGD algorithms to the hypothesis class of matrices with a $k \times \ell$ latent block structure. Such matrices have the property that both the margin complexity and the max-norm are bounded by $\min(\sqrt{k}, \sqrt{\ell})$. Recall that the mistake bounds that we proved in the previous chapter are of the form $\tilde{O}(\mathcal{D}/\gamma^2)$ and $O(\mathcal{D}^2/\gamma^2)$, where $\mathcal{D}$ is the quasi-dimension term and $1/\gamma^2$ is a parameter of the algorithm which serves as an upper estimate of the squared margin complexity $\mathrm{mc}(\boldsymbol{U})^2$ of the comparator matrix $\boldsymbol{U}$. In this chapter, we will upper bound the quasi-dimension term $\mathcal{D}$ for different scenarios in the transductive and inductive settings.

Recall that in the case of vacuous side information, $\mathcal{D} = m + n$. However, if

there is a $k \times \ell$ latent block structure and the side information is predictive, then we show that $\mathcal{D} \in O(k + \ell)$ in the transductive setting. This results in an MEG mistake bound that is $\tilde{O}(k\ell)$, which we will later argue is optimal. In the inductive setting, where we are given only a pair of kernel functions, we give an example with the min kernel where the quasi-dimension $\mathcal{D}$ is now bounded by $O(k^2 + \ell^2)$, giving an MEG mistake bound that is $\tilde{O}(\max(k, \ell)^2 \min(k, \ell))$. Although latent block structure may appear to be a "fragile" measure of matrix complexity, our regret bound implies that performance will scale smoothly in the case of adversarial noise.

The chapter is organized as follows. First, we give a background on matrices with a latent block structure. In Section 4.3, we formally introduce the concept of latent block structure (Definition 56) and provide an upper bound (Theorem 57) for the quasi-dimension $\mathcal{D}$ when the matrix has latent block structure. For the transductive setting, we provide an example that bounds $\mathcal{D}$ when we have graph-based side information (Section 4.4.1); and a further example (Section 4.4.2) when the matrix has additionally a "community" structure. For the inductive setting, we present an example illustrating a bound on $\mathcal{D}$ when the side information comes as vectors in $\mathfrak{R}^d$ which are separated by a clustering via hyper-rectangles in Section 4.5.1 and hyper-spheres in Section 4.5.2. Finally, we perform synthetic experiments in Section 4.6.

4.2 Background

Hartigan [72] introduced the idea of permuting a matrix by both the rows and columns into a few homogeneous blocks. This is equivalent to assuming that each row (column) has an associated row (column) class, and that the matrix entry is completely determined by its corresponding row and column classes. This has since become known as co- or bi-clustering. This same assumption has become the basis for probabilistic models which can then be used to "complete" a matrix with missing entries. The authors of [73] give rate-optimal results for this problem in the batch setting and provide a literature overview. It is natural to compare this assumption to the dominant alternative, which assumes that there exists a low rank decomposition

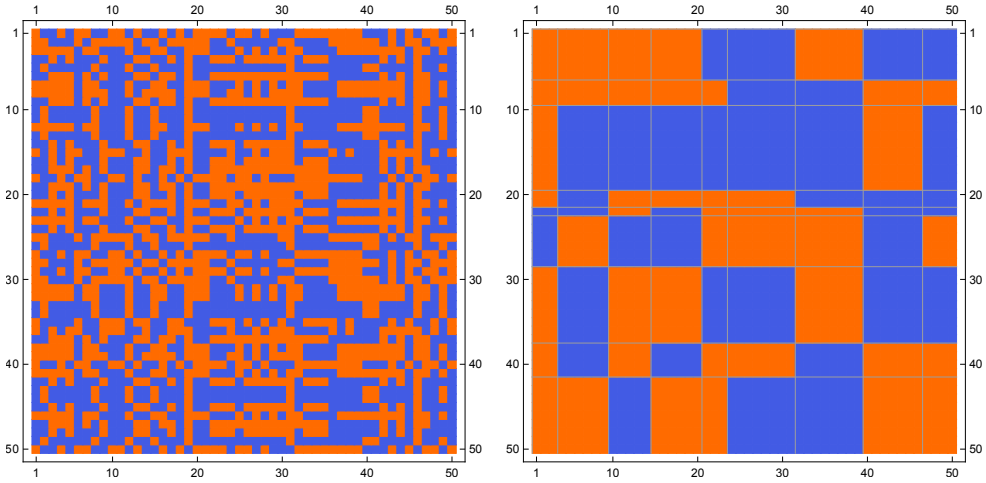


Figure 4.1: A $(9, 9)$-biclustered 50×50 binary matrix before/after permuting into latent blocks.

of the matrix to be completed, see for instance [74]. Common to both approaches is that associated with each row and column, there is an underlying *latent* factor so that the given matrix entry is determined by a function on the appropriate row and column factor. The low-rank assumption is that the latent factors are vectors in $\mathfrak{R}^d$ and that the function is the dot product. The latent block structure assumption is that the latent factors are instead categorical and that the function between factors is arbitrary.

4.3 Definition and Quasi-Dimension Bound

We introduce the concept class of (k, ℓ)-binary-biclustered matrices (previously defined in [34, Section 5]), in the following definition. A visualization of an example (9,9)-biclustered matrix can be found in Figure 4.1. We then give an upper bound to the quasi-dimension term $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ when a matrix has this type of latent structure in Theorem 57. The magnitude of the bound will depend on how "predictive" matrices $\boldsymbol{M}$ and $\boldsymbol{N}$ are of the latent block structure. In Sections 4.4.1 and 4.4.2, we will use a variant of the discrete Laplacian matrix for $\boldsymbol{M}$ and $\boldsymbol{N}$ to encode side information and illustrate the resultant bounds for idealized scenarios.

Definition 56. *The class of* (k, ℓ)*-binary-biclustered matrices is defined as*

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

Thus each row r_i is associated with a latent factor in $[k]$ and each column c_j is associated with a latent factor in $[\ell]$ and the interaction of factors is determined by a matrix $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$. More visually, a binary matrix is (k,ℓ)-biclustered if there exists some permutation of the rows and columns into a $k \times \ell$ grid of blocks each uniformly labeled -1 or $+1$. Determining if a matrix is in $\mathbb{B}^{m,n}_{k,\ell}$, may be done directly by a greedy algorithm. However, the problem of determining if a matrix with missing entries may be completed to a matrix in $\mathbb{B}^{m,n}_{k,\ell}$ was shown in [75, Lemma 8] to be NP-complete by reducing the problem to Clique Cover.

Many natural functions of matrix complexity are invariant to the presence of block structure. A function $f : \mathcal{X} \to \mathfrak{R}$ with respect to a class of matrices $\mathcal{X}$ is *block-invariant* if for all $m, k, n, \ell \in \mathbb{N}^+$ with $m \geq k$, $n \geq \ell$, $\boldsymbol{R} \in \mathcal{B}^{m,k}$ and $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ we have that $f(\boldsymbol{X}) = f(\boldsymbol{R}\boldsymbol{X}\boldsymbol{C}^\top)$ for any $k \times \ell$ matrix $\boldsymbol{X} \in \mathcal{X}$. The max-norm, margin complexity, rank and VC-dimension[1] are all block-invariant. From the block-invariance of the max-norm, we may conclude that for $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$,

$$\operatorname{mc}(\boldsymbol{U}) \leq \|\boldsymbol{U}\|_{\max} = \|\boldsymbol{U}^*\|_{\max} \leq \min(\sqrt{k}, \sqrt{\ell}). \tag{4.1}$$

This follows since we may decompose $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ for some $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$, $\boldsymbol{R} \in \mathcal{B}^{m,k}$ and $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ and then use the observation in the preliminaries that the max-norm of any matrix in $\{-1,1\}^{m\times n}$ is bounded by $\min(\sqrt{m}, \sqrt{n})$.

In the following theorem, we give a bound for the quasi-dimension $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ which will scale with the dimensions of the latent block structure and the "predictivity" of $\boldsymbol{M}$ and $\boldsymbol{N}$ with respect to that block structure. The bound is independent of γ in so far as $\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ is finite.

[1] Here, a hypothesis class $\mathcal{H}$ defines a matrix via $\boldsymbol{U} := (h(x))_{h\in\mathcal{H},x\in\mathcal{X}}$.

Theorem 57. *If* $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$ *define*

$$\mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \begin{cases} 2\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + 2\operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} + 2k + 2\ell & \boldsymbol{M},\ \boldsymbol{N} \text{ are PDLaplacians} \\ k\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + \ell\operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} & \boldsymbol{M} \in \boldsymbol{S}^m_{++} \text{ and } \boldsymbol{N} \in \boldsymbol{S}^n_{++} \end{cases}, \tag{4.2}$$

as the minimum over all decompositions of $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ *for* $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ *and* $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$. *Thus for* $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$,

$$\begin{aligned} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) &\leq \mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \qquad (\textit{if } \|\boldsymbol{U}\|_{max} \leq 1/\gamma) \\ \min_{\boldsymbol{V}\in \mathrm{SP}^1(\boldsymbol{U})} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}) &\leq \mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \qquad (\textit{if } \operatorname{mc}(\boldsymbol{U}) \leq 1/\gamma)\,. \end{aligned}$$

The bound $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \leq \mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ allows us to bound the quality of the side information in terms of a hypothetical learning problem. Recall that $\operatorname{argmin}_{r_i y_i \geq 1 : i\in[m]}(\boldsymbol{r}^\top \boldsymbol{M}\boldsymbol{r})\mathcal{R}_{\boldsymbol{M}}$ is the upper bound on the mistakes per Novikoff's theorem [76] for predicting the elements of vector $\boldsymbol{y} \in \{-1,1\}^m$ with a kernel perceptron using $\boldsymbol{M}^{-1}$ as the kernel. Hence the term $O(\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}})$ in (4.2) may be interpreted as a bound for a one-versus-all k-class kernel perceptron where $\boldsymbol{R}$ encodes a labeling from $[k]^m$ as one-hot vectors.

It is also possible to prove a similar bound for $\mathcal{A}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \leq \mathcal{A}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$, where

$$\mathcal{A}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \begin{cases} 4(\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + k)(\operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} + \ell) & \boldsymbol{M},\ \boldsymbol{N} \text{ are PDLaplacians} \\ k\ell \cdot \operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} \cdot \operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} & \boldsymbol{M} \in \boldsymbol{S}^m_{++} \text{ and } \boldsymbol{N} \in \boldsymbol{S}^n_{++} \end{cases}. \tag{4.3}$$

This bound may be used when applying the mistake and regret bounds of the MGD algorithm. Recalling that $\mathcal{A}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \leq \frac{(\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}))^2}{4}$ from Chapter 3, we can observe that the upper bounds satisfy a similar relationship, i.e., $\mathcal{A}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \leq \frac{(\mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}))^2}{4}$.

In the remaining chapters, we will apply Theorem 57 on various examples in the transductive and inductive settings. The inductive setting is more general, as it only requires a pair of kernel functions $\mathcal{M}^+$ and $\mathcal{N}^+$, from which we can build $\boldsymbol{M}$ and $\boldsymbol{N}$. However, only in a technical sense will it be possible to model inductive side information via a PDLaplacian, since $\boldsymbol{M}^+$ can only be computed given knowl-

edge of the graph in advance. This is unfortunate since the quasi-dimension upper bound for PDLaplacians in Theorem 57 does not have the multiplicative factors in k and ℓ as compared with general positive definite matrices.

We next show an example where $\mathcal{D}^{\circ}_{M,N}(\boldsymbol{U}) \in O(k+\ell)$ with "ideal" side information.

4.4 Transductive Setting

4.4.1 Graph-based Side Information

We may use a pair of separate graph Laplacians to represent the side information on the "rows" and the "columns." A given row (column) corresponds to a vertex in the "row graph" ("column graph"). The weight of edge (i, j) represents our prior belief that row (column) i and row (column) j share the same underlying factor. Such graphs may be inherent to the data. For example, we have a social network of users and a network based on shared actors or genres for the movies in a "Netflix" type scenario. Alternatively, as is common in graph-based semi-supervised learning [77, 78] we may build a graph based on vectorial data associated with the rows (columns), for example, user demographics. Although the value of $\mathcal{D}$ will vary smoothly with the predictivity of $\boldsymbol{M}$ and $\boldsymbol{N}$ of the factor structure, in the following we give an example to quantify $\mathcal{D}^{\circ}$ in a best case scenario.

Bounding $\mathcal{D}^{\circ}$ for "ideal" graph-based side information. In this ideal case we are assuming that we know the partition of $[m]$ that maps rows to factors. The rows that share factors have an edge between them and there are no other edges. Given k factors, we then have a graph that consists of k disjoint cliques. However, to meet the technical requirement that the side information matrix $\boldsymbol{M}(\boldsymbol{N})$ is positive definite, we need to connect the cliques in a minimal fashion. We achieve this by connecting the cliques like a "star" graph. Specifically, a clique is arbitrarily chosen as the center and a vertex in that clique is arbitrarily chosen as the central vertex. From each of the other cliques, a vertex is chosen arbitrarily and connected to the central vertex. Fig. 4.2 illustrates a graph with 4 cliques that is constructed using this methodology, where the edges that connect the cliques to the central vertex are

highlighted in red. Observe that a property of this construction is that there is a path of length ≤ 4 between any pair of vertices. Now we can use the bound from Theorem 57,

$$\mathcal{D}^\circ = 2\,\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + 2\,\mathrm{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} + 2k + 2\ell\,,$$

to bound $\mathcal{D} \leq \mathcal{D}^\circ$ in this idealized case. We focus on the rows, as a parallel argument may be made for the side information on the columns. Consider the term $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}}$, where $\boldsymbol{M} := \boldsymbol{L}^\circ$ is the PDLaplacian formed from a graph with Laplacian $\boldsymbol{L}$. Then using the observation from the preliminaries that $(\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u})\mathcal{R}_{\boldsymbol{L}^\circ} \leq 2(\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u}\,\mathcal{R}_{\boldsymbol{L}} + 1)$, we have that $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} \leq 2\,\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R})\mathcal{R}_{\boldsymbol{L}} + 2k$. To evaluate this, we use the well-known equality of $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R}) = \sum_{(i,j)\in E} \left\|\boldsymbol{R}_i - \boldsymbol{R}_j\right\|^2$. Observing that each of the m rows of $\boldsymbol{R}$ is a "one-hot" encoding of the corresponding factor, only the edges between classes then contribute to the sum of the norms, and thus by construction $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R}) \leq k - 1$. We bound $\mathcal{R}_{\boldsymbol{L}} \leq 4$, using the fact that the graph diameter is a bound on $\mathcal{R}_{\boldsymbol{L}}$ (see [79, Theorem 4.2]). Combining terms and assuming similar idealized side information on the columns, we obtain $\mathcal{D}^\circ \in O(k+\ell)$. Observe then that since the comparator matrix is (k,ℓ)-biclustered, we have in the realizable case (with exact tuning), that $\mathrm{mc}(\boldsymbol{U})^2 \leq \min(k,\ell)$ by (4.1). Thus, the mistakes of the MEG and MGD algorithms are bounded by $\tilde{O}(\mathrm{mc}(\boldsymbol{U})^2\mathcal{D}^\circ) = \tilde{O}(k\ell)$ and $O(\mathrm{mc}(\boldsymbol{U})^2(\mathcal{D}^\circ)^2) = O(\min(k,\ell)\max(k,\ell)^2)$ respectively. This MEG upper bound is tight up to logarithmic factors as we may decompose $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ for some $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$, $\boldsymbol{R} \in \mathcal{B}^{m,k}$ and $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ and force a mistake for each of the $k\ell$ entries in $\boldsymbol{U}^*$.

Can side information provably help? Unsurprisingly, yes. Consider the set of matrices such that each row is either all '+1' or all '-1'. This set is exactly $\mathbb{B}^{m,n}_{2,1}$. Clearly, an adversary can force m mistakes, whereas with "ideal" side information the upper bound is $\tilde{O}(1)$.

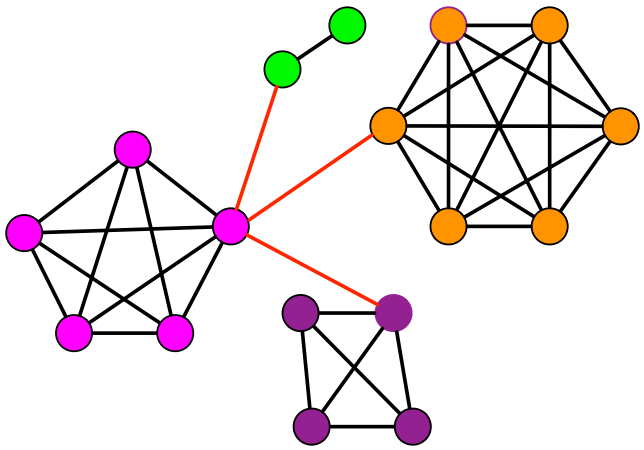


Figure 4.2: Example graph for learning graph-based side information with 4 latent factors.

4.4.2 Online Community Membership Prediction

A special case of matrix completion is the case where there are m objects which are assumed to lie in k classes (communities). In this case, the underlying matrix $\boldsymbol{U} \in \{-1, 1\}$ is given by $U_{ij} = 1$ if i and j are in the same class and $U_{ij} = -1$ otherwise. Thus this may be viewed as an online version of community detection or "similarity" prediction. In [38], this problem was addressed when the side information was encoded in a graph and the aim was to perform well when there were few edges between classes (communities).

Observe that this is an example of a (k, k)-biclustered $m \times m$ matrix where $\boldsymbol{U}^* = 2\boldsymbol{I}^k - \boldsymbol{1}\boldsymbol{1}^\top$ and there exists $\boldsymbol{R} \in \mathcal{B}^{m,k}$ such that $\boldsymbol{U} := \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{R}^\top$. Since the max-norm is block-invariant, we have that $\|\boldsymbol{U}\|_{\max} = \|\boldsymbol{U}^*\|_{\max}$. In the case of a general $k \times k$ biclustered matrix, $\|\boldsymbol{U}^*\|_{\max} \leq \sqrt{k}$ (see (4.1)). However in the case of "similarity prediction", we have $\|\boldsymbol{U}^*\|_{\max} \in O(1)$. This follows since we have a decomposition $\boldsymbol{U}^* = \boldsymbol{P}\boldsymbol{Q}^\top$ by $\boldsymbol{P}, \boldsymbol{Q} \in \Re^{k,k+1}$ with $\boldsymbol{P} := (P_{ij} = \sqrt{2}[i = j] + [j = k + 1])_{i\in[k], j\in[k+1]}$ and $\boldsymbol{Q} := (Q_{ij} = \sqrt{2}[i = j] - [j = k + 1])_{i\in[k], j\in[k+1]}$, thus giving $\|\boldsymbol{U}^*\|_{\max} \leq 3$. This example also shows that there may be an arbitrary gap between rank and max-norm of ± 1 matrices as the rank of $\boldsymbol{U}^*$ is k (in [9] this gap between the max-norm and rank was previously observed). Therefore, if the side-information matrices are taken to be the same PDLaplacian $\boldsymbol{M} = \boldsymbol{N}$ defined from a Laplacian $\boldsymbol{L}$, we have that since $\|\boldsymbol{U}\|_{\max} \in O(1)$ and $\mathcal{D}^\circ \in O(\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R})\mathcal{R}_{\boldsymbol{L}})$, mistake bounds of $\tilde{O}(\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R})\mathcal{R}_{\boldsymbol{L}})$ and $O(\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R})^2(\mathcal{R}_{\boldsymbol{L}})^2)$ are obtained for the MEG and MGD algorithms respectively,

which recover the bounds of [38, Proposition 2] up to constant factors. This work extends the results in [38] for similarity prediction to regret bounds, and to the inductive setting with general positive definite matrices. In the next section, we will see how this type of result may be extended to an inductive setting.

4.5 Inductive Setting

In the previous section, with the idealized graph-based side information, one may be dissatisfied as the skeleton of the latent structure is essentially encoded into $\boldsymbol{M}(\boldsymbol{N})$. In the inductive setting, the side information is instead revealed in an online fashion. The definition of (k, ℓ)-binary-biclustered matrices for the inductive setting may be extended to the following:

$$\mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}} = \{\boldsymbol{U} \in \{-1,1\}^{\mathcal{I}\times\mathcal{J}} : \boldsymbol{r} \in [k]^{\mathcal{I}}, \boldsymbol{c} \in [\ell]^{\mathcal{J}}, \boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}, \; U_{ij} = U^*_{r_\imath c_\jmath}, \; \imath \in \mathcal{I}, \jmath \in \mathcal{J}\}.$$

If we set $m = |\{\imath_t : t \in [T]\}|$ and $n = |\{\jmath_t : t \in [T]\}|$, then given comparator matrix $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}}$, we define $\dot{\boldsymbol{U}} \in \mathbb{B}_{k,\ell}^{m,n}$, such that $\dot{\boldsymbol{U}}_{i_t,j_{t'}} = \boldsymbol{U}_{\imath_t,\jmath_{t'}}$ for $t, t' \in [T]$ where $i_t = \min\{s : \imath_s = \imath_t\}$ and $j_t = \min\{s : \jmath_s = \jmath_t\}$. The importance of side information is also highlighted in this setting. In the absence of side information, the MEG and MGD bounds become linear and quadratic respectively with respect to the matrix dimension T, making them vacuous.

In the following examples, we receive a row and column vector $\imath_t, \jmath_t \in \mathfrak{R}^d \times \mathfrak{R}^{d'}$ on each trial; these vectors will be the indices to our row and column kernels. We assume that these vectors may be separated into distinct clusters retrospectively, where each cluster corresponds to a row or column latent class. In Section 4.5.1, we will assume box-shaped clusters and apply the min kernel on the row and column vectors, obtaining MEG mistake bounds of $\tilde{O}(\min(k,\ell)\max(k^2,\ell^2))$. In Section 4.5.2, we apply the Gaussian kernel on ball-shaped clusters, and obtain MEG mistake bounds of $\tilde{O}(\min(k,\ell)\max(k,\ell)^2\,((1+\sqrt{3})\max(k,\ell))^{\frac{6\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}})$, where ρ and δ_2^* are properties of the clusters.

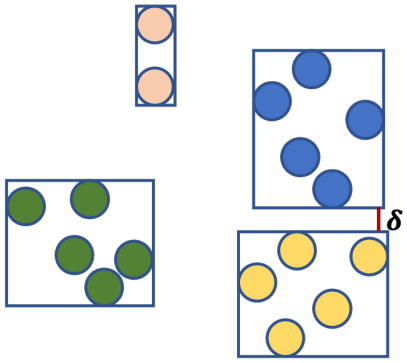


Figure 4.3: Side information vectors for $d = 2$, with the corresponding separating boxes.

4.5.1 Side Information in Boxes

In the following, we show an example for predicting a matrix $\boldsymbol{U} \in \mathbb{B}^{\mathcal{I},\mathcal{J}}_{k,\ell}$ such that for online side information in $[-r, r]^d$ that is well-separated into *boxes*, there exists a kernel for which the quasi-dimension grows no more than quadratically with the number of latent factors (but exponentially with the dimension d). The online side information is given by a row and column vector $\iota_t, \jmath_t \in [-r, r]^d \times [-r', r']^{d'}$ which we receive on each trial, and for simplicity we set $r = r'$ and $d = d'$. Fig. 4.3 illustrates side information vectors for $d = 2$ and the boxes that separate these. For simplicity, we use the *min* kernel, which approximates functions by linear interpolation.

Bounding $\mathcal{D}^\circ$ for the *min* kernel. Define the transformation $s(\boldsymbol{x}) := \frac{r-1}{2r}\boldsymbol{x} + \frac{r+1}{2}$ and the *min* kernel $\mathcal{K} : [0, r]^d \times [0, r]^d \to \mathfrak{R}$ as $\mathcal{K}(\boldsymbol{x}, \boldsymbol{t}) := \prod_{i=1}^d \min(x_i, t_i)$. Also define $\delta_\infty(S_1, \ldots, S_k) := \min_{1 \le i < j \le k} \min_{\boldsymbol{x} \in S_i, \boldsymbol{x}' \in S_j} \|\boldsymbol{x} - \boldsymbol{x}'\|_\infty$. A *box* in $\mathfrak{R}^d$ is a set $\{\boldsymbol{x} : a_i \le x_i \le b_i, i \in [d]\}$ defined by a pair of vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathfrak{R}^d$.

Proposition 58. *Given k boxes $S_1, \ldots, S_k \subset [-r, r]^d$, $r \ge 2$, $\delta^*_\infty = \min\left(2, \frac{1}{4}\delta(S_1, \ldots, S_k)\right)$, and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \cup_{i=1}^k S_i$, if $\boldsymbol{R} = ([\boldsymbol{x}_i \in S_j])_{i \in [m], j \in [k]}$ and $\boldsymbol{K} = (\mathcal{K}(s(\boldsymbol{x}_i), s(\boldsymbol{x}_j)))_{i,j \in [m]}$ then $\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{K}^{-1} \boldsymbol{R}) \le k \left(\frac{4}{\delta^*_\infty}\right)^d$.*

Recall the bound (see (4.2)) on the quasi-dimension for a matrix $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$, where we have $\mathcal{D} \le \mathcal{D}^\circ = k \operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M} \boldsymbol{R}) \mathcal{R}_M + \ell \operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N} \boldsymbol{C}) \mathcal{R}_N$ for positive definite matrices. If we assume that the side information on the rows (columns) lies in $[-r, r]^d$, then $\mathcal{R}_M \le \mathcal{R}_\mathcal{M} \le r^d$ ($\mathcal{R}_N \le \mathcal{R}_\mathcal{N} \le r^d$) for the min kernel. Thus by applying the above proposition separately for the rows and columns and substituting

into (4.2), we have that

$$\mathcal{D} \le \mathcal{D}^\circ = (k^2 + \ell^2)\left(\frac{4r}{\delta_\infty^*}\right)^d,$$

where $\delta_\infty^* = \min\left\{2, \frac{1}{4}\delta_\infty(S_1, \ldots, S_k), \frac{1}{4}\delta_\infty(\tilde{S}_1, \ldots, \tilde{S}_\ell)\right\} > 0$, $S_1, \ldots, S_k$ are the given k boxes that cluster the row side information vectors, and $\tilde{S}_1, \ldots, \tilde{S}_\ell$ are the ℓ boxes that cluster the column side information vectors.

Corollary 59. *Assume that we receive $\iota_t, \jmath_t \in [-r, r]^d \times [-r, r]^d$ on each trial, where $r \ge 2$, and that there exist k boxes $S_1, \ldots, S_k \subset [-r, r]^d$, and ℓ boxes $\tilde{S}_1, \ldots, \tilde{S}_\ell \subset [-r, r]^d$, such that $\iota_1, \ldots, \iota_T \in \cup_{i=1}^k S_i$, $\jmath_1, \ldots, \jmath_T \in \cup_{i=1}^\ell \tilde{S}_i$ and $\delta_\infty^* = \min\left\{2, \frac{1}{4}\delta_\infty(S_1, \ldots, S_k), \frac{1}{4}\delta_\infty(\tilde{S}_1, \ldots, \tilde{S}_\ell)\right\} > 0$.*

The mistakes of Algorithm 3 in the realizable case with conservative updates and parameters $\widehat{\mathcal{D}} = (k^2 + \ell^2)\left(\frac{4r}{\delta_\infty^*}\right)^d$, $\eta = \frac{1}{\min(\sqrt{k}, \sqrt{\ell})}$, are bounded by,

$$|\mathbb{M}| \le 3.6 \min(k, \ell)\left(k^2 + \ell^2\right)\left(\frac{4r}{\delta_\infty^*}\right)^d \log(m + n),$$

for any $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}}$ for which $\dot{\boldsymbol{U}}$ has a decomposition $\boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ which satisfies $\boldsymbol{R}_{i_t,a} = [\iota_t \in S_a]$ and $\boldsymbol{C}_{j_t,a} = [\jmath_t \in \tilde{S}_a]$, and $y_t = \operatorname{sign}(U_{\iota_t \jmath_t})$ for all $t \in \mathbb{M}$.

The expected mistakes of Algorithm 3 with non-conservative updates and parameters $\widehat{\mathcal{D}} = (k^2 + \ell^2)\left(\frac{4r}{\delta_\infty^*}\right)^d$, $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$, are bounded by

$$\mathbb{E}[|\mathbb{M}|] \le \sum_{t \in [T]} [y_t \ne U_{\iota_t \jmath_t}] + 3.5\sqrt{\min(k, \ell)\left(k^2 + \ell^2\right)\left(\frac{4r}{\delta_\infty^*}\right)^d \log(m + n)T}$$

for any $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}}$ for which $\dot{\boldsymbol{U}}$ has a decomposition $\boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ which satisfies $\boldsymbol{R}_{i_t,a} = [\iota_t \in S_a]$ and $\boldsymbol{C}_{j_t,a} = [\jmath_t \in \tilde{S}_a]$.

Hence we have that, with an optimal tuning and well-separated side information on the rows and columns, the MEG mistake bound for a (k, ℓ)-biclustered matrix in the inductive setting is of $\tilde{O}(\min(k, \ell)\max(k, \ell)^2)$, and the MGD mistake bound is $O(\min(k, \ell)\max(k, \ell)^4)$. By using the bound on the quasi-area in Equation (4.3), we obtain an MGD mistake bound of $O(\min(k, \ell)k^2\ell^2)$. However, our

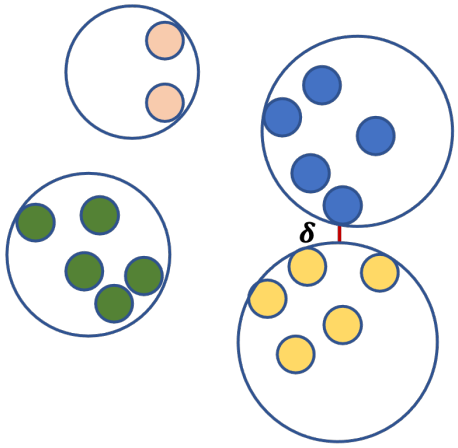


Figure 4.4: Side information vectors for $d = 2$, with the corresponding separating balls.

best lower bound in terms of k and ℓ is just $k\ell$, as in the transductive setting. An open problem is to resolve this gap.

4.5.2 Side Information in Balls

In the following, we show an example for predicting a matrix $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}}$ for side information that is well-separated into balls. Similar to the min kernel example, we receive a row and column vector $\iota_t, \jmath_t \in \mathfrak{R}^d \times \mathfrak{R}^{d'}$ on each trial; these vectors will be the indices to our row and column kernels, and for simplicity we set $d = d'$. Fig. 4.4 illustrates example side information vectors and their separating balls for the $d = 2$ case. For this example, we use the *Gaussian* kernel. Instead of using Theorem 57 to bound the quasi-dimension which only holds for binary $\boldsymbol{U}$, we will use Proposition 60, which bounds the quasi-dimension for a real-valued matrix $\bar{\boldsymbol{U}}$ that is sign-consistent with $\boldsymbol{U}$. This is done to simplify the downstream analysis, which requires the interpolation of points defined by the data using RKHS functions of the Gaussian kernel. If we were to use Theorem 57, the interpolation values would be at exactly 0 and 1. However, Proposition 60 allows for interpolation values to be between 0 and maximum height h, which is more natural for the Gaussian kernel.

This proposition requires that

$$\|\bar{\boldsymbol{U}}\|_{\max} \leq (h+1)^2 \min(\sqrt{k}, \sqrt{\ell}),$$

where there is an additional multiplicative $(h+1)^2$ term as compared with its binary counterpart in (4.1). Under this condition, the proposition bounds $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\bar{\boldsymbol{U}}) \leq O\left(k\,\mathcal{R}_{\boldsymbol{M}}\,\mathrm{tr}(\tilde{\boldsymbol{R}}^{\top}\boldsymbol{M}\tilde{\boldsymbol{R}}) + \ell\,\mathcal{R}_{\boldsymbol{N}}\,\mathrm{tr}(\tilde{\boldsymbol{C}}^{\top}\boldsymbol{N}\tilde{\boldsymbol{C}})\right)$ where $\tilde{\boldsymbol{R}}$ and $\tilde{\boldsymbol{C}}$ are the real-valued counterparts to the block-expansion matrices $\boldsymbol{R}$ and $\boldsymbol{C}$ in the decomposition of the binary biclustered matrix $\boldsymbol{U}$.

Proposition 60. *For $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$ with decomposition $\boldsymbol{R}\boldsymbol{U}^{*}\boldsymbol{C}^{\top}$ where $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{U}^{*} \in \{-1,1\}^{k\times\ell}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$, $k > 1$, $\ell > 1$, and for $\frac{1}{\gamma} = (h+1)^2 \min(\sqrt{k}, \sqrt{\ell})$, there exists a matrix $\bar{\boldsymbol{U}} \in \mathrm{SP}^{1}(\boldsymbol{U})$, such that $\|\bar{\boldsymbol{U}}\|_{max} \leq \frac{1}{\gamma}$ and*

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\bar{\boldsymbol{U}}) \leq \frac{1}{3}\left(k\,\mathcal{R}_{\boldsymbol{M}}\,\mathrm{tr}(\tilde{\boldsymbol{R}}^{\top}\boldsymbol{M}\tilde{\boldsymbol{R}}) + \ell\,\mathcal{R}_{\boldsymbol{N}}\,\mathrm{tr}(\tilde{\boldsymbol{C}}^{\top}\boldsymbol{N}\tilde{\boldsymbol{C}})\right),$$

for any $\tilde{\boldsymbol{R}}$ and $\tilde{\boldsymbol{C}}$ where

$$\tilde{\boldsymbol{R}}_{ij} \in \begin{cases} [1+\sqrt{3}, h] & \text{if } R_{ij} = 1 \\ [0, \frac{1}{k-1}] & \text{if } R_{ij} = 0 \end{cases} \quad \text{and} \quad \tilde{\boldsymbol{C}}_{ij} \in \begin{cases} [1+\sqrt{3}, h] & \text{if } C_{ij} = 1 \\ [0, \frac{1}{\ell-1}] & \text{if } C_{ij} = 0 \end{cases}.$$

Bounding $\mathcal{D}$ for the *Gaussian* kernel. Define the *Gaussian* kernel $\mathcal{K}_{\beta} : \mathfrak{R}^d \times \mathfrak{R}^d \to \mathfrak{R}$ as $\mathcal{K}_{\beta}(\boldsymbol{x},\boldsymbol{t}) := \exp(-\beta\|\boldsymbol{x}-\boldsymbol{t}\|^2)$. Also define $\delta_2(S_1,\ldots,S_k) := \min_{1\leq i<j\leq k} \min_{\boldsymbol{x}\in S_i, \boldsymbol{x}'\in S_j} \|\boldsymbol{x}-\boldsymbol{x}'\|_2$. A ball in $\mathfrak{R}^d$ is the set of points $\{\boldsymbol{x} : \|\boldsymbol{x}-\boldsymbol{t}\|_2 \leq \rho\}$, defined by the centre $\boldsymbol{t} \in \mathfrak{R}^d$ and radius ρ.

Proposition 61. *Given k balls $S_1,\ldots,S_k$, with respective radii $\rho_1,\ldots,\rho_k$, maximal radius $\rho = \max_{i\in[k]}\rho_i$, $\delta_2^{*} = \delta(S_1,\ldots,S_k)$, $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_m \in \cup_{i=1}^{k} S_i$, and $\boldsymbol{K} = (\mathcal{K}_{\beta}(\boldsymbol{x}_i,\boldsymbol{x}_j))_{i,j\in[m]}$, where $\beta = \frac{1}{(2\rho\delta_2^{*}+(\delta_2^{*})^2)d}\log((1+\sqrt{3})(k-1))$, there exists an $\tilde{\boldsymbol{R}} \in \mathfrak{R}^{m\times k}$ such that*

$$\tilde{\boldsymbol{R}}_{ij} \in \begin{cases} [1+\sqrt{3}, h] & \text{if } \boldsymbol{x}_i \in S_j \\ [0, \frac{1}{k-1}] & \text{if } \boldsymbol{x}_i \notin S_j \end{cases}$$

and $\mathrm{tr}(\tilde{\boldsymbol{R}}^{\top}\boldsymbol{K}^{-1}\tilde{\boldsymbol{R}}) \leq kh^2$ where $h = (1+\sqrt{3})\left((1+\sqrt{3})(k-1)\right)^{\frac{\rho^2}{2\rho\delta_2^{*}+(\delta_2^{*})^2}}$.

Applying Proposition 61 to Proposition 60, and observing that for the Gaussian

kernel $\mathcal{R}_M = \mathcal{R}_N = 1$, we have

$$\mathcal{D}(\bar{\boldsymbol{U}}) \leq \frac{1}{3}(k^2 + \ell^2)h^2,$$

where $h := (1 + \sqrt{3})\left((1 + \sqrt{3})\max(k-1, \ell-1)\right)^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}}$, δ_2^* is the smallest separation between any two given balls, and ρ is the largest radius of a ball. Using the fact that $\operatorname{sign}(\bar{U}_{i_t j_t}) = \operatorname{sign}(U_{i_t j_t})$ and $\max_{i,j} |A|_{ij} \leq \|\boldsymbol{A}\|_{\max}$ for any matrix $\boldsymbol{A}$, we derive the following corollary.

Corollary 62. *Assume that we receive $\imath_t, \jmath_t \in \mathfrak{R}^d \times \mathfrak{R}^d$ on each trial, and that there exist k balls $S_1, \ldots, S_k \subset \mathfrak{R}^d$ with respective radii $\rho_1, \ldots, \rho_k$, and ℓ balls $\tilde{S}_1, \ldots, \tilde{S}_\ell \subset \mathfrak{R}^d$ with respective radii $\tilde{\rho}_1, \ldots, \tilde{\rho}_\ell$, such that $\imath_1, \ldots, \imath_T \in \cup_{i=1}^k S_i$, $\jmath_1, \ldots, \jmath_T \in \cup_{i=1}^\ell \tilde{S}_i$, $\delta_2^* := \min(\delta_2(S_1, \ldots, S_k), \delta_2(\tilde{S}_1, \ldots, \tilde{S}_\ell)) > 0$, $\rho := \max(\max_{i\in[k]} \rho_i, \max_{j\in[\ell]} \tilde{\rho}_j)$ and $h := (1 + \sqrt{3})\left((1 + \sqrt{3})\max(k-1, \ell-1)\right)^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}}$.*

The mistakes of Algorithm 3 in the realizable case with conservative updates and parameters $\widehat{\mathcal{D}} = \frac{1}{3}(k^2 + \ell^2)h^2$, $\eta = \frac{1}{(h+1)^2 \min(\sqrt{k}, \sqrt{\ell})}$, are then bounded by,

$$|\mathbb{M}| \leq 1.2\,(h+1)^4 h^2 \min(k, \ell)(k^2 + \ell^2)\log(m+n)\,,$$

for any $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}}$ for which $\dot{\boldsymbol{U}}$ has a decomposition $\boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ which satisfies $\boldsymbol{R}_{i_t,a} = [\imath_t \in S_a]$ and $\boldsymbol{C}_{j_t,a} = [\jmath_t \in \tilde{S}_a]$, and $y_t = \operatorname{sign}(U_{\imath_t \jmath_t})$ for all $t \in \mathbb{M}$.

The expected mistakes of Algorithm 3 with non-conservative updates and parameters $\widehat{\mathcal{D}} = \frac{1}{3}(k^2 + \ell^2)h^2$, $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$, are then bounded by

$$\mathbb{E}[|\mathbb{M}|] \leq \frac{1 + (h+1)^2 \min(\sqrt{k}, \sqrt{\ell})}{2} \sum_{t\in[T]} [y_t \neq U_{i_t j_t}] + 3.5\sqrt{\frac{(h+1)^4 h^2}{3} \min(k, \ell)\,(k^2 + \ell^2)\log(m+n)T}$$

for any $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{\mathcal{I},\mathcal{J}}$ for which $\dot{\boldsymbol{U}}$ has a decomposition $\boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ which satisfies $\boldsymbol{R}_{i_t,a} = [\imath_t \in S_a]$ and $\boldsymbol{C}_{j_t,a} = [\jmath_t \in \tilde{S}_a]$.

We therefore have an MEG mistake bound of $\tilde{O}(\min(k, \ell)\max(k, \ell)^2 h^6)$.

4.5.3 Balls vs. Boxes

It is hard to directly give a meaningful comparison between the Gaussian kernel mistake bound in Corollary 62 with the min kernel bound in Corollary 59, since the assumption that the side information is well-separated into boxes does not imply that it can be well-separated into balls and vice versa. For ease of comparison, we shall therefore assume that the side information can be well-separated into both balls and boxes.

We first note that the Gaussian kernel and min kernel bounds scale similarly except that the former has a h^6 term whereas the latter has a $(4r/\delta_\infty^*)^d$ term. In the case where $\rho = \delta_2^*$, $h^6 \in O(\max(k,\ell)^2)$, and the Gaussian kernel bound has a superior scaling when $O(\max(k,\ell)^2) < O((4r/\delta_\infty^*)^d)$. More generally, observing that $\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2} \leq \min(\frac{\rho}{2\delta_2^*}, \frac{\rho^2}{(\delta_2^*)^2})$, we have that the Gaussian kernel mistake bound term scales with $\left((1+\sqrt{3})(k-1)\right)^{\frac{3\rho}{\delta_2^*}}$. For the min kernel bound, assuming that $k > \ell$, and that the largest box dimension $a \geq \delta_\infty^*$, we can roughly bound $r \leq 2ka$ so that the mistake bound scales with $(\frac{8ka}{\delta_\infty^*})^d$. Hence, we have that the ratio $\frac{\rho}{\delta_2^*}$ scales exponentially in the Gaussian bound as opposed to the polynomial scaling of $\frac{a}{\delta_\infty^*}$. However, the min kernel bound scales exponentially with d, whereas there is no dependence on d in the Gaussian bound.

The regret bounds are harder to compare, owing to the fact that the regret bound with the Gaussian kernel in Corollary 62 has a constant factor of $(h+1)^2 \min(\sqrt{k}, \sqrt{\ell})$ in front of the comparator matrix whereas the min kernel bound in Corollary 59 does not.

4.6 Synthetic Experiments

To illustrate the algorithm's performance, synthetic experiments were performed in the transductive setting with graph side information. We assess the performance of the algorithm with varying levels of side information noise, and apply a sketching method [80] which approximates the exponential in our algorithm and offers an improved time complexity.

In particular, the comparator matrix $\boldsymbol{U}$ is sampled uniformly at random from

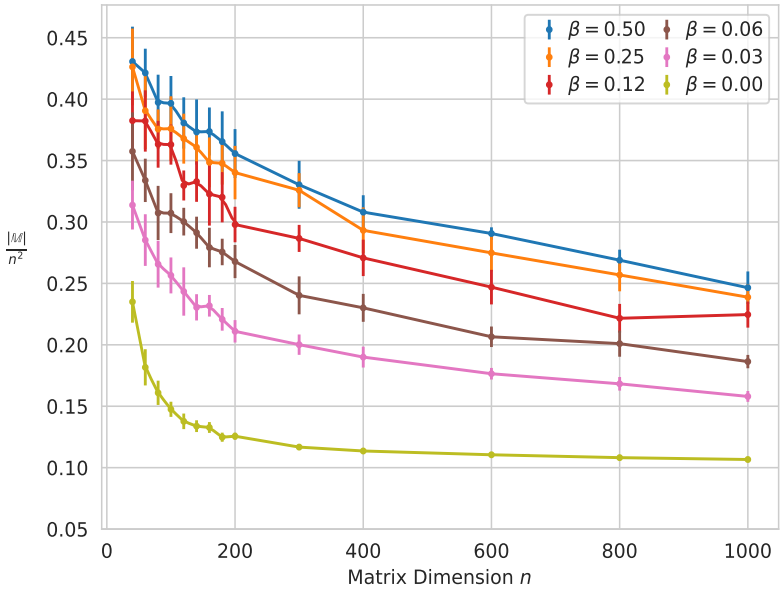


Figure 4.5: Error rates for predicting a noisy $(9, 9)$-biclustered matrix with side information, with side information noise $\beta \in [0.0, 0.5]$.

the set of all square (9,9)-biclustered matrices, after which i.i.d. noise is added. A visualization of a noise-free example matrix can be found in Figure 4.1. The noise process flipped the label of each matrix entry independently with probability $p = 0.10$.

4.6.1 Side information Noise

The side information on the rows and columns are represented by PDLaplacian matrices, for which the underlying graphs were constructed in the manner described in Section 4.4.1 (see Fig. 4.2). For this experiment, varying levels of side information noise $\beta \in [0.0, 0.5]$ were applied. This was introduced by considering every pair of vertices independently from the constructed graph and flipping the state between EDGE/NOT-EDGE with probability β. A final step is added to ensure the graph is connected. In this step a random pair of components is connected by a random edge, recursively. The process terminates when the graph is connected.

The parameters were chosen so that the expected regret bound in Theorem 1 would apply to our experimental setting. We use the quasi-dimension upper bound

$\widehat{\mathcal{D}} := \mathcal{D}^{\circ}_{M,N}(U) = 2\,\mathrm{tr}(R^{\top}MR)\mathcal{R}_M + 2\,\mathrm{tr}(C^{\top}NC)\mathcal{R}_N + 4k$, as developed in Theorem 57 for PDLaplacians. The learning rate was set as $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(2n)}{2T}}$. Since each run of the algorithm consisted of predicting all n^2 matrix entries sampled uniformly at random without replacement, we set $T = n^2$. As for the margin estimate, due to the requirement that $\gamma \leq 1/\|U\|_{\max}$, a suitable value can be extracted from Equation (4.1), giving $\gamma = 1/\sqrt{k}$.

The per trial mistake rate is shown in Fig. 4.5 for matrix dimension $n = 40, \ldots, 400$, where each data point is averaged over 10 runs. We observe that for random side information $\beta = 0.5$, the term $\widehat{\mathcal{D}}$ could lead to a bound which is vacuous (for small n), however, the algorithm's error rate was in the range of $[0.25, 0.45]$, being well below chance. With ideal side information, $\beta = 0.0$, the performance improved drastically, as suggested by the bounds, to an error rate in $[0.10, 0.25]$. Observe that since there is 10% label noise for all values of β, the curves are converging to an online mistake rate of 10%. The data points for the plot can be found below. For our implementation of $n = 1000$ at a noise level of $\beta = 0.5$, 10 runs required approximately 37 hours on an Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz with 4GB of RAM and a NVIDIA Tesla P100 GPU. At a noise level of $\beta = 0.0$ (no noise), 10 runs required approximately 27 hours. Note that this was run on a shared node on a cluster, meaning that the run time may vary depending on the other jobs that were running.

4.6.2 Sketching Method

This experiment aims to assess the effectiveness of applying the sketching method [80] to our algorithm, which reduces the computational complexity by approximating the computation of the dominant step, the exponential on $\log(\tilde{W}^t)$. It approximates the term $\exp(\log(\tilde{W}^t)) \approx \exp(\log(\tilde{W}^t)/2)RR^{\top}\exp(\log(\tilde{W}^t)/2)$, where $R \in \Re^{2n\times\kappa}$ has entries sampled from a standard Gaussian disitribution. The term $\exp(\log(\tilde{W}^t)/2)R$ can be evaluated efficiently through the iterative Lanczos method, which has a time complexity in $O(\kappa n^2 s + s^2)$, where s is the number of iterations. We performed experiments with the same parameters as those in the previous section, with $\kappa \in \{1, 2, 5, 10, 20, 100\}$, and no side information noise $\beta = 0.0$.

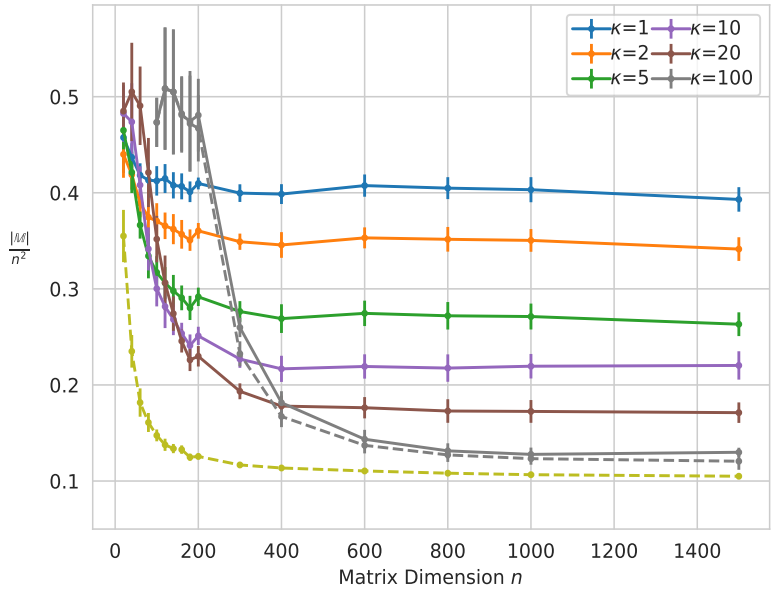


Figure 4.6: Error rates for predicting a noisy $(9, 9)$-biclustered matrix with side information with the sketching method. The bottom curve represents the error rates with a direct exponential computation.

For $\kappa \in \{1, 2, 5\}$, we required more iterations with $s = 20$, whereas for $\kappa \in \{10, 20\}$, we only did $s = 10$ iterations, and for $\kappa = 100$, we performed only 5 iterations. At this value of κ, we also compare against the methodology in [81], which evaluates an approximate matrix exponential vector product (as shown by the dotted line). The time complexity of this method is $O(\kappa n^2 s)$, where s is dependent on the 1-norm of $\tilde{W}^t$.

The plots in Fig. 4.6 shows the error rates for the different values of κ. We observe that using only $\kappa = 1$, as proposed in [80], yields error rates which plateau around 0.4. As expected, higher κ converge to smaller error rates, with $\kappa = 100$ converging to error rates below 0.15. In the regime that we explored, we found that the run time with the sketching method was only faster for $\kappa \in \{1, 2\}$. Using the same hardware in the previous section, each run for $n = 1500$ and $\kappa = 1, 2$ and 100 took on average around 6.9, 11.0 and 54.8 hours respectively, compared to 11.1 hours for the direct exponential computation. However, the sketching method has a lower time complexity, suggesting that the method may result in quicker computation times for

	Min Kernel Bound	Gaussian Kernel Bound
MEG	$\tilde{O}(\min(k,\ell)\max(k,\ell)^2(\frac{4r}{\delta_\infty^*})^d)$	$\tilde{O}(\min(k,\ell)\max(k,\ell)^2 h^6)$
MGD sum squared kernel	$O(\min(k,\ell)\max(k,\ell)^4(\frac{4r}{\delta_\infty^*})^{2d})$	$O(\min(k,\ell)\max(k,\ell)^4 h^8)$
MGD product kernel	$O(\min(k,\ell)k^2\ell^2(\frac{4r}{\delta_\infty^*})^{2d})$	$O(\min(k,\ell)k^2\ell^2 h^8)$

Table 4.1: The mistake bounds of the MEG and MGD algorithms in the inductive setting as applied to biclustered matrices. We recall that $h = O(((1+\sqrt{3})\max(k,\ell))^{\frac{\rho^2}{2\rho\delta_2^*+(\delta_2^*)^2}})$.

larger matrices.

4.7 Discussion

In this chapter, we applied the bounds developed in Chapter 3 to the hypothesis class of (k,ℓ)-biclustered matrices. We recall that the MEG and MGD algorithms achieve mistake bounds of $\tilde{O}\left(\mathcal{D}/\gamma^2\right)$ and $O\left(\mathcal{D}^2/\gamma^2\right)$ respectively. For this hypothesis class, the margin term $1/\gamma^2 \leq \min(k,\ell)$ when exactly tuned as the margin complexity. We bounded the $\mathcal{D}$ term for various examples. In the transductive setting with ideal graph-based side information, $\mathcal{D} \in O(\max(k,\ell))$. When applying our bound to the problem of online community membership prediction, we recovered and extended the result in [38]. The synthetic experiments corroborate our theoretical results and suggest the possibility of applying the sketching method on our algorithm.

In Section 3.2.2, we discussed two possible embeddings: one which has a product kernel in its dual form (see (3.10)), and one which has a sum squared kernel (see (3.12)). Observing that we can apply the quasi-area bound in (4.3) for the product kernel, but can use the tighter quasi-dimension bound (4.2) for the sum squared kernel, we then show a comparison of the two kernels against MEG in Table 4.1. We observe that the predictions for both kernels in the GD algorithm naturally extend to the tensor case of order p. Given kernels $\mathcal{M}_1^+, \ldots, \mathcal{M}_p^+$ and side

information vectors $\iota^1, \dots \iota^p$, the predictions on trial t would be in the form of

$$\hat{y}_t = \eta \sum_{s \in \mathbb{U}_t} y_s \prod_{j=1}^{p} \frac{1}{\mathcal{R}_{\mathcal{M}_j}} \mathcal{M}_j^+(\iota_t^j, \iota_s^j) \qquad \text{(product kernel)} \tag{4.4}$$

$$\hat{y}_t = \eta \sum_{s \in \mathbb{U}_t} y_s \left(\sum_{j=1}^{p} \frac{1}{2\mathcal{R}_{\mathcal{M}_j}} \mathcal{M}_j^+(\iota_t^j, \iota_s^j) \right)^2 \qquad \text{(sum squared kernel)}\,. \tag{4.5}$$

It is not difficult to write the bounds. The problem, as highlighted in Table 4.1, is that the GD bounds can be vacuous in cases when the EG bound is not. In what follows, we hypothesize that bounds can be proven for an efficient algorithm completing a tensor of order p that has a generalized notion of biclustering, where we now receive p side information vectors that are well-separated in balls or boxes, and we use min or Gaussian kernels to inform similarity.

We first introduce our generalized notion of biclustering for tensors. The class of $(k_1, \dots, k_p)$-binary-clustered tensors of order p is defined as

$$\mathbb{B}_{k_1,\dots,k_p}^{m_1,\dots,m_p} = \{\boldsymbol{U} \in \{-1,1\}^{m_1 \times \dots \times m_p} : \\ \boldsymbol{r}^q \in [k_q]^{m_q},\ \boldsymbol{U}^* \in \{-1,1\}^{k_1 \times \dots \times k_p},\ U_{i_1 \dots i_p} = U^*_{r^1_{i_1} \dots r^p_{i_p}}, q \in [p]\}\,.$$

For simplicity, we assume $k = k_1 = \dots = k_p$. On each trial t, we receive side information vectors $\iota_t^1, \dots, \iota_t^p \in \mathfrak{R}^d$. We have the realizable case when $y_t = U_{i_t^1,\dots,i_t^p}$ for some $\boldsymbol{U} \in \mathbb{B}_{k,\dots,k}^{m_1,\dots,m_p}$, where $i_t^q = \min\{s : \iota_s^q = \iota_t^q\}$ for all $q \in [p]$. Assume that for all $q \in [p]$, the vectors $\{\iota_t^q\}_{t \in [T]}$ from all T trials can be well-separated into k clusters $S_1^q, \dots, S_k^q$, where if vectors ι_s^q and ι_t^q are in the same cluster, then they correspond to the same latent class, i.e. $r_{i_s}^q = r_{i_t}^q$ in the definition of the biclustered tensor. Define $\delta_\infty(S_1^1, \dots, S_k^p) = \min_{q \in [p]} \min_{1 \le a < b \le k} \min_{\boldsymbol{x} \in S_a^q, \boldsymbol{x}' \in S_b^q} \|\boldsymbol{x} - \boldsymbol{x}'\|_\infty$ and similarly, $\delta_2(S_1^1, \dots, S_k^p) = \min_{q \in [p]} \min_{1 \le a < b \le k} \min_{\boldsymbol{x} \in S_a^q, \boldsymbol{x}' \in S_b^q} \|\boldsymbol{x} - \boldsymbol{x}'\|_2$.

Conjecture 63. *If the clusters are box-shaped, then there exists a polynomial-time algorithm for which the number of mistakes in the realizable case can be bounded by* $\tilde{O}(k^{p+1}(\frac{4r}{\delta_\infty})^d)$*, where* $\delta_\infty^* := \min(2, \frac{1}{4}\delta_\infty(S_1^1, \dots, S_k^p))$*, and* $r \ge 2$ *is the largest* ℓ_∞*-norm of* ι_t^q *for all* q *and* t*. If the clusters are ball-shaped, then there exists a*

polynomial-time algorithm for which the number of mistakes in the realizable case can be bounded by $\tilde{O}(k^{p+1}(h'_p)^{p(p+1)})$*, where we define* $h'_p := c_p(c_p(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}}$*,* c_p *is a constant which increases with* p*,* ρ *is the radius of the largest ball and* $\delta_2^* := \delta_2(S^1_1, \ldots, S^p_k)$*.*

The conjecture is motivated by the trends which follow from the bounds for the first-order tensor (vector) and the second-order tensor (matrix) cases. For the first-order tensor case, the task is to complete a binary vector, where we are given a single side information vector ι_t on each trial t and a single kernel function which informs similarity between the entries $\mathcal{M}^+$. The predictions are given by (4.5) where $p = 1$, for which the kernel perceptron algorithm gives mistake bounds of the form $O(\min_{h \in \mathcal{H}_{\mathcal{M}^+} : h(\iota_t) y_t \geq 1} \|h\|^2_{\mathcal{M}^+} X^2_{\mathcal{M}^+})$, where $X^2_{\mathcal{M}^+} = \max_t \mathcal{M}^+(\iota_t, \iota_t)$. If the side information vectors can be well-separated into boxes $S_1, \ldots S_k$, then we can consider the hypothesis $h(x) = \sum_{i \in [k]} \alpha_i f_i(x)$, where $\alpha_i \in \{-1, 1\}$, $f_i(x) = [x \in S_i]$, and we have that $\|h\|^2_{\mathcal{M}^+} = \|(\sum_{i \in [k]} \alpha_i f_i(x))\|^2 \leq (\sum_{i \in [k]} \alpha_i \|f_i(x)\|)^2$. By Lemma 65, we have that $\|f_i(x)\|^2 \leq (\frac{4}{\delta_\infty^*})^d$ for all i, so that $\|h\|^2_{\mathcal{M}^+} \leq (\frac{4}{\delta_\infty^*})^d k^2$. Combining with $X^2_{\mathcal{M}^+} \leq r^d$, the mistake bound can thus be written as $O\left(k^2 \left(4r/\delta_\infty^*\right)^d\right)$. Similarly, if the side information vectors can be separated into balls $S_1, \ldots, S_k$, we can consider the hypothesis $h(x) = \sum_{i \in [k]} \alpha_i f_i(x)$, where $\alpha_i \in \{-1, 1\}$, $f_i(x) \in \begin{cases} [2, h'_1] & \text{if } x \in S_i \\ [0, \frac{1}{k-1}] & \text{otherwise} \end{cases}$, and $h'_1 := 2(2(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}}$. Using a similar analysis to that in Lemma 68, we obtain $\|f_i(x)\|^2 \leq (h'_1)^2$ for $\beta := \frac{1}{(2\rho\delta_2^* + (\delta_2^*)^2)d} \log(2(k-1))$. Hence, the mistakes are in $O\left(k^2 (h'_1)^2\right)$. For the matrix $p = 2$ case, we consider the bounds given by the MEG algorithm. It remains an open problem to prove this conjecture for higher orders of p.

4.8 Proofs

4.8.1 Proof of Theorem 57

We recall Theorem 57 and then prove it.

Theorem 57. *If* $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$ *define*

$$\mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \begin{cases} 2\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + 2\operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} + 2k + 2\ell & \boldsymbol{M},\ \boldsymbol{N} \text{ are PDLaplacians} \\ k\operatorname{tr}(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R})\mathcal{R}_{\boldsymbol{M}} + \ell\operatorname{tr}(\boldsymbol{C}^\top \boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} & \boldsymbol{M} \in \boldsymbol{S}^m_{++} \text{ and } \boldsymbol{N} \in \boldsymbol{S}^n_{++} \end{cases},$$

as the minimum over all decompositions of $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ *for* $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ *and* $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$. *Thus for* $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$,

$$\begin{aligned} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) &\le \mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \qquad (\textit{if } \|\boldsymbol{U}\|_{max} \le 1/\gamma) \\ \min_{\boldsymbol{V}\in \mathrm{SP}^1(\boldsymbol{U})} \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{V}) &\le \mathcal{D}^{\circ}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \qquad (\textit{if } \operatorname{mc}(\boldsymbol{U}) \le 1/\gamma)\,. \end{aligned}$$

Proof. A γ-*decomposition* of matrix $\boldsymbol{U}$ is given by a $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and a $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ such that $\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma\boldsymbol{U}$. A *block-invariant decomposition* of matrix $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$ is given by a $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and a $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ for some d such that there exists a $\delta \in (0,1]$, $\hat{\boldsymbol{P}}^* \in \mathcal{N}^{k,d}$, and a $\hat{\boldsymbol{Q}}^* \in \mathcal{N}^{\ell,d}$, so that $\hat{\boldsymbol{P}} = \boldsymbol{R}\hat{\boldsymbol{P}}^*$, $\hat{\boldsymbol{Q}} = \boldsymbol{C}\hat{\boldsymbol{Q}}^*$ and $\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \delta\boldsymbol{U}$.

We now prove the following intermediate result,

> **Lemma:** If $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$, then for every $\gamma \in (0, 1/\|\boldsymbol{U}\|_{\max})$, there exists a block-invariant γ-decomposition of $\boldsymbol{U}$.

> *Proof.* Since $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$ we have that $\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ for some $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$ and $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$. Observe by block invariance we have that $\|\boldsymbol{U}\|_{\max} = \|\boldsymbol{U}^*\|_{\max}$ and by the definition of $\|\cdot\|_{\max}$ we have that there exists a $\left(\frac{1}{\|\boldsymbol{U}\|_{\max}}\right)$-decomposition of $\boldsymbol{U}^*$ via factors $\hat{\boldsymbol{P}}^* \in \mathcal{N}^{k,d}$, and a $\hat{\boldsymbol{Q}}^* \in \mathcal{N}^{\ell,d}$, this implies that $\hat{\boldsymbol{P}} := \boldsymbol{R}\hat{\boldsymbol{P}}^*$, $\hat{\boldsymbol{Q}} := \boldsymbol{C}\hat{\boldsymbol{Q}}^*$ is a $\left(\frac{1}{\|\boldsymbol{U}\|_{\max}}\right)$-block-invariant decomposition of $\boldsymbol{U}$.

Now given any $\gamma \in (0, 1/\|\boldsymbol{U}\|_{\max})$ we construct a γ-block-invariant decomposition of $\boldsymbol{U}$. Set $c := \gamma\|\boldsymbol{U}\|_{\max}$. We construct new factor matrices $\hat{\boldsymbol{P}}' \in \mathcal{N}^{m,d+1}$ and $\hat{\boldsymbol{Q}}' \in \mathcal{N}^{n,d+1}$

$$\hat{\boldsymbol{P}}' := \begin{pmatrix} c\hat{\boldsymbol{P}} & (\sqrt{1-c^2})\mathbf{1} \end{pmatrix}; \qquad \hat{\boldsymbol{Q}}' := \begin{pmatrix} \hat{\boldsymbol{Q}} & \mathbf{0} \end{pmatrix}.$$

Observe that $(\hat{\boldsymbol{P}}', \hat{\boldsymbol{Q}}')$ is the required γ-block-invariant decomposition of $\boldsymbol{U}$. □

Recall (5.6),

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^{\top}=\gamma\boldsymbol{U}} \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^{\top}\boldsymbol{M}\hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^{\top}\boldsymbol{N}\hat{\boldsymbol{Q}}\right). \tag{4.6}$$

Observe that when the feasible set of the optimization that defines $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ is non-empty and $\boldsymbol{U} \in \mathbb{B}^{m,n}_{k,\ell}$, there exists a member of the feasible set which is a block-invariant decomposition of $\boldsymbol{U}$ by the lemma above. We proceed by proving an upper bound of

$$\mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^{\top}\boldsymbol{M}\hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^{\top}\boldsymbol{N}\hat{\boldsymbol{Q}}\right)$$

for every block-invariant decomposition of $\boldsymbol{U}$.

First we will bound the term $\operatorname{tr}(\hat{\boldsymbol{P}}^{\top}\boldsymbol{M}\hat{\boldsymbol{P}})$ for general positive definite matrices and then for PDLaplacians. By symmetry, the bound will also hold for $\operatorname{tr}(\hat{\boldsymbol{Q}}^{\top}\boldsymbol{N}\hat{\boldsymbol{Q}})$.

Suppose $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}})$ is a block-invariant decomposition of $\boldsymbol{U}$. Then, we have

$$\begin{aligned} \operatorname{tr}(\hat{\boldsymbol{P}}^{\top}\boldsymbol{M}\hat{\boldsymbol{P}}) &= \operatorname{tr}((\boldsymbol{R}\hat{\boldsymbol{P}}^{*})^{\top}\boldsymbol{M}\boldsymbol{R}\hat{\boldsymbol{P}}^{*}) = \operatorname{tr}(\hat{\boldsymbol{P}}^{*}(\hat{\boldsymbol{P}}^{*})^{\top}\boldsymbol{R}^{\top}\boldsymbol{M}\boldsymbol{R}) \\ &\leq \operatorname{tr}(\hat{\boldsymbol{P}}^{*}(\hat{\boldsymbol{P}}^{*})^{\top}) \operatorname{tr}(\boldsymbol{R}^{\top}\boldsymbol{M}\boldsymbol{R}) = k \operatorname{tr}(\boldsymbol{R}^{\top}\boldsymbol{M}\boldsymbol{R}), \end{aligned}$$

where the inequality comes from the fact that $\operatorname{tr}(\boldsymbol{A}\boldsymbol{B}) \leq \lambda_{\max}(\boldsymbol{A}) \operatorname{tr}(\boldsymbol{B}) \leq \operatorname{tr}(\boldsymbol{A}) \operatorname{tr}(\boldsymbol{B})$ for $\boldsymbol{A}, \boldsymbol{B} \in \boldsymbol{S}_{+}$. By symmetry we have demonstrated the inequality for positive definite matrices.

We now consider PDLaplacians. Assume $\boldsymbol{M} := \boldsymbol{L}^{\circ} = \boldsymbol{L} + \left(\frac{1}{m}\right)\left(\frac{1}{m}\right)^{\top} \mathcal{R}_{\boldsymbol{L}}^{-1}$, a PD-Laplacian. Recall the following two elementary inequalities from the preliminaries

(Section 1.3): if $\boldsymbol{u} \in [-1,1]^m$, then

$$(\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u})\mathcal{R}_{\boldsymbol{L}} \leq \frac{1}{2}(\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u})\mathcal{R}_{\boldsymbol{L}^\circ}\,, \tag{4.7}$$

$$(\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u})\mathcal{R}_{\boldsymbol{L}^\circ} \leq 2(\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u}\,\mathcal{R}_{\boldsymbol{L}} + 1)\,. \tag{4.8}$$

Observe that for an $m \times m$ graph Laplacian $\boldsymbol{L}$ with adjacency matrix $\boldsymbol{A}$ that for $\boldsymbol{X} \in \mathfrak{R}^{m\times d}$,

$$\operatorname{tr}(\boldsymbol{X}^\top \boldsymbol{L}\boldsymbol{X}) = \sum_{(i,j)\in E} A_{ij} \left\|\boldsymbol{X}_i - \boldsymbol{X}_j\right\|^2 \,. \tag{4.9}$$

Suppose $(\hat{\boldsymbol{P}}, \hat{\boldsymbol{Q}})$ is a block-invariant decomposition of $\boldsymbol{U}$ then the row vectors $\hat{\boldsymbol{P}}_1, \ldots, \hat{\boldsymbol{P}}_m$ come in at most k distinct varieties, that is $|\bigcup_{i\in[m]} \hat{\boldsymbol{P}}_i| \leq k$. The same holds for $\boldsymbol{R}$ and furthermore $(\hat{\boldsymbol{P}}_r = \hat{\boldsymbol{P}}_s) \iff (\boldsymbol{R}_r = \boldsymbol{R}_s)$ for $r, s \in [m]$. Observe that given $r, s \in [m]$ that if $\boldsymbol{R}_r \neq \boldsymbol{R}_s$ then $\|\boldsymbol{R}_r - \boldsymbol{R}_s\|^2 = 2$ and $\left\|\hat{\boldsymbol{P}}_r - \hat{\boldsymbol{P}}_s\right\|^2 \leq 4$ since they are coordinate and unit vectors respectively. This then implies,

$$\operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{L}\hat{\boldsymbol{P}}\right) \leq 2\operatorname{tr}\left(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R}\right)\,. \tag{4.10}$$

Thus we have

$$\begin{aligned}
\operatorname{tr}(\hat{\boldsymbol{P}}^\top \boldsymbol{M}\hat{\boldsymbol{P}})\mathcal{R}_{\boldsymbol{M}} &\leq 2\operatorname{tr}(\hat{\boldsymbol{P}}^\top \boldsymbol{L}\hat{\boldsymbol{P}})\mathcal{R}_{\boldsymbol{L}} + 2k && \text{by (4.8)} \\
&\leq 4\operatorname{tr}\left(\boldsymbol{R}^\top \boldsymbol{L}\boldsymbol{R}\right)\mathcal{R}_{\boldsymbol{L}} + 2k && \text{by (4.10)} \\
&\leq 2\operatorname{tr}\left(\boldsymbol{R}^\top \boldsymbol{M}\boldsymbol{R}\right)\mathcal{R}_{\boldsymbol{M}} + 2k && \text{by (4.7)}
\end{aligned}$$

By symmetry we have demonstrated the inequality for PDLaplacians. □

4.8.2 Proof of Proposition 58

In the following, we define $\mathcal{K}_{\boldsymbol{x}}(\cdot) := \mathcal{K}(\boldsymbol{x}, \cdot)$. If $r \geq 2$, $\delta^*_\infty := \min\left(2, \frac{1}{4}\delta(S_1, \ldots, S_k)\right)$. This implies that $\delta^*_\infty \leq \min\left(2, \frac{r-1}{2r}\delta(S_1, \ldots, S_k)\right)$. Recall that $s(\boldsymbol{x}) := \frac{r-1}{2r}\boldsymbol{x} + \frac{r+1}{2}\mathbf{1}$. Then observe that, given that the transformation $\tilde{\boldsymbol{x}}_i = s(\boldsymbol{x}_i)$ holds true for all $i \in [m]$, requiring $S_1, \ldots, S_k \subset [-r, r]^d$ with $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m \in \cup_{i=1}^k S_i$ and $\delta^*_\infty \leq \min\left(2, \frac{r-1}{2r}\delta(S_1, \ldots, S_k)\right)$ is equivalent to the requirement that $\tilde{S}_1, \ldots, \tilde{S}_k \subset [1, r]^d$

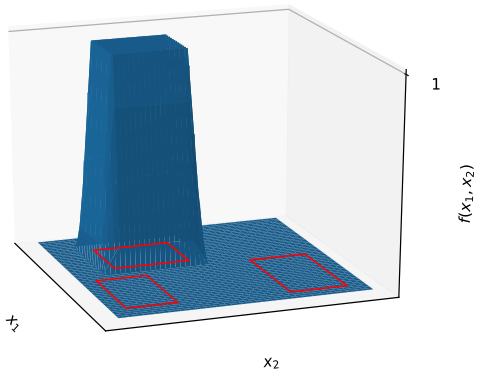


Figure 4.7: Visualization of the function $f(x_1, x_2)$ with S_1, S_2 and S_3 represented as red rectangles in the $x_1 - x_2$ plane.

with $\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_m \in \cup_{i=1}^{k} \tilde{S}_i$ and $\delta_\infty^* \leq \min\left(2, \delta(\tilde{S}_1, \ldots, \tilde{S}_k)\right)$. Furthermore, for all $i \in [m]$ and $j \in [k]$, we have that $\boldsymbol{x}_i \in S_j$ if and only if $\tilde{\boldsymbol{x}}_i \in \tilde{S}_j$. We shall proceed with the latter set of requirements for simplicity. Recall that the RKHS for the $d = 1$ min kernel $\mathcal{H}_{\mathcal{K}}^1$ is the set of all absolutely continuous functions from $[0, \infty)^d \to \mathfrak{R}$ that satisfy $f(0) = 0$ and $\int_0^\infty [f'(x)]^2 \mathrm{d}x < \infty$.

Lemma 64. *The inner product for $f \in \mathcal{H}_{\mathcal{K}}^1$ may be computed by,*

$$\langle f, g \rangle = \int_0^\infty f'(x) g'(x) \mathrm{d}x \,.$$

Proof. We show this by the reproducing property:

$$\langle f, \mathcal{K}_x \rangle = f(x).$$

Defining $\mathbf{1}_x(t)$ as the step function that evaluates to 1 for $t \leq x$ and 0 otherwise, we note that the derivative of $\min(x, t)$ with respect to t is equal to $\mathbf{1}_x(t)$. This gives rise to

$$\int_0^\infty f'(t) \mathcal{K}'(x, t) \mathrm{d}t = \int_0^\infty f'(t) \mathbf{1}_x(t) \mathrm{d}t = \int_0^x f'(t) \mathrm{d}t = f(x) \,.$$

Using the condition of $f(0) = 0$, we then obtain the reproducing property. □

Lemma 65. *Given k boxes $\tilde{S}_1, \ldots, \tilde{S}_k \subset [1, r]^d$, $\delta^*_\infty \leq \min\left(2, \delta(\tilde{S}_1, \ldots, \tilde{S}_k)\right)$ and $\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_m \in \cup_{i=1}^k \tilde{S}_i$, there exists a function $f \in H_\mathcal{K}$ for which $f(\tilde{\boldsymbol{x}}_j) = [\tilde{\boldsymbol{x}}_j \in \tilde{S}_1]$ for $j \in [m]$ and this function has norm*

$$\|f\|^2 = \left(\frac{4}{\delta^*_\infty}\right)^d.$$

Proof. Recall that a *box* in $\mathfrak{R}^d$ is a set $\{\boldsymbol{x} : a_i \leq x_i \leq b_i, i \in [d]\}$ defined by a pair of vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathfrak{R}^d$. First, we consider the case of $d = 1$, with the coordinates of $\tilde{S}_1$ defined by a and b. Defining the function that interpolates the points $\tilde{\boldsymbol{x}}_1, \ldots \tilde{\boldsymbol{x}}_m$ in one dimension as $f^1 \in \mathcal{H}^1_\mathcal{K}$, we chose f^1 to be the following:

$$f^1(x) = \begin{cases} 0 & \text{for} \quad x \leq a - \frac{\delta^*_\infty}{2} \\ \frac{2}{\delta^*_\infty} x + 1 - \frac{2}{\delta^*_\infty} a & \text{for} \quad a - \frac{\delta^*_\infty}{2} < x \leq a \\ 1 & \text{for} \quad a < x \leq b \\ -\frac{2}{\delta^*_\infty} x + 1 + \frac{2}{\delta^*_\infty} b & \text{for} \quad b < x \leq b + \frac{\delta^*_\infty}{2} \\ 0 & \text{for} \quad x > b + \frac{\delta^*_\infty}{2}. \end{cases}$$

This function is picked from the space $\mathcal{H}^1_\mathcal{K}$ so that $\int_0^\infty [(f^1)'(x)]^2 \mathrm{d}x$ is minimized with respect to "worst-case" constraints. The condition on δ^*_∞ implies that $\delta^*_\infty \leq 2$, so that $f^1(0) = 0$. It also implies that $\delta^*_\infty \leq \delta(S_1, \ldots S_k)$ so that for all $i \in [m]$, $f^1(\tilde{x}_i) = 0$ if $\tilde{x}_i \notin S_1$. The norm $\|f^1\|^2$, then becomes

$$\begin{aligned} \|f^1\|^2 &= \int_0^\infty |(f^1)'(x)|^2 \mathrm{d}x \\ &= \int_{a - \frac{\delta^*_\infty}{2}}^{a} \left(\frac{2}{\delta^*_\infty}\right)^2 \mathrm{d}x + \int_b^{b + \frac{\delta^*_\infty}{2}} \left(\frac{2}{\delta^*_\infty}\right)^2 \mathrm{d}x \\ &= 2\left(\frac{2}{\delta^*_\infty}\right)^2 \left(\frac{\delta^*_\infty}{2}\right) = \frac{4}{\delta^*_\infty}. \end{aligned}$$

This can be extended to multiple dimensions by observing that the induced product norm of f is the product of the norms of f^1 in each dimension, thus giving the required bound. In this case also, the condition on δ^*_∞ ensures both $f(\mathbf{0}) = 0$

and $f(\tilde{\boldsymbol{x}}_i) = 0$ for $\tilde{\boldsymbol{x}}_i \notin \tilde{S}_1$,where $i \in [m]$. For an illustration of this function in two dimensions, see Figure 4.7. □

Lemma 66. *Given k boxes $\tilde{S}_1, \ldots, \tilde{S}_k \subset [1, r]^d$, $\delta_\infty^* \leq \min\left(2, \delta(\tilde{S}_1, \ldots, \tilde{S}_k)\right)$ and $\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_m \in \cup_{i=1}^k \tilde{S}_i$, if $\boldsymbol{u} = (u_i = [\tilde{\boldsymbol{x}}_i \in \tilde{S}_1])_{i\in[m]}$ and $\boldsymbol{K} = (\mathcal{K}(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j))_{i,j\in[m]}$ then $\boldsymbol{u}^\top \boldsymbol{K}^{-1}\boldsymbol{u} \leq \left(\frac{4}{\delta_\infty^*}\right)^d$.*

Proof. Using a well-known equality (see e.g. [82, Proposition 12.32]) and Lemma 65, we observe that,

$$\boldsymbol{u}^\top \boldsymbol{K}^{-1}\boldsymbol{u} = \min_{f\in H_{\mathcal{K}}: f(\tilde{\boldsymbol{x}}_i)=[\tilde{\boldsymbol{x}}_i\in\tilde{S}_1], i\in[m]} \|f\|_{\mathcal{K}}^2 \leq \left(\frac{4}{\delta_\infty^*}\right)^d,$$

for $\boldsymbol{u} := (u_i = [\tilde{\boldsymbol{x}}_i \in \tilde{S}_1])_{i\in[m]}$, $\boldsymbol{K} := (\mathcal{K}(\tilde{\boldsymbol{x}}_i, \tilde{\boldsymbol{x}}_j))_{i,j\in[m]}$ and $\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_m \in \cup_{i\in[k]}\tilde{S}_i$. □

Defining $\boldsymbol{u}_i$ as the i^{th} column of $\boldsymbol{R}$, then observe that the term $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{K}^{-1}\boldsymbol{R}) = \sum_{i\in[k]} \boldsymbol{u}_i^\top \boldsymbol{K}^{-1}\boldsymbol{u}_i$. Thus by applying Lemma 66 to each $\boldsymbol{u}_i$, we have that $\mathrm{tr}(\boldsymbol{R}^\top \boldsymbol{K}^{-1}\boldsymbol{R}) \leq k(\frac{4}{\delta_\infty^*})^d$. ■

4.8.3 Proof of Proposition 60

We recall the proposition and first provide an intermediate lemma.

Proposition 60. For $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{m,n}$ with decomposition $\boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ where $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{U}^* \in \{-1, 1\}^{k\times\ell}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$, $k > 1$, $\ell > 1$, and for $\frac{1}{\gamma} = (h+1)^2 \min(\sqrt{k}, \sqrt{\ell})$, there exists a matrix $\bar{\boldsymbol{U}} \in \mathrm{SP}^1(\boldsymbol{U})$, such that $\|\bar{\boldsymbol{U}}\|_{\max} \leq \frac{1}{\gamma}$ and

$$\mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma}(\bar{\boldsymbol{U}}) \leq \frac{(h+1)^2}{1+\sqrt{3}}\left(k\,\mathcal{R}_{\boldsymbol{M}}\,\mathrm{tr}(\tilde{\boldsymbol{R}}^\top \boldsymbol{M}\tilde{\boldsymbol{R}}) + \ell\,\mathcal{R}_{\boldsymbol{N}}\,\mathrm{tr}(\tilde{\boldsymbol{C}}^\top \boldsymbol{N}\tilde{\boldsymbol{C}})\right),$$

for any $\tilde{\boldsymbol{R}}$ and $\tilde{\boldsymbol{C}}$ where

$$\tilde{\boldsymbol{R}}_{ij} \in \begin{cases} [1+\sqrt{3}, h] & \text{if } R_{ij} = 1 \\ [0, \frac{1}{k-1}] & \text{if } R_{ij} = 0 \end{cases} \quad \text{and} \quad \tilde{\boldsymbol{C}}_{ij} \in \begin{cases} [1+\sqrt{3}, h] & \text{if } C_{ij} = 1 \\ [0, \frac{1}{\ell-1}] & \text{if } C_{ij} = 0 \end{cases}.$$

Lemma 67. *For any $\boldsymbol{U} \in \mathbb{B}_{k,\ell}^{m,n}$ with decomposition $\boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top$ where $\boldsymbol{U}^* \in \{-1, 1\}^{k\times\ell}$, $\boldsymbol{R} \in \mathcal{B}^{m,k}$, $\boldsymbol{C} \in \mathcal{B}^{n,\ell}$, $h > 1 + \sqrt{3}$, and any $\tilde{\boldsymbol{R}}$ and $\tilde{\boldsymbol{C}}$ where $\tilde{\boldsymbol{R}}_{ij} \in$*

$$\begin{cases} [1+\sqrt{3}, h] & \text{if } R_{ij}=1 \\ [0, \frac{1}{k-1}] & \text{if } R_{ij}=0 \end{cases} \quad \text{and} \quad \tilde{\boldsymbol{C}}_{ij} \in \begin{cases} [1+\sqrt{3}, h] & \text{if } C_{ij}=1 \\ [0, \frac{1}{\ell-1}] & \text{if } R_{ij}=0 \end{cases},$$

$$\tilde{\boldsymbol{R}}\boldsymbol{U}^*\tilde{\boldsymbol{C}}^\top \in \mathrm{SP}^1(\boldsymbol{U}).$$

Proof. We first write $(\tilde{\boldsymbol{R}}\boldsymbol{U}^*\tilde{\boldsymbol{C}}^\top)_{ij} = \tilde{\boldsymbol{R}}_i^\top \boldsymbol{U}^* \tilde{\boldsymbol{C}}_j$. We proceed to evaluate the value of $(\tilde{\boldsymbol{R}}\boldsymbol{U}^*\tilde{\boldsymbol{C}}^\top)_{ij}$ for arbitrary i and j. We denote $(\boldsymbol{v}^i)^\top := \tilde{\boldsymbol{R}}_i^\top \boldsymbol{U}^*$. We also denote the indices of the largest elements of $\tilde{\boldsymbol{R}}_i$ and $\tilde{\boldsymbol{C}}_j$ as κ_{max} and l_{max} respectively, and define $a_i := \tilde{\boldsymbol{R}}_{i\kappa_{max}}$. By the restrictions on $\tilde{\boldsymbol{R}}_{i\kappa}$ for any $\kappa \in [k]$ and $\boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}$, we have that $a_i = \tilde{R}_{i\kappa_{max}} \geq 1+\sqrt{3}$ and $|\sum_{\kappa'\in[k]\setminus\{\kappa_{max}\}} \tilde{\boldsymbol{R}}_{i\kappa'}\boldsymbol{U}^*_{\kappa' l}| \leq 1$ for any $l \in [\ell]$. Hence, for any $l \in [\ell]$

$$\boldsymbol{v}^i_l \in [-(a_i+1), -(a_i-1)] \cup [a_i-1, a_i+1]$$

and

$$|\tilde{R}_{i\kappa_{max}}\boldsymbol{U}^*_{\kappa_{max}l}| = \tilde{R}_{i\kappa_{max}} > \left|\sum_{\kappa'\in[k]\setminus\{\kappa_{max}\}} \tilde{\boldsymbol{R}}_{i\kappa'}\boldsymbol{U}^*_{\kappa' l}\right|. \tag{4.11}$$

Since $\mathrm{sign}(\tilde{R}_{i\kappa_{max}}\boldsymbol{U}^*_{\kappa_{max}l}) = \mathrm{sign}(\boldsymbol{U}^*_{\kappa_{max}l})$ and $\boldsymbol{U}^*_{\kappa_{max}} = \tilde{\boldsymbol{R}}_i^\top \boldsymbol{U}^*$, Equation (4.11) implies

$$(\boldsymbol{v}^i)^\top = \tilde{\boldsymbol{R}}_i^\top \boldsymbol{U}^* \in \mathrm{SP}(\boldsymbol{R}_i^\top \boldsymbol{U}^*). \tag{4.12}$$

We now show that $\mathrm{sign}(\tilde{\boldsymbol{R}}_i^\top \boldsymbol{U}^* \tilde{\boldsymbol{C}}_j) = \mathrm{sign}(U_{ij})$. By Equation (4.12) and the fact that $U_{ij} = \boldsymbol{R}_i^\top \boldsymbol{U}^*_{l_{max}}$, this is equivalent to showing that the sign of $\tilde{\boldsymbol{R}}_i^\top \boldsymbol{U}^* \tilde{\boldsymbol{C}}_j = (\boldsymbol{v}^i)^\top \tilde{\boldsymbol{C}}_j$ is equal to the sign of $v^i_{l_{max}}$. We will show this by proving that $|v^i_{l_{max}} \tilde{C}_{jl_{max}}| > |\sum_{l'\in[\ell]\setminus\{l_{max}\}} v^i_{l'} \tilde{C}_{jl'}|$.

In the worst case, $|v^i_{l_{max}}| = a_i - 1$ and for all $l' \in [\ell] \setminus \{l_{max}\}$, $|v^i_{l'}| = a_i + 1$, $v^i_{l'} v^i_{l_{max}} < 0$, $\tilde{\boldsymbol{C}}_{jl_{max}} = 1+\sqrt{3}$ and $\tilde{\boldsymbol{C}}_{jl'} = \frac{1}{\ell-1}$. Recall that we require that $|v^i_{l_{max}} \tilde{C}_{jl_{max}}| = (1+\sqrt{3})(a_i-1) > |\sum_{l'\in[\ell]\setminus\{l_{max}\}} v^i_{l'} \tilde{C}_{jl'}| = a_i + 1$. This holds for any $a_i > 1 + \frac{2}{3}\sqrt{3}$, and holds in our case since $a_i \geq 1+\sqrt{3} > 1+\frac{2}{3}\sqrt{3}$. We have now shown $\tilde{\boldsymbol{R}}\boldsymbol{U}^*\tilde{\boldsymbol{C}}^\top \in$

SP($\boldsymbol{U}$). To conclude the proof, we show $|(\boldsymbol{v}^i)^\top \tilde{\boldsymbol{C}}_j| \geq 1$. We have

$$\begin{aligned}
|(\boldsymbol{v}^i)^\top \tilde{\boldsymbol{C}}_j| &\geq |v^i_{l_{max}} \tilde{C}_{jl_{max}} - \sum_{l' \in [\ell] \setminus \{l_{max}\}} v^i_{l'} \tilde{C}_{jl'}| \\
&\geq (1 + \sqrt{3})(a_i - 1) - (a_i + 1) \\
&\geq \sqrt{3} a_i - 2 - \sqrt{3} \\
&\geq \sqrt{3}(1 + \sqrt{3}) - 2 - \sqrt{3} \\
&= 1.
\end{aligned}$$

□

For a matrix $\boldsymbol{A} \in \Re^{a \times b}$, we define the diagonal matrix $\boldsymbol{D_A}$ with entries $(\boldsymbol{D_A})_{ii} = \frac{1}{\|A_i\|}$. The proposition can be proven by combining the bounds derived from setting the following values of γ and real-valued comparator matrix $\bar{\boldsymbol{U}}$

$$\gamma = \frac{1}{(h+1)^2 \sqrt{\ell}} \qquad \bar{\boldsymbol{U}} = \frac{1}{\gamma} \boldsymbol{D}_{\tilde{\boldsymbol{R}}\boldsymbol{U}^*} \tilde{\boldsymbol{R}} \boldsymbol{U}^* (\tilde{\boldsymbol{C}} \boldsymbol{D}_{\tilde{\boldsymbol{C}}})^\top \tag{4.13}$$

and

$$\gamma = \frac{1}{(h+1)^2 \sqrt{k}} \qquad \bar{\boldsymbol{U}} = \frac{1}{\gamma} \boldsymbol{D}_{\tilde{\boldsymbol{R}}} \tilde{\boldsymbol{R}} (\tilde{\boldsymbol{C}} (\boldsymbol{U}^*)^\top \boldsymbol{D}_{\boldsymbol{U}^* \tilde{\boldsymbol{C}}^\top})^\top.$$

For conciseness, we proceed to prove using the first set of values as set out in (4.13), noting that the bound for the second set of values follows analogously.

By the restrictions on the elements of $\tilde{\boldsymbol{R}}$ and $\boldsymbol{U}^* \in \{-1, 1\}^{k \times \ell}$, we have that for all $i \in [m]$ and for all $l \in [\ell]$, $\max_{\kappa \in [k]} \tilde{\boldsymbol{R}}_{i\kappa} \in [1 + \sqrt{3}, h]$ and $|\sum_{\kappa' \in [k] \setminus \{\kappa_{max}\}} \tilde{\boldsymbol{R}}_{i\kappa'} \boldsymbol{U}^*_{\kappa' l}| \leq 1$. Therefore, we have

$$\tilde{\boldsymbol{R}}_i \boldsymbol{U}^* \in [\sqrt{3}, h+1]^\ell, \tag{4.14}$$

giving

$$\max_{i \in [m]} \|(\tilde{\boldsymbol{R}} \boldsymbol{U}^*)_i\| \leq (h+1) \sqrt{\ell}. \tag{4.15}$$

We also have

$$\max_{j \in [n]} \|\tilde{\boldsymbol{C}}_j\| \leq (h+1) \tag{4.16}$$

since in the worst case $\tilde{\boldsymbol{C}}_j$ has one element as h and all the other elements $\frac{1}{\ell-1}$, so that $\max_{j\in[n]} \left\|\tilde{\boldsymbol{C}}_j\right\| \leq \sqrt{h^2+1} \leq h+1$. By Lemma 67, $\tilde{\boldsymbol{R}}\boldsymbol{U}^*\tilde{\boldsymbol{C}}^\top \in \mathrm{SP}^1(\boldsymbol{U})$, and by Equations (4.15) and (4.16), we have that for all $i \in [m]$ and $j \in [n]$, $\frac{(h+1)\sqrt{\ell}}{\|(\tilde{\boldsymbol{R}}\boldsymbol{U}^*)_i\|} \geq 1$ and $\frac{h+1}{\|\tilde{\boldsymbol{C}}_j\|} \geq 1$, so that the γ term and the two $\boldsymbol{D}$ matrices only scale the values of $\bar{\boldsymbol{U}}$ to be larger. Hence $\bar{\boldsymbol{U}} \in \mathrm{SP}^1(\boldsymbol{U})$.

We now show that $\|\bar{\boldsymbol{U}}\|_{\max} \leq \frac{1}{\gamma} = (h+1)^2\sqrt{\ell}$. Recall the definition of the max-norm:

$$\|\bar{\boldsymbol{U}}\|_{\max} = \min_{\boldsymbol{P}\boldsymbol{Q}^\top=\bar{\boldsymbol{U}}} \max_i \|\boldsymbol{P}_i\| \max_j \|\boldsymbol{Q}_j\|.$$

Defining $\boldsymbol{P}' := (h+1)\sqrt{\ell}\,\boldsymbol{D}_{\tilde{\boldsymbol{R}}\boldsymbol{U}^*}\tilde{\boldsymbol{R}}\boldsymbol{U}^*$ and $\boldsymbol{Q}' := (h+1)\boldsymbol{D}_{\tilde{\boldsymbol{C}}}\tilde{\boldsymbol{C}}$, we then have that $(\boldsymbol{P}', \boldsymbol{Q}')$ is in the feasible set of the optimization, so that $\|\bar{\boldsymbol{U}}\|_{\max} \leq \max_i \|\boldsymbol{P}'_i\| \max_j \|\boldsymbol{Q}'_j\|$. Since $\max_i \|(\boldsymbol{D}_{\boldsymbol{A}}\boldsymbol{A})_i\| = 1$ for an arbitrary matrix $\boldsymbol{A}$, we then have

$$\|\bar{\boldsymbol{U}}\|_{\max} \leq (h+1)^2\sqrt{\ell}.$$

We now prove the quasi-dimension bound. To do so, we will make use of the following facts:

- $$\min_{i\in[m]} \|(\tilde{\boldsymbol{R}}\boldsymbol{U}^*)_i\| \geq \sqrt{3\ell}, \tag{4.17}$$

  which holds due to (4.14).

- $$\min_{j\in[n]} \|\tilde{\boldsymbol{C}}_j\| \geq 1 + \sqrt{3}, \tag{4.18}$$

  since in the worst case $\tilde{\boldsymbol{C}}_j$ has one non-zero element with value $1+\sqrt{3}$.

- $$\mathrm{tr}((\boldsymbol{D}_{\boldsymbol{A}}\boldsymbol{A})^\top \boldsymbol{B}\boldsymbol{D}_{\boldsymbol{A}}\boldsymbol{A}) \leq \frac{1}{\min_{i\in[a]} \|\boldsymbol{A}_i\|^2} \sum_{i\in[b]} \boldsymbol{a}_i^\top \boldsymbol{B}\boldsymbol{a}_i, \tag{4.19}$$

  where $\boldsymbol{a}_i$ is the i^{th} column of $\boldsymbol{A}$, $\boldsymbol{A} \in \Re^{a\times b}$ and $\boldsymbol{B} \in \Re^{b\times b}$. The inequality holds since, $\mathrm{tr}((\boldsymbol{D}_{\boldsymbol{A}}\boldsymbol{A})^\top \boldsymbol{B}\boldsymbol{D}_{\boldsymbol{A}}\boldsymbol{A}) = \sum_{i\in[b]} \tilde{\boldsymbol{a}}_i \boldsymbol{B}\tilde{\boldsymbol{a}}_i^\top$, where $\tilde{\boldsymbol{a}}_i$ is the i^{th} column of $\boldsymbol{D}_{\boldsymbol{A}}\boldsymbol{A}$, and can be evaluated as $\tilde{\boldsymbol{a}}_i = \boldsymbol{a}_i \odot (1/\|\boldsymbol{A}_1\|, \ldots, 1/\|\boldsymbol{A}_a\|) \leq \frac{1}{\min_{i\in[a]} \|\boldsymbol{A}_i\|}\boldsymbol{a}_i$.

We define $\hat{\boldsymbol{P}}' = \boldsymbol{D}_{\tilde{\boldsymbol{R}}\boldsymbol{U}^*}\tilde{\boldsymbol{R}}\boldsymbol{U}^*$ and $\hat{\boldsymbol{Q}}' = \boldsymbol{D}_{\tilde{\boldsymbol{C}}}\tilde{\boldsymbol{C}}$ and note that $\|\hat{\boldsymbol{P}}'_i\| = 1$ for

all $i \in [m]$ and $\|\hat{\boldsymbol{Q}}'_j\| = 1$ for all $j \in [n]$. Then recall the definition of the quasi-dimension:

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma \boldsymbol{U}} \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M} \hat{\boldsymbol{P}}\right) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N} \hat{\boldsymbol{Q}}\right),$$

where $\hat{\boldsymbol{P}}$ and $\hat{\boldsymbol{Q}}$ are row-normalized matrices. We observe that $\hat{\boldsymbol{P}}'(\hat{\boldsymbol{Q}}')^\top = \gamma \bar{\boldsymbol{U}}$, and hence we can bound

$$\begin{aligned}
\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\bar{\boldsymbol{U}}) &\le \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}((\hat{\boldsymbol{P}}')^\top \boldsymbol{M} \hat{\boldsymbol{P}}') + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left((\hat{\boldsymbol{Q}}')^\top \boldsymbol{N} \hat{\boldsymbol{Q}}'\right) \\
&= \mathcal{R}_{\boldsymbol{M}} \operatorname{tr}((\boldsymbol{D}_{\tilde{\boldsymbol{R}}\boldsymbol{U}^*} \tilde{\boldsymbol{R}} \boldsymbol{U}^*)^\top \boldsymbol{M} \boldsymbol{D}_{\tilde{\boldsymbol{R}}\boldsymbol{U}^*} \tilde{\boldsymbol{R}} \boldsymbol{U}^*) + \mathcal{R}_{\boldsymbol{N}} \operatorname{tr}\left((\boldsymbol{D}_{\tilde{\boldsymbol{C}}} \tilde{\boldsymbol{C}})^\top \boldsymbol{N} \boldsymbol{D}_{\tilde{\boldsymbol{C}}} \tilde{\boldsymbol{C}}\right) \\
&\le \frac{\mathcal{R}_{\boldsymbol{M}}}{\min_{i\in[m]} \|(\tilde{\boldsymbol{R}}\boldsymbol{U}^*)_i\|^2} \operatorname{tr}\left((\tilde{\boldsymbol{R}}\boldsymbol{U}^*)^\top \boldsymbol{M} \tilde{\boldsymbol{R}} \boldsymbol{U}^*\right) + \frac{\mathcal{R}_{\boldsymbol{N}}}{\min_{j\in[n]} \|\tilde{\boldsymbol{C}}_j\|^2} \operatorname{tr}\left(\tilde{\boldsymbol{C}}_j^\top \boldsymbol{N} \tilde{\boldsymbol{C}}_j\right) \qquad (4.20) \\
&\le \frac{\mathcal{R}_{\boldsymbol{M}}}{3\ell} \operatorname{tr}((\tilde{\boldsymbol{R}}\boldsymbol{U}^*)^\top \boldsymbol{M} \tilde{\boldsymbol{R}} \boldsymbol{U}^*) + \frac{\mathcal{R}_{\boldsymbol{N}}}{(1+\sqrt{3})^2} \operatorname{tr}\left(\tilde{\boldsymbol{C}}^\top \boldsymbol{N} \tilde{\boldsymbol{C}}\right) \qquad (4.21) \\
&= \frac{\mathcal{R}_{\boldsymbol{M}}}{3\ell} \operatorname{tr}((\boldsymbol{U}^*)^\top \tilde{\boldsymbol{R}}^\top \boldsymbol{M} \tilde{\boldsymbol{R}} \boldsymbol{U}^*) + \frac{\mathcal{R}_{\boldsymbol{N}}}{(1+\sqrt{3})^2} \operatorname{tr}(\tilde{\boldsymbol{C}}^\top \boldsymbol{N} \tilde{\boldsymbol{C}}) \\
&= \frac{\mathcal{R}_{\boldsymbol{M}}}{3\ell} \operatorname{tr}(\boldsymbol{U}^* (\boldsymbol{U}^*)^\top \tilde{\boldsymbol{R}}^\top \boldsymbol{M} \tilde{\boldsymbol{R}}) + \frac{\mathcal{R}_{\boldsymbol{N}}}{(1+\sqrt{3})^2} \operatorname{tr}(\tilde{\boldsymbol{C}}^\top \boldsymbol{N} \tilde{\boldsymbol{C}}) \\
&\le \frac{\mathcal{R}_{\boldsymbol{M}}}{3\ell} \operatorname{tr}(\boldsymbol{U}^* (\boldsymbol{U}^*)^\top) \operatorname{tr}(\tilde{\boldsymbol{R}}^\top \boldsymbol{M} \tilde{\boldsymbol{R}}) + \frac{\mathcal{R}_{\boldsymbol{N}}}{(1+\sqrt{3})^2} \operatorname{tr}(\tilde{\boldsymbol{C}}^\top \boldsymbol{N} \tilde{\boldsymbol{C}}) \qquad (4.22) \\
&= \frac{\mathcal{R}_{\boldsymbol{M}}}{3} k \operatorname{tr}(\tilde{\boldsymbol{R}}^\top \boldsymbol{M} \tilde{\boldsymbol{R}}) + \frac{\mathcal{R}_{\boldsymbol{N}}}{(1+\sqrt{3})^2} \operatorname{tr}(\tilde{\boldsymbol{C}}^\top \boldsymbol{N} \tilde{\boldsymbol{C}}) \qquad (4.23)
\end{aligned}$$

where (4.20) follows from (4.19), (4.21) follows from (4.17) and (4.18), (4.22) follows from $\operatorname{tr}(\boldsymbol{AB}) \le \lambda_{\max}(\boldsymbol{A}) \operatorname{tr}(\boldsymbol{B}) \le \operatorname{tr}(\boldsymbol{A}) \operatorname{tr}(\boldsymbol{B})$ for $\boldsymbol{A}, \boldsymbol{B} \in \boldsymbol{S}_+$ and (4.23) follows from $\operatorname{tr}(\boldsymbol{U}^*(\boldsymbol{U}^*)^\top) = k\ell$.

■

4.8.4 Proof of Proposition 61

We first prove an intermediate lemma.

Lemma 68. *Given k balls $S_1, \dots, S_k$, with respective radii $\rho_1, \dots, \rho_k$, maximal radius $\rho = \max_{i\in[k]} \rho_i$, $\delta_2^* = \delta(S_1, \dots, S_k)$, $\boldsymbol{x}_1, \dots, \boldsymbol{x}_m \in \cup_{i=1}^k S_i$, and $\boldsymbol{K} = (\mathcal{K}_\beta(\boldsymbol{x}_i, \boldsymbol{x}_j))_{i,j\in[m]}$, where $\beta = \frac{1}{(2\rho\delta_2^* + (\delta_2^*)^2)d} \log((1+\sqrt{3})(k-1))$, there exists a $\boldsymbol{u} \in \mathfrak{R}^m$*

such that

$$u_i \in \begin{cases} [1+\sqrt{3}, h] & \text{if } \boldsymbol{x}_i \in S_1 \\ [0, \frac{1}{k-1}] & \text{otherwise} \end{cases}$$

and $\boldsymbol{u}^\top \boldsymbol{K}^{-1}\boldsymbol{u} \leq h^2$,*where* $h := (1+\sqrt{3})((1+\sqrt{3})(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}}$.

Proof. We make use of the following equality, (see e.g. [82, Proposition 12.32])

$$\boldsymbol{u}^\top \boldsymbol{K}^{-1}\boldsymbol{u} = \min_{f \in H_{\mathcal{K}}: f(\boldsymbol{x}_i) = u_i, i \in [m]} \|f\|_{\mathcal{K}}^2$$

Taking $f'(\cdot) = h\mathcal{K}_\beta(\cdot, \boldsymbol{t})$, where $\boldsymbol{t} \in \mathfrak{R}^d$ is the center of S_1, and $\beta = \frac{1}{(2\rho\delta_2^* + (\delta_2^*)^2)d}\log((1+\sqrt{3})(k-1))$, the lemma is proven if f' is in the feasible set, since we would then have

$$\min_{f \in H_{\mathcal{K}}: f(\boldsymbol{x}_i) = u_i, i \in [m]} \|f\|_{\mathcal{K}}^2 \leq \|f'\|_{\mathcal{K}}^2 \leq h^2.$$

We proceed to prove that f' is in the feasible set. To do so, we prove that the following statements are true: $f'(\boldsymbol{t} + \rho\mathbf{1}) = 1+\sqrt{3}$ and $f'(\boldsymbol{t} + (\rho + \delta_2^*)\mathbf{1}) = \frac{1}{k-1}$. These constraints ensure that the function $f'(\boldsymbol{x})$ must be sufficiently large for any point $\boldsymbol{x}$ within the radius ρ (corresponding to the case when $\boldsymbol{x} \in S_1$), and that $f'(\boldsymbol{x})$ decays sufficiently for any point $\boldsymbol{x}$ beyond the given separation δ_2^* (when $\boldsymbol{x} \notin S_1$).

$$\begin{aligned} f'(\boldsymbol{t} + \rho\mathbf{1}) &= h\mathcal{K}_\beta(\boldsymbol{t} + \rho\mathbf{1}, \boldsymbol{t}) \\ &= h\exp(-\beta\rho^2 d) \\ &= (1+\sqrt{3})((1+\sqrt{3})(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}} \exp\left(-\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}\log((1+\sqrt{3})(k-1))\right) \\ &= (1+\sqrt{3})((1+\sqrt{3})(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}}((1+\sqrt{3})(k-1))^{\frac{-\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}} \\ &= 1+\sqrt{3}. \end{aligned}$$

$$
\begin{aligned}
f'(\boldsymbol{t} + (\rho + \delta_2^*)\mathbf{1}) &= h\mathcal{K}_\beta(\boldsymbol{t} + (\rho + \delta_2^*)\mathbf{1}, \boldsymbol{t}) \\
&= h \exp(-\beta(\rho + \delta_2^*)^2 d) \\
&= (1 + \sqrt{3})((1 + \sqrt{3})(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}} \exp\left(-\frac{(\rho + \delta_2^*)^2}{2\rho\delta_2^* + (\delta_2^*)^2} \log((1 + \sqrt{3})(k-1))\right) \\
&= (1 + \sqrt{3})((1 + \sqrt{3})(k-1))^{\frac{\rho^2}{2\rho\delta_2^* + (\delta_2^*)^2}} ((1 + \sqrt{3})(k-1))^{\frac{-(\rho+\delta_2^*)^2}{2\rho\delta_2^* + (\delta_2^*)^2}} \\
&= (1 + \sqrt{3})((1 + \sqrt{3})(k-1))^{\frac{-2\rho\delta_2^* - (\delta_2^*)^2}{2\rho\delta_2^* + (\delta_2^*)^2}} \\
&= (1 + \sqrt{3})((1 + \sqrt{3})(k-1))^{-1} \\
&= \frac{1}{k-1}\,.
\end{aligned}
$$

□

Defining $\boldsymbol{u}_i$ as the i^{th} column of $\tilde{\boldsymbol{R}}$, we have $\tilde{\boldsymbol{R}}^\top \boldsymbol{K}^{-1} \tilde{\boldsymbol{R}} = \sum_{i=1}^k \boldsymbol{u}_i^\top \boldsymbol{K}^{-1} \boldsymbol{u}_i$. We then obtain the result by applying the intermediate lemma k times.

■

Chapter 5

Online Multitask Learning with Long-Term Memory

In this chapter, we introduce a novel online multitask setting. In this setting, each task is partitioned into a sequence of segments that is unknown to the learner. Associated with each segment is a hypothesis from some hypothesis class. We give an algorithm that is designed to exploit the scenario where there are many such segments but significantly fewer associated hypotheses. This algorithm is designed for infinite hypothesis classes from a reproducing kernel Hilbert space. Its per trial time complexity is cubic in the number of cumulative trials. We prove a regret bound that holds for any segmentation of the tasks and any association of hypotheses to the segments. In the single-task setting, this is equivalent to *switching with long-term memory* in the sense of [83]. In the single-task special case, this is the first example of an efficient regret-bounded switching algorithm with long-term memory for a non-parametric hypothesis class.

5.1 Introduction

We consider a model of online prediction in a non-stationary environment with multiple interrelated tasks. Associated with each task is an asynchronous data stream. As an example, consider a scenario where a team of drones may need to decontaminate an area of toxic waste. In this example, the tasks correspond to drones.

Each drone is receiving a data stream from its sensors. The data streams are non-stationary but interdependent as the drones are travelling within a common site. At any point in time, a drone receives an instance x and is required to predict its label y. The aim is to minimize mispredictions. As is standard in regret-bounded learning we have no statistical assumptions on the data-generation process. Instead, we aim to predict well relative to some hypothesis class of predictors. Unlike a standard regret model, where we aim to predict well in comparison to a single hypothesis, we instead aim to predict well relative to a completely unknown sequence of hypotheses in each task's data stream, as illustrated by the "*coloring*" in Figure 5.1. Each *mode* (color) corresponds to a distinct hypothesis from the hypothesis class. A *switch* is said to have occurred whenever we move between modes temporally within the same task. Thus in task 1, there are three modes and four switches. We are particularly motivated by the case that a mode once present will possibly recur multiple times even within different tasks, i.e., "modes" $\ll$ "switches." We will give algorithms and regret bounds for infinite non-parametric Reproducing Kernel Hilbert Space (RKHS) [84] hypothesis classes.

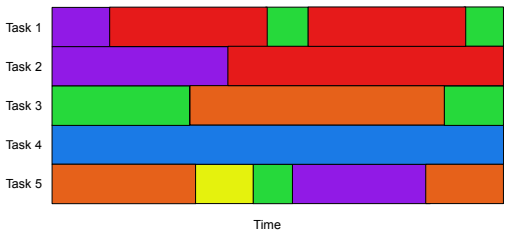


Figure 5.1: A Coloring of Data Streams (5 tasks, 6 modes, and 11 switches).

For $\tau = 1$ to T do
  Receive task $\ell^\tau \in [s]$.
  Set $i \leftarrow \ell^\tau$; $t \leftarrow \sigma(\tau)$.
  Receive instance $x^\tau \equiv x_t^i \in \mathcal{X}$.
  Predict $\hat{y}^\tau \equiv \hat{y}_t^i \in \{-1, 1\}$.
  Receive label $y^\tau \equiv y_t^i \in \{-1, 1\}$.
  Incur Loss $\mathcal{L}_{01}(y^\tau, \hat{y}^\tau)$.

Figure 5.2: The Switching Multitask Model

The chapter is organized as follows. In the next section, we introduce our formal model for online switching multitask learning. In doing so, we provide a brief review of some related online learning results which enable us to provide a prospectus for attainable regret bounds. This is done by considering the bounds achievable by non-polynomial time algorithms. We then provide a brief survey of related work. In Section 5.3, we provide the algorithm and the bound for RKHS

hypotheses classes. We complement our theoretical results with synthetic experiments in Section 5.4. Finally, we provide a few concluding remarks in Section 5.5 and Section 5.6 contains the proofs.

5.2 Online Learning with Switching, Memory, and Multiple Tasks

We review the models and regret bounds for online learning in the single-task, switching, and switching with memory models as background for our multitask switching model with memory.

In the single-task online model a *learner* receives data sequentially so that on a trial $t = 1, \ldots, T$, the learner:

1. receives an instance $x_t \in \mathcal{X}$ from the *environment*,
2. predicts a label $\hat{y}_t \in \{-1, 1\}$
3. receives a label from an environment $y_t \in \{-1, 1\}$
4. incurs a *zero-one* loss $\mathcal{L}_{01}(y_t, \hat{y}_t)$, where we recall that $\mathcal{L}_{01}(y_t, \hat{y}_t) := [y_t \neq \hat{y}_t]$.

There are no probabilistic assumptions on how the environment generates its instances or their labels; it is an arbitrary process which in fact may be adversarial. The only restriction on the environment is that it does not "see" the learner's $\hat{y}_t$ until after it reveals y_t. The learner's aim will be to compete with a hypothesis class of predictors $\mathcal{H} \subseteq \mathfrak{R}^{\mathcal{X}}$ so as to minimize its *expected c-regret*, $R_T^c(h) := \sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - c\mathcal{L}_{01}(y_t, \text{sign}(h(x_t)))$ for every hypothesis $h \in \mathcal{H}$, where the expectation is with respect to the learner's internal randomization.

In this chapter we will consider the hypothesis class given by a set $\mathcal{H}_K$ induced by a kernel K. Given a reproducing kernel $K : \mathcal{X} \times \mathcal{X} \to \mathfrak{R}$ we denote the induced norm of the reproducing kernel Hilbert space (RKHS) $\mathcal{H}_K$ as $\|\cdot\|_K$ (for details on RKHS see [84]). Given an instance sequence $\boldsymbol{x} := (x_1, \ldots, x_T)$, we let $\mathcal{H}_K^{(\boldsymbol{x},\lambda)} := \{h \in \mathcal{H}_K : h(x_t) \in ([-\lambda, -1] \cup [1, \lambda]), \forall t \in [T]\}$ denote the functions in $\mathcal{H}_K$ that have a bounded range on the sequence. In the case where $\lambda = 1$, the functions are

binary-valued. An analysis of online gradient descent (OGD_K) with the hinge loss, kernel K and randomized prediction [12, see e.g., Ch. 2 & 3] (proof included in Section 5.6.3 for completeness) gives an expected regret bound of

$$R_T^{\frac{1+\lambda}{2}}(h) \in O\left(\sqrt{\|h\|_K^2 X_K^2 T}\right) \quad (\forall h \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)}), \tag{5.1}$$

where $X_K^2 \geq \max_{t\in[T]} K(x_t, x_t)$. This bound does not have a dependence on λ, unlike the bound to be introduced in Theorem 69. It remains an open problem whether the dependence on λ in the latter bound can be removed.

In the switching single-task model the hypothesis becomes a sequence of hypotheses $\boldsymbol{h} = (h_1, h_2, \ldots, h_T) \in \mathcal{H}^T$ and the c-regret is $R_T^c(\boldsymbol{h}) := \sum_{t=1}^T \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - c\mathcal{L}_{01}(y_t, \mathrm{sign}(h_t(x_t)))$. Two parameters of interest are the number of *switches* $k := \sum_{t=1}^{T-1}[h_t \neq h_{t+1}]$ and the number of *modes* $m := |\cup_{t=1}^T \{h_t\}|$, i.e., the number of the distinct hypotheses in the sequence. In this chapter we are interested in *long-term memory*, that is, algorithms and bounds that are designed to exploit the case of $m \ll k$.

For the hypothesis class $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$, we may give non-memory bounds of the form $R_T^{\frac{1+\lambda}{2}}(\boldsymbol{h}) \in O\left(\sqrt{k \max_t \|h_t\|_K^2 X_K^2 T}\right)$ by using a simple modification [85] of OGD_K (see Section 5.6.3). To the best of our knowledge, there are no previous long-term memory bounds for $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$, even if $\lambda = 1$ (however see the discussion of [39] in Section 5.2.2); these will be a special case of our multitask model, to be introduced next.

5.2.1 Switching Multitask Model

In Figure 5.2, we illustrate the protocol for our multitask model. The model is essentially the same as the switching single-task model, except that we now have s tasks. On each (global) trial τ, the environment reveals the active task $\ell^\tau \in [s]$. The ordering of tasks chosen by the environment is arbitrary, and therefore we may switch tasks on every (global) trial τ. We use the following notational convention: (global time) $\tau \equiv {}_t^i$ (local time) where $i = \ell^\tau$, $t = \boldsymbol{\sigma}(\tau)$ and $\boldsymbol{\sigma}(\tau) := \sum_{j=1}^\tau [\ell^j = \ell^\tau]$. Thus $x^\tau \equiv x_t^i$, $y^\tau \equiv y_t^i$, etc., where the mapping is determined implicitly by the task

vector $\boldsymbol{\ell} \in [s]^T$. Each task $i \in [s]$ has its own data pair (instance, label) sequence $(x_1^i, y_1^i), \ldots, (x_{T^i}^i, y_{T^i}^i)$, where $T = T^1 + \ldots + T^s$. The *multitask hypotheses multiset* is denoted as $\boldsymbol{h}^* = (h^1, \ldots, h^T) \equiv (h_1^1, \ldots, h_{T^1}^1, \ldots, h_1^s, \ldots, h_{T^s}^s) \in \mathcal{H}^T$. In the multitask model, we denote the number of switches as $k(\boldsymbol{h}^*) := \sum_{i=1}^{s} \sum_{t=1}^{T^i-1} [h_t^i \neq h_{t+1}^i]$, the set of modes as $m(\boldsymbol{h}^*) := \cup_{i=1}^{s} \cup_{t=1}^{T^i} \{h_t^i\}$ and the multitask regret as $R_T(\boldsymbol{h}^*) := \sum_{i=1}^{s} \sum_{t=1}^{T^i} \mathbb{E}[\mathcal{L}_{01}(y_t^i, \hat{y}_t^i)] - \mathcal{L}_{01}(y_t^i, h_t^i(x_t^i))$. In the following, we give motivating upper bounds based on exponential-time algorithms induced by "meta-experts." We provide a lower bound with respect to $\mathcal{H}_K^{(\boldsymbol{x},1)}$ in Proposition 70.

The idea of "meta-experts" is to take the base class of hypotheses ($\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$) and to construct a class of "meta-hypotheses" by combining the original hypotheses to form new ones, and then apply a meta-algorithm to the constructed class; in other words, we reduce the "meta-model" to the "base-model." The meta-algorithm that we will use is the "multiplicative weight" (MW) algorithm [86] which learns a finite hypothesis class $\mathcal{H}_{\text{fin}}$, and a regret bound[1] of the form

$$R_T(h) \in O\Big(\sqrt{\log(|\mathcal{H}_{\text{fin}}|)T}\Big) \quad (\forall h \in \mathcal{H}_{\text{fin}})$$

was given in [87]. This is a special case of the framework of "prediction with expert advice" introduced in [88, 19]. We cannot however use hypotheses from $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ directly since the cardinality, in general, is infinite, and additionally we do not know $\boldsymbol{x}$ in advance. Instead of using hypotheses from $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ as building blocks to construct meta-hypotheses, we use multiple instantiations of an online algorithm for $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ as our building blocks to construct meta-algorithms. The MW algorithm is then used as a meta-meta-algorithm to combine these meta-algorithms.

We let $\mathcal{A}_K := \{a[1], \ldots, a[m]\}$ denote our set of m instantiations that will act as a surrogate for the hypothesis class $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$. We then construct the set, $\bar{\mathcal{A}}_K(k, m, s, T^1, \ldots, T^s) := \{\bar{\boldsymbol{a}} \in \mathcal{A}_K^T : k = k(\bar{\boldsymbol{a}}), m = |m(\bar{\boldsymbol{a}})|\}$. Each $\bar{\boldsymbol{a}} \in \bar{\mathcal{A}}_K$ now defines a meta-algorithm for the multitask setting. That is, given an online multitask data sequence $(x_1^i, y_1^i), \ldots, (x_{T^j}^j, y_{T^j}^j)$, each element of $\bar{\boldsymbol{a}}$ will "color" the

[1]Technically, when we say that an algorithm *achieves a bound*, it may be that the algorithm depends on a small set of parameters which we have then assumed are "tuned" optimally.

corresponding data pair with one of the m instantiations (we will use the function $\alpha : \{(t,i) : t \in [T^i], i \in [s]\} \to [m]$ to denote this mapping with respect to $\bar{\boldsymbol{a}}$). Each instantiation will *receive as inputs only the online sequence of the data pairs corresponding to its "color"*; likewise, the prediction of meta-algorithm $\bar{\boldsymbol{a}}$ will be that of the instantiation active on that trial. We will use as our base algorithm OGD_K. Thus for the meta-algorithm $\bar{\boldsymbol{a}}$ with (binary) predictions $\bar{\boldsymbol{a}}(x_t^i)$, we have from (5.1),

$$\sum_{i=1}^{s}\sum_{t=1}^{T^i}\mathbb{E}[\mathcal{L}_{01}(y_t^i, \bar{\boldsymbol{a}}(x_t^i))] \leq \frac{1+\lambda}{2}\sum_{i=1}^{s}\sum_{t=1}^{T^i}\mathcal{L}_{01}(y_t^i, \mathrm{sign}(h[\alpha(_t^i)](x_t^i))) + \sum_{j=1}^{m} O\left(\sqrt{\|h[j]\|_K^2\, X^2 T}\right) \quad (5.2)$$

for any received instance sequence $\boldsymbol{x} \in \mathcal{X}^T$ and for any $h[1], \ldots, h[m] \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)}$.

The MW algorithm [19, 88, 11] does not work just for hypothesis classes; more generally, it works for collections of algorithms. Hence we may run the MW as a meta-meta-algorithm to combine all of the meta-algorithms $\bar{\boldsymbol{a}} \in \bar{\mathcal{A}}_K$, with the corresponding regret bound of

$$\sum_{i=1}^{s}\sum_{t=1}^{T^i}\mathbb{E}[\mathcal{L}_{01}(y_t^i, \hat{y}_t^i)] \leq \sum_{i=1}^{s}\sum_{t=1}^{T^i}\mathbb{E}[\mathcal{L}_{01}(y_t^i, \bar{\boldsymbol{a}}(x_t^i))] + O\left(\sqrt{\log(|\bar{\mathcal{A}}_K|)T}\right). \quad (5.3)$$

Thus by substituting the loss for each meta-algorithm $\bar{\boldsymbol{a}}$ (the R.H.S. of (5.2)) into (5.3) and using the upper bound $\binom{T-s}{k} m^s (m-1)^k$ for the cardinality of $\bar{\mathcal{A}}_K$, we obtain (using upper bounds for binomial coefficients and the inequality $\sum_i \sqrt{p_i q_i} \leq \sqrt{(\sum_i p_i)(\sum_i q_i)}$),

$$R_T^{\frac{1+\lambda}{2}}(\boldsymbol{h}^*) \in O\left(\sqrt{(\textstyle\sum_{h\in m(\boldsymbol{h}^*)}\|h\|_K^2\, X_K^2 + s\log m + k\log m + k\log((T-s)/k))T}\right), \quad (5.4)$$

for any received instance sequence $\boldsymbol{x} \in \mathcal{X}^T$ and for any $\boldsymbol{h}^* \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)^T}$ such that $k = k(\boldsymbol{h}^*)$ and $m = |m(\boldsymbol{h}^*)|$.

The term $\sum_{h\in m(\boldsymbol{h}^*)}\|h\|_K^2\, X_K^2$ may be viewed as a *learner complexity*, i.e., the price we "pay" for identifying the hypotheses that fit the modes. A salient feature

of long-term memory bounds is that although the data pairs associated with each hypothesis are intermixed in the multitask sequence, we pay the learner complexity only modestly in terms of potentially leading multiplicative constants. A switching algorithm without long-term memory "forgets" and pays the full price for a mode on every switch or new task. We have thus given an exponential-time algorithm for $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ with $O(1)$ leading multiplicative constants. In Section 5.3, we give an efficient algorithm for $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$, with a time complexiy of $O(T^3)$ per trial, and in terms of learner complexities, it gains only leading multiplicative constants of $O(\lambda^2 \log T)$. In the case of binary RKHS functions ($\lambda = 1$), this becomes $O(\log T)$.

5.2.2 Related Work

In this section we briefly describe other related work in the online setting that considers either *switching* or *multitask* models.

The first result for switching in the experts model was the WML algorithm [19], which was generalized in [89]. There is an extensive literature building on those papers, with some prominent results including [83, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99]. Relevant for our model are those papers [83, 91, 94, 92, 97, 98, 99] that address the problem of long-term memory ($m \ll k$), in particular [83, 91, 94].

Analogous to the problem of long-term memory in online learning is the problem of catastrophic forgetting in artificial neural network research [100, 101]. That is the problem of how a system can adapt to new information without forgetting the old. In online learning, that is the problem of how an algorithm can both quickly adapt its prediction hypothesis and recall a previously successful prediction hypothesis when needed. In the experts model this problem was first addressed by [83], which gave an algorithm that stores each of its past state vectors, and then at each update mixes these vectors into the current state vector. In [91], an algorithm and bounds were given that extended the base comparision class of experts to include Bernoulli models. An improved algorithm with a Bayesian intepretation based on the idea of "circadian specialists" was given for this setting in [94]. In the work that this chapter is based on [8], an algorithm for multitask learning with memory in the expert setting is given, and is directly inspired by the methodology in [94].

The problem of linear regression with long term memory was posed as an open problem in [94, Sec. 5]. Algorithm 4 gives an algorithm for linear regression for bounded functions in a RKHS with a regret bound that reflects long-term memory. Switching linear prediction has been considered in [85, 14, 102, 39]. Only [39] addresses the issue of long-term memory. The methodology of [39] is a direct inspiration for Algorithm 4. We significantly extend the result of [39, Eq. (1)]. Their result was i) restricted to a mistake as opposed to a regret bound, ii) restricted to finite positive definite matrices and iii) in their mistake bound the term analogous to $\sum_{h \in m(\boldsymbol{h}^*)} \|h\|_K^2 X_K^2$ was increased by a multiplicative factor of $\tilde{O}(|m(\boldsymbol{h}^*)|)$, a significantly weaker result.

Multitask learning has been considered extensively in the batch setting, with some prominent early results including [103, 104, 105]. In the online multitask *expert* setting, [106, 107, 108, 94] considered a model where each task is associated only with a single hypothesis, i.e., no internal switching within a task. Also in the expert setting, [109, 110] considered models where the prediction was made for all tasks simultaneously. In [110], the aim was to predict well relative to a set of possibly predefined task interrelationships and in [109], the interrelationships were to be discovered algorithmically. The online multitask *linear* prediction setting was considered in [111, 112, 113]. The models of [112, 113] are similar to ours, but like previous work in the expert setting, these models are limited to one "hypothesis" per task. In the work of [111], the predictions were made for all tasks simultaneously through a joint loss function.

5.3 RKHS Hypothesis Classes

Our algorithm and its analysis builds on Algorithm 3 for online inductive matrix completion with side-information and its corresponding performance guarantee in Theorem 1. We recall the following notation from Section 1.3.

The max-norm (or γ_2 norm [9]) of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m \times n}$ is defined by

$$\|\boldsymbol{U}\|_{\max} := \min_{\boldsymbol{P}\boldsymbol{Q}^\top = \boldsymbol{U}} \left\{ \max_{1 \le i \le m} \|\boldsymbol{P}_i\| \times \max_{1 \le j \le n} \left\|\boldsymbol{Q}_j\right\| \right\}, \tag{5.5}$$

where the minimum is over all matrices $\boldsymbol{P} \in \mathfrak{R}^{m\times d}$ and $\boldsymbol{Q} \in \mathfrak{R}^{n\times d}$ and every integer d. We denote the class of $m \times d$ *row-normalized* matrices as $\mathcal{N}^{m,d} := \{\hat{\boldsymbol{P}} \subset \mathfrak{R}^{m\times d} : \left\|\hat{\boldsymbol{P}}_i\right\| = 1,\ i \in [m]\}$. The quasi-dimension of a matrix is defined as follows.

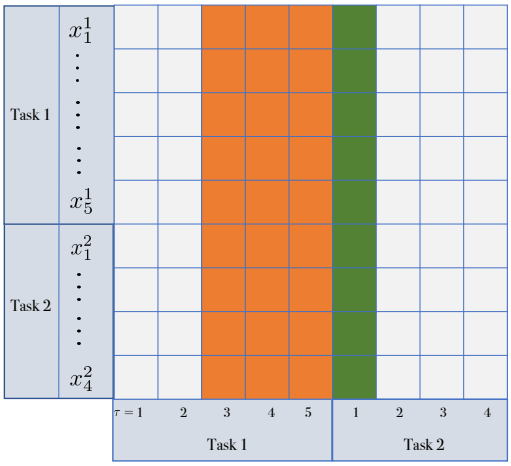
Definition. *The* quasi-dimension *of a matrix $\boldsymbol{U} \in \mathfrak{R}^{m\times n}$ with respect to $\boldsymbol{M} \in \boldsymbol{S}^m_{++}$, $\boldsymbol{N} \in \boldsymbol{S}^n_{++}$ at γ as*

$$\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top=\gamma\boldsymbol{U}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top \boldsymbol{M}\hat{\boldsymbol{P}}\right)\mathcal{R}_{\boldsymbol{M}} + \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top \boldsymbol{N}\hat{\boldsymbol{Q}}\right)\mathcal{R}_{\boldsymbol{N}}\,, \qquad (5.6)$$

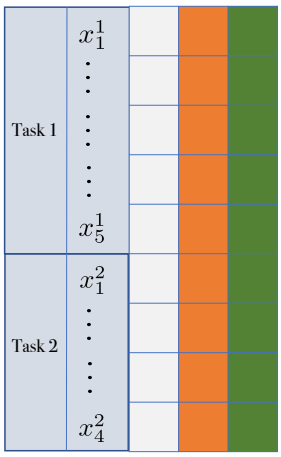
where the infimum is over all row-normalized matrices $\hat{\boldsymbol{P}} \in \mathcal{N}^{m,d}$ and $\hat{\boldsymbol{Q}} \in \mathcal{N}^{n,d}$ and every integer d. If the infimum does not exist then $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := +\infty$ (The infimum exists iff $\|\boldsymbol{U}\|_{max} \le 1/\gamma$).

Algorithm 3 addresses the problem of the online prediction of a comparator matrix $\boldsymbol{U}$ with side information. The side information is supplied as a pair of kernels over the row indices and the column indices. In Theorem 1, a regret bound $\tilde{O}\Big(\sqrt{(\widehat{\mathcal{D}}/\gamma^2)T}\Big)$ is given, where $1/\gamma^2 \ge \|\boldsymbol{U}\|^2_{\max}$ and $\widehat{\mathcal{D}} \ge \mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$ are parameters of the algorithm that serve as upper estimates on $\|\boldsymbol{U}\|^2_{\max}$ and $\mathcal{D}^{\gamma}_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U})$. The first estimate $1/\gamma^2$ is an upper bound on the squared max-norm (Eq. (5.5)), which like the trace-norm may be seen as a convex proxy for the rank of the matrix [3]. The second estimate $\widehat{\mathcal{D}}$ is an upper bound of the quasi-dimension (Eq. (5.6)), which measures the quality of the side-information. The quasi-dimension depends upon the "best" factorization $(1/\gamma)\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \boldsymbol{U}$, which will be smaller when the row $\hat{\boldsymbol{P}}$ (column $\hat{\boldsymbol{Q}}$) factors are in congruence with the row (column) kernel. We bound the quasi-dimension in Theorem 71 in Section 5.6 as a key step to proving Theorem 69.

In the reduction of our problem to a matrix completion problem with side information, the row indices correspond to the domain of the learner-supplied kernel K and the column indices correspond to the temporal dimension. On each trial we receive an x^τ (a.k.a. x^i_t). The comparator matrix $\boldsymbol{U}$ is now an embedding of $\boldsymbol{H}$, where the column of $\boldsymbol{H}$ corresponding to time τ will contain the entries $H^\tau = (h^\tau(x^\upsilon))_{\upsilon\in[T]}$. Fig 5.3 illustrates an example comparator matrix and its corresponding latent matrix $\boldsymbol{U}^*$. Although we are predicting functions that are changing over time, the



(a) Example matrix $\boldsymbol{U}$. Each row corresponds to a distinct instance, and the columns represent the different trials. Both rows and columns are ordered by task for convenience. On each trial, we are asked to predict a diagonal element of this matrix.



(b) Corresponding latent matrix $\boldsymbol{U}^*$. The rows represent the instances, and the columns correspond to the distinct hypotheses, m.

Figure 5.3: The reduction of the online multitask learning problem to the matrix completion problem with a (T,m)-biclustered matrix $\boldsymbol{U}$. In this example, $m = 3$.

underlying assumption is that the change is sporadic; otherwise it is infeasible to prove a non-vacuous bound. Thus we expect $H_t^i \approx H_{t+1}^i$ and as such our column side-information kernel should reflect this expectation. Topologically, we would therefore expect a kernel to present as s separate time *paths*, where nearness in time corresponds to nearness on the path. In the following we introduce the *path-tree-kernel* (the essence of the construction was first introduced in [114]), which satisfies this expectation in the single-task case. We then adapt this construction to the multitask setting.

A *path-tree* kernel $P : [T] \times [T] \to \mathfrak{R}$, is formed via the Laplacian of a fully complete binary *tree* with $N := 2^{\lceil \log_2 T \rceil + 1} - 1$ vertices. The *path* corresponds to the first T leaves of the tree, numbered sequentially from the leftmost to the rightmost leaf of the first T leaves. Denote this Laplacian as $\boldsymbol{L}$ where the path is identified with $[T]$ and the remaining vertices are identified with $[N] \setminus [T]$. Then using the definition $\boldsymbol{L}^\circ := \boldsymbol{L} + \left(\frac{1}{N}\right)\left(\frac{1}{N}\right)^\top \mathcal{R}_{\boldsymbol{L}}^{-1}$, we define $P(\tau, \upsilon) := (\boldsymbol{L}^\circ)^+_{\tau\upsilon}$ where $\tau, \upsilon \in [T]$. We extend the path-tree kernel to a *multitask-path-tree* kernel by dividing the path into s contiguous segments, where segment i is a path

of length T^i, and the task vector $\boldsymbol{\ell} \in [s]^T$ determines the mapping from global trial τ to task ℓ^τ and local trial $\boldsymbol{\sigma}(\tau)$. We define $\tilde{P}^{\boldsymbol{\ell},T^1,\dots,T^s} : [T] \times [T] \to \mathfrak{R}$ as $\tilde{P}^{\boldsymbol{\ell},T^1,\dots,T^s}(\tau,\upsilon) := P\left(\sum_{i=1}^{\ell^\tau-1} T^i + \boldsymbol{\sigma}(\tau), \sum_{i=1}^{\ell^\upsilon-1} T^i + \boldsymbol{\sigma}(\upsilon)\right)$. Observe we do not need to know the task vector $\boldsymbol{\ell}$ in advance; we only require upper bounds on the lengths of the tasks to be able to use this kernel. Finally, we note that it is perhaps surprising that we use a tree rather than a path directly. We discuss this issue following Lemma 73 in Section 5.6.

Algorithm 4 requires $O(t^3)$ time per trial t since we need to compute the eigendecomposition of three $O(t) \times O(t)$ matrices as well as sum $O(t) \times O(t)$ matrices up to t times. We bound the regret of the algorithm as follows.

Algorithm 4 Predicting $\mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ in a switching multitask setting.

Parameters: Tasks $s \in \mathbb{N}$, task lengths $T^1,\dots,T^s \in \mathbb{N}$, $T := \sum_{i=1}^s T^i$, learning rate: $\eta > 0$, complexity estimate: $\hat{C} > 0$, modes: $m \in [T]$, RKHS function upper bound: $\lambda \geq 1$,SPD Kernel $K : \mathcal{X} \times \mathcal{X} \to \mathfrak{R}$, $\tilde{P} := \tilde{P}^{\boldsymbol{\ell},T^1,\dots,T^s} : [T] \times [T] \to \mathfrak{R}$, with $\max_{\tau\in[T]} K(x^\tau,x^\tau) \leq \hat{X}_K^2$, and $\hat{X}_P^2 := 2\lceil \log_2 T \rceil$.

Initialization: $\mathbb{U} \leftarrow \emptyset, \mathcal{X}^1 \leftarrow \emptyset, \mathcal{T}^1 \leftarrow \emptyset$.

For $\tau = 1,\dots,T$

- Receive task $\ell^\tau \in [s]$.
- Receive $x^\tau \in \mathcal{X}$.
- Set $i \leftarrow \ell^\tau$; $t \leftarrow \boldsymbol{\sigma}(\tau)$; $x_t^i \equiv x^\tau$.
- Define

$$\boldsymbol{K}^\tau := (K(x,z))_{x,z\in\mathcal{X}^\tau\cup\{x^\tau\}}; \quad \boldsymbol{P}^\tau := (\tilde{P}(\tau,\upsilon))_{\tau,\upsilon\in\mathcal{T}^\tau\cup\{\tau\}},$$

$$\tilde{\boldsymbol{X}}^\tau(\upsilon) := \left[\frac{\sqrt{\boldsymbol{K}^\tau}\boldsymbol{e}^{x^\upsilon}}{\sqrt{2\hat{X}_K^2}}; \frac{\sqrt{\boldsymbol{P}^\tau}\boldsymbol{e}^{\upsilon}}{\sqrt{2\hat{X}_P^2}}\right]\left[\frac{\sqrt{\boldsymbol{K}^\tau}\boldsymbol{e}^{x^\upsilon}}{\sqrt{2\hat{X}_K^2}}; \frac{\sqrt{\boldsymbol{P}^\tau}\boldsymbol{e}^{\upsilon}}{\sqrt{2\hat{X}_P^2}}\right]^\top,$$

$$\tilde{\boldsymbol{W}}^\tau \leftarrow \exp\left(\log\left(\frac{\hat{C}}{2Tm\lambda^2}\right)\boldsymbol{I}^{|\mathcal{X}^\tau|+|\mathcal{T}^\tau|+2} + \sum_{\upsilon\in\mathbb{U}} \eta y_\upsilon \tilde{\boldsymbol{X}}^\tau(\upsilon)\right).$$

- Predict

$$Y^\tau \sim \textsc{Uniform}\left(-\frac{1}{\sqrt{m\lambda^2}}, \frac{1}{\sqrt{m\lambda^2}}\right); \quad \bar{y}^\tau \leftarrow \operatorname{tr}\left(\tilde{\boldsymbol{W}}^\tau \tilde{\boldsymbol{X}}^\tau(\tau)\right) - 1; \quad \hat{y}_t^i := \hat{y}^\tau \leftarrow \operatorname{sign}(\bar{y}^\tau - Y^\tau).$$

- Receive label $y_t^i := y^\tau \in \{-1,1\}$.
- If $y^\tau \bar{y}^\tau \leq \frac{1}{\sqrt{m\lambda^2}}$ then

$$\mathbb{U} \leftarrow \mathbb{U} \cup \{\tau\}, \ \mathcal{X}^{\tau+1} \leftarrow \mathcal{X}^\tau \cup \{x^\tau\}, \text{ and } \mathcal{T}^{\tau+1} \leftarrow \mathcal{T}^\tau \cup \{\tau\}.$$

- Else $\mathcal{X}^{\tau+1} \leftarrow \mathcal{X}^\tau$ and $\mathcal{T}^{\tau+1} \leftarrow \mathcal{T}^\tau$.

Theorem 69. *The $\frac{1+\lambda^2}{2}$-regret of Algorithm 4 with upper estimates, $k \geq k(\boldsymbol{h}^*)$, $m \geq |m(\boldsymbol{h}^*)|$,*

$$\hat{C} \geq C(\boldsymbol{h}^*) := \lambda^2 \left(\sum_{h \in m(\boldsymbol{h}^*)} \|h\|_K^2 \, X_K^2 + 2(s+k-1)m\lceil \log_2 T \rceil^2 + 2m^2 \right),$$

$\hat{X}_K^2 \geq \max_{\tau \in [T]} K(x^\tau, x^\tau)$, and learning rate $\eta = \sqrt{\frac{\hat{C}\log(2T)}{2Tm\lambda^2}}$ is bounded by

$$\sum_{i=1}^{s} \sum_{t=1}^{T^i} \mathbb{E}[\mathcal{L}_{01}(y_t^i, \hat{y}_t^i)] - \frac{1+\lambda^2}{2} \mathcal{L}_{01}(y_t^i, \operatorname{sign}(h_t^i(x_t^i))) \leq 3.5 \sqrt{\hat{C}\, T \log(2T)} \tag{5.7}$$

with received instance sequence $\boldsymbol{x} \in \mathcal{X}^T$ and for any $\boldsymbol{h}^* \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)^T}$ where $\lambda \geq 1$.

Comparing roughly to the bound of the exponential-time algorithm (see (5.4)), we see that the $\log m$ term has been replaced by an m term, we have worse scalings in terms of λ, and we have gained a multiplicative factor of $\log 2T$. From the perspective of long-term memory, we note that the potentially dominant learner complexity term $\sum_{h \in m(\boldsymbol{h}^*)} \|h\|_K^2 \, X_K^2$ has only increased by a slight $\log 2T$ term if $\lambda = 1$. To gain more insight into the problem, we also have the following simple lower bound. We observe that even though it is written in terms of some $\boldsymbol{h}^* \in \mathcal{H}_K^{(\boldsymbol{x},1)^T}$, it is also a lower bound for $\mathcal{H}_K^{(\boldsymbol{x},\lambda)^T}$, since $\mathcal{H}_K^{(\boldsymbol{x},1)^T} \subseteq \mathcal{H}_K^{(\boldsymbol{x},\lambda)^T}$.

Proposition 70. *For any (randomized) algorithm and any $s, k, m, \Gamma \in \mathbb{N}$, with $k+s \geq m > 1$ and $\Gamma \geq m \log_2 m$, there exists a kernel K and a $T_0 \in \mathbb{N}$ such that for every $T \geq T_0$:*

$$\sum_{\tau=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y^\tau, \hat{y}^\tau)] - \mathcal{L}_{01}(y^\tau, h^\tau(x^\tau)) \in \Omega\left(\sqrt{(\Gamma + s\log m + k \log m)\, T} \right),$$

for some multitask sequence $(x^1, y^1), \ldots, (x^T, y^T) \in (\mathcal{X} \times \{-1, 1\})^T$ and some $\boldsymbol{h}^* \in [\mathcal{H}_K^{(\boldsymbol{x},1)}]^T$ such that $m \geq |m(\boldsymbol{h}^*)|$, $k \geq k(\boldsymbol{h}^*)$, $\sum_{h \in m(\boldsymbol{h}^*)} \|h\|_K^2 \, X_K^2 \geq |m(\boldsymbol{h}^*)| \log_2 m$, where $X_K^2 = \max_{\tau \in [T]} K(x^\tau, x^\tau)$.

Comparing the above proposition to the bound of the exponential-time algorithm (see (5.4)), the most striking difference is the absence of the $\log T$ terms. We

conjecture that these terms are not necessary for the zero-one loss. In the following section, we give a proof sketch of Theorem 69.

5.3.1 Proof Sketch of Theorem 69 and Theorem 71

In the proof, we give a reduction of Algorithm 4 to Algorithm 3. Two necessary additional results that we need include Theorem 71 and Corollary 74. In Corollary 74, we bound a normalized margin-like quantity of the multitask-path-tree kernel used in the algorithm. Then in Theorem 71, we bound the quasi-dimension which indicates how to set the parameters of Algorithm 4 as well as determines the value of $C(\boldsymbol{h}^*)$ in the main theorem. As this bound of the quasi-dimension is a key element of our proof, we contrast it to a parallel result proved in Theorem 57.

We recall that the regret (see Theorem 1) of Algorithm 3 is $\tilde{O}\Big(\sqrt{(\widehat{\mathcal{D}}/\gamma^2)T}\Big)$ where $1/\gamma^2 \geq \|\boldsymbol{U}\|_{\max}^2$ and $\widehat{\mathcal{D}} \geq \mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma}(\boldsymbol{U})$. We will prove in our setting $1/\gamma^2 = m\lambda^2 \geq \|\boldsymbol{U}\|_{\max}^2$ in the discussion following (5.26).

We now contrast our bound on the quasi-dimension (Theorem 71) to the bound of Theorem 57. The quasi-dimension depends on γ so that if $\gamma \leq \gamma'$, then $\mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma}(\boldsymbol{U}) \leq \mathcal{D}_{\boldsymbol{M},\boldsymbol{N}}^{\gamma'}(\boldsymbol{U})$. In Theorem 57, the given bound on quasi-dimension is independent of the value of γ. Thus to minimize the regret bound of $\tilde{O}\Big(\sqrt{(\widehat{\mathcal{D}}/\gamma^2)T}\Big)$, it is sensible in Theorem 57 to select the smallest possible $1/\gamma^2 = \|\boldsymbol{U}\|_{\max}^2$. The situation in this chapter is essentially reversed. In the following theorem, it is required that $1/\gamma^2 = m\lambda^2 \geq \|\boldsymbol{U}\|_{\max}^2$. In fact, $m \max_{ij} |U_{ij}|^2$ is the maximum possible value of the squared max norm in the case that $m = |m(\boldsymbol{h}^*)|$ with respect to all possible comparators $\boldsymbol{h}^*$ (see (5.26)). Thus in contrast to Theorem 57, the following result trades off a potentially larger value in $1/\gamma^2$ for a smaller possible $\widehat{\mathcal{D}}$. If we were instead to use the bound of Theorem 57, then the term in this chapter $\sum_{h \in m(\boldsymbol{h}^*)} \|h\|_K^2 X_K^2$ would gain a leading multiplicative factor of m^2 (terrible!).

We recall the following notation. The class of $m \times d$ *row-normalized* is denoted as $\mathcal{N}^{m,d} := \{\hat{\boldsymbol{P}} \subset \Re^{m \times d} : \big\|\hat{\boldsymbol{P}}_i\big\| = 1, i \in [m]\}$ and the class of *block expansion* matrices is defined as $\mathcal{B}^{m,d} := \{\boldsymbol{R} \subset \{0,1\}^{m \times d} : \|\boldsymbol{R}_i\| = 1 \text{ for } i \in [m], \text{rank}(\boldsymbol{R}) = d\}$. Block expansion matrices may be seen as a generalization of permutation matrices, additionally duplicating rows (columns) by left (right) multiplication.

We define the class of (k, ℓ, λ)-biclustered matrices as

$$\mathbb{B}_{k,\ell}^{m,n,\lambda} = \{\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top \in ([-\lambda,-1]\cup[1,\lambda])^{m\times n} : \boldsymbol{R} \in \mathcal{B}^{m,k}, \boldsymbol{C} \in \mathcal{B}^{n,\ell}, \\ \boldsymbol{U}^* \in ([-\lambda,-1]\cup[1,\lambda])^{k\times\ell}\}\,.$$

In the case that $\lambda = 1$, we retrieve the definition of the (k, ℓ)-binary-biclustered matrices as seen in Chapter 4.

Theorem 71. *If* $\boldsymbol{H} \in \mathbb{B}_{p,m}^{p,T,\lambda}$, $\gamma = 1/\sqrt{m\lambda^2}$, $\lambda \geq 1$ *and if*

$$\mathcal{D}^\star_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{H}) := \gamma^2\lambda^2 \operatorname{tr}((\boldsymbol{H}^*)^\top \boldsymbol{M}\boldsymbol{H}^*)\mathcal{R}_{\boldsymbol{M}} + \operatorname{tr}(\boldsymbol{C}^\top\boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}} \tag{5.8}$$

is defined as the minimum over all decompositions of $\boldsymbol{H} = \boldsymbol{H}^*\boldsymbol{C}^\top$ *for* $\boldsymbol{H}^* \in ([-\lambda,-1]\cup[1,\lambda])^{p\times m}$, $\boldsymbol{C} \in \mathcal{B}^{T,m}$ *then for* $\boldsymbol{U} = \boldsymbol{D}\boldsymbol{H}$ *where* $\boldsymbol{D}$ *is a diagonal matrix with entries* $D_{ii} = \frac{1}{\gamma\|\boldsymbol{H}_i^*\|}$ *for* $i \in [p]$

$$\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) \leq \mathcal{D}^\star_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{H}) \qquad (\gamma = 1/\sqrt{m\lambda^2})\,.$$

Proof. Recall by supposition $\gamma = 1/\sqrt{m\lambda^2}$, and define $\boldsymbol{U}^* := \boldsymbol{D}\boldsymbol{H}^*$. Note that for all $i \in [p]$, $\|\boldsymbol{U}_i^*\| = \frac{1}{\gamma}$. Set $\hat{\boldsymbol{P}}' := \gamma\boldsymbol{U}^*$ and $\hat{\boldsymbol{Q}}' := \boldsymbol{C}$ hence $\hat{\boldsymbol{P}}' \in \mathcal{N}^{p,m}$, $\hat{\boldsymbol{Q}}' \in \mathcal{N}^{T,m}$ and $\hat{\boldsymbol{P}}'\hat{\boldsymbol{Q}}'^\top = \gamma\boldsymbol{U}^*\boldsymbol{C}^\top = \gamma\boldsymbol{D}\boldsymbol{H}^*\boldsymbol{C}^\top = \gamma\boldsymbol{U}$.

Recall (5.6),

$$\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) := \min_{\hat{\boldsymbol{P}}\hat{\boldsymbol{Q}}^\top = \gamma\boldsymbol{U}} \operatorname{tr}\left(\hat{\boldsymbol{P}}^\top\boldsymbol{M}\hat{\boldsymbol{P}}\right)\mathcal{R}_{\boldsymbol{M}} + \operatorname{tr}\left(\hat{\boldsymbol{Q}}^\top\boldsymbol{N}\hat{\boldsymbol{Q}}\right)\mathcal{R}_{\boldsymbol{N}}\,. \tag{5.9}$$

Observe that $(\hat{\boldsymbol{P}}', \hat{\boldsymbol{Q}}')$ is in the feasible set of the above optimization. Hence

$$\begin{aligned}\mathcal{D}^\gamma_{\boldsymbol{M},\boldsymbol{N}}(\boldsymbol{U}) &\leq \operatorname{tr}\left(\hat{\boldsymbol{P}}'^\top\boldsymbol{M}\hat{\boldsymbol{P}}'\right)\mathcal{R}_{\boldsymbol{M}} + \operatorname{tr}\left(\hat{\boldsymbol{Q}}'^\top\boldsymbol{N}\hat{\boldsymbol{Q}}'\right)\mathcal{R}_{\boldsymbol{N}} \\ &= \gamma^2 \operatorname{tr}((\boldsymbol{U}^*)^\top\boldsymbol{M}\boldsymbol{U}^*)\mathcal{R}_{\boldsymbol{M}} + \operatorname{tr}(\boldsymbol{C}^\top\boldsymbol{N}\boldsymbol{C})\mathcal{R}_{\boldsymbol{N}}\,.\end{aligned}$$

We also have for all $i \in [p]$, $j \in [m]$ that $\boldsymbol{U}_{ij}^* = D_{ii}\boldsymbol{H}_{ij}^* = \frac{\sqrt{m\lambda^2}}{\|\boldsymbol{H}_i^*\|}\boldsymbol{H}_{ij}^*$. Recalling

that $|\boldsymbol{H}^*_{ij}| \geq 1$ so that $\|\boldsymbol{H}^*_i\| \geq \sqrt{m}$, we then have

$$\boldsymbol{U}^*_{ij} \leq \lambda \boldsymbol{H}^*_{ij}.$$

Observing that $\mathrm{tr}((\boldsymbol{U}^*)^\top \boldsymbol{M}\boldsymbol{U}^*)$ scales quadratically with the entries of $\boldsymbol{U}^*$ then gives the result. □

5.4 Synthetic Experiments

As the online multitask setting with memory setting is a novel one, there are no alternative algorithms that we can directly compare against. Instead, we show the memory and multitask features of our algorithm separately. To demonstrate the former, we compare our algorithm to the (kernel) switching and non-switching perceptron algorithms in the single task case. Then we show the multitask learning feature by comparing our algorithm which learns over 1024 trials in a single task to the same algorithm which learns across two tasks, with 1024 trials per task. On each trial, the task number is selected uniformly at random, until all 1024 trials have been exhausted for one of the tasks; when this happens, all trials will occur in the remaining task. In the following experiments, we will assess the performance of our algorithm with the RKHS space induced by the Dirac kernel. In addition, we consider introducing decay methods, which reduces the time complexity of the algorithm.

5.4.1 Dirac Kernel

In this experiment, we consider the instance space $\mathcal{X} = [d]$, hypothesis space $\mathcal{H}_d = \{-1, 1\}^d$ and the kernel $\mathcal{K}_d(x, x') = 2[x = x'] - 1$. For this setup, we have that $\max_{x\in[d]} K_d(x, x) = 1$ and that $\|h\|^2_{K_d} = d$ for all $h \in \mathcal{H}_d$. Since the hypotheses are binary, we set $\lambda = 1$, and consider 3 modes ($m = 3$). All hypotheses and instances are sampled uniformly at random from their respective spaces. We set the parameters according to Theorem 69, with $\hat{C} := 3d + 6(s + k - 1)m\lceil\log_2 T\rceil^2 + 18$, $\eta = \sqrt{\frac{\hat{C}\log(2T)}{6T}}$, and average our results across 10 runs. For all experiments, we feed identical input sequences to the different algorithms.

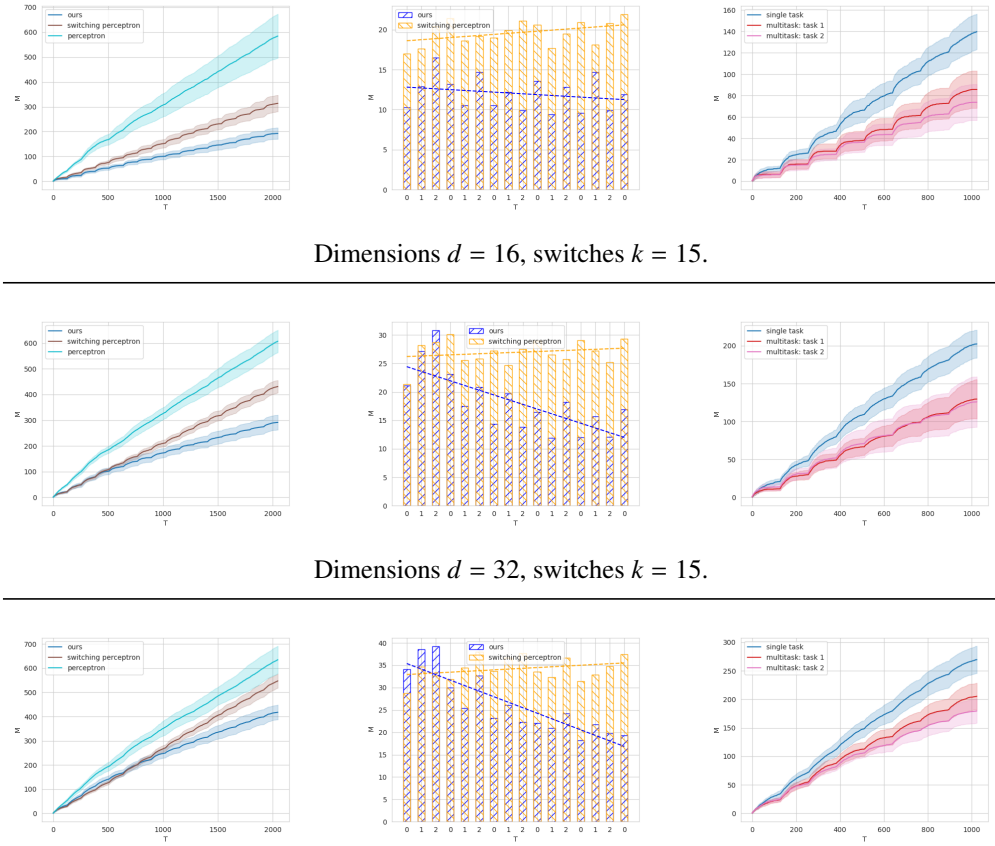


Figure 5.4: Results for the worst-case kernel. The shaded regions show the standard deviation. For the plots in the second column, the dotted lines show the linear fit.

We consider dimensions $d \in \{16, 32, 64\}$, $k = 15$ switches which occur at regular intervals every 64 trials and switch between the three modes consecutively. Across the different setups, the first two columns of Figure 5.4 show that in the single task case ($s = 1$), our algorithm has fewer cumulative mistakes than the perceptron algorithms and our algorithm exhibits memory behaviour as the average number of mistakes per segment decreases over time. Figure 5.4 (third column) shows that the mistakes per task in the multitask setting with $s = 2$ are consistently lower compared to the single task setting.

5.4.2 Decay Methods

The time complexity of our algorithm is $O(|\mathbb{U}_t|^3)$ per trial t. We therefore seek to constrain $|\mathbb{U}_t| \leq B$ for all t, by applying the decay methods inspired by [115, 102], bounds for which have been proven only for the kernel perceptron. In [115], the contributions of the elements in $\mathbb{U}_t$ are decayed after which the earliest trial in $\mathbb{U}_t$ is removed. In contrast, [102] has the simpler strategy of removing an element at random without any decay.[2]. For these experiments, we consider the same setup as the one in Section 5.4.1. Figure 5.5 shows the results. For simplicity, we did not shrink the contributions for the strategy in [115]. We choose the smallest B for which there was no overlap between the error bars of the switching perceptron and the decay method in the single task setting. We used this value of B for both the single task and multitask experiments. For the random removal strategy, we required a higher $B = 192$, whereas the earliest trial removal strategy only required $B = 48$. For both methods, there was no discernible memory behaviour as the number of mistakes per segments increases. However, the multitask decay methods did outperform the single task algorithm with no decay in both cases.

5.5 Discussion

We have presented a novel multitask setting which generalizes single-task switching under the long-term memory setting. We gave an algorithm for the RKHS hypothesis class with per trial prediction times of $O(t^3)$, for which we proved both upper and lower bounds on the regret. An open problem is to resolve the gap between the two bounds. On the algorithmic side, the algorithm depends on a number of parameters, such as $\hat{C}$, which depends on s, k and m. The knowledge of these parameters can be circumvented through the doubling trick. Alternatively, there is extensive research in online learning methods to design parameter-free methods. Can some of these methods be applied here (see e.g., [116])? For a non-parametric hypothesis class, intuitively it seems we must expect some dependence on T. In terms of the time

[2]For simplicity, we implement the transductive algorithm (Algorithm 1). This is possible due to the finite-dimensional domains of both the row and column kernels, and is equivalent to the inductive algorithm due to Proposition 6.

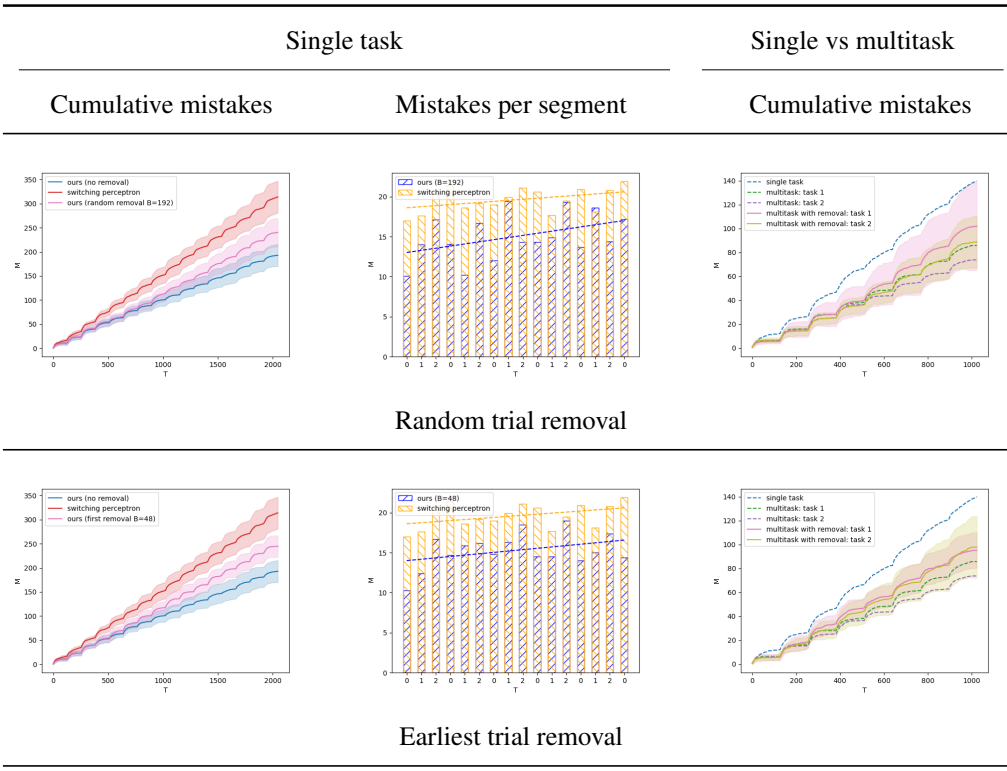


Figure 5.5: Results for the decay methods.

complexity, can we perhaps prove bounds for sketching methods [80] that have had success in simpler models to improve the running times? More broadly, for what other infinite hypothesis classes can we give efficient regret-bounded algorithms in this switching multitask setting with long-term memory?

5.6 Proofs

5.6.1 Proof of Theorem 69

Proof of Corollary 74

In this section, we prove Corollary 74, which is utilized in the proof of Theorem 69. The corollary is first stated below.

Corollary 74. *If* $f \in \mathcal{H}_{\tilde{P}} \cap \{0,1\}^T$ *then*

$$\|f\|_{\tilde{P}}^2 \max_{\tau \in [T]} \tilde{P}(\tau, \tau) \leq (k(f) + s(f))\lceil \log_2 T \rceil^2 + 2\,, \tag{5.10}$$

where $\tilde{P} = \tilde{P}^{\ell, T^1, \ldots, T^s}$, $k(f) := \sum_{i=1}^{s} \sum_{t=1}^{T^i - 1} [f(\binom{i}{t}) \neq f(\binom{i}{t+1})]$ *and* $s(f) := \sum_{i=1}^{s-1} [f(\binom{i}{T^i}) \neq f(\binom{i+1}{1})]$.

We recall the notions of *effective resistance* between vertices in a graph and the *resistance diameter* of a graph. A graph may naturally interpreted as an resistive network where each edge in the graph is viewed as a unit resistor. Thus the *effective resistance* between two vertices is the potential difference needed to induce a unit current flow between them and the *resistance diameter* is the maximum effective resistance between all pairs of vertices.

To prove the corollary, we will need to bound the diagonal element of the Laplacian pseudo-inverse by the resistance diameter. In the following Lemma, we will improve upon [10, Eq. (9)] by a factor of $\frac{1}{2}$ for the special case of fully complete trees.

Lemma 72. *For the graph Laplacian* $\boldsymbol{L} \in \mathfrak{R}^{N \times N}$ *of a fully complete tree graph,*

$$\mathcal{R}_{\boldsymbol{L}} = \max_{i \in [N]} L_{ii}^{+} \leq \frac{1}{2} \mathcal{R}_{diam}(\boldsymbol{L}),$$

where $R_{diam}(\boldsymbol{L})$ *is the resistance diameter of the graph described by* $\boldsymbol{L}$.

Proof. Before proving the result, we shall recall 4 general facts about graphs, trees and Laplacians. We also denote the set of vertices at a given depth a *level.* The root is at level 0.

1. The effective resistance between vertices i and j is given by (see [117]),

$$\mathcal{R}(i, j) = L^+_{ii} + L^+_{jj} - 2L^+_{ij}. \tag{5.11}$$

2. The diagonal element of L^+ is given by (see eg. [118])

$$L^+_{ii} = \frac{\mathcal{R}(i)}{N} - \frac{\mathcal{R}_{tot}}{N^2}, \tag{5.12}$$

   where $\mathcal{R}(i) = \sum_{j=1}^{N} \mathcal{R}(i, j)$ and $\mathcal{R}_{tot} = \sum_{i,j<i} \mathcal{R}(i, j)$.

3. For fully complete trees, we have that $\mathcal{R}(i) = \mathcal{R}(j)$ and $L^+_{ii} = L^+_{jj}$ if i and j are in the same level due to symmetry.

4. For trees, the effective resistance between vertices i and j is given by the geodesic distance (path length) between the two vertices.

Next, we prove the following intermediate result.

Lemma: For a given vertex i, the vertex j that minimizes L^+_{ij} is the leaf vertex with the largest geodesic distance from i.

Proof. Define h to be the height of the tree. We take vertex i' to be at level $k \in [h-1]$ and vertex j' at level $k+1$. Recalling that $\mathcal{R}(i) = \sum_{j=1}^{N} \mathcal{R}(i, j)$, we will consider the individual summands that compose $\mathcal{R}(j')$ and $\mathcal{R}(i')$, given by the geodesic distances between i' and j' respectively and the other vertices due to fact 4. From fact 3 (with respect to the summands), we can assume without loss of generality that vertex j' is the child of i'. Going from the summation of $\mathcal{R}(j')$ to the summation of $\mathcal{R}(i')$, there are 3 possible changes to the geodesic distances in the summation:

1. the descendants of j' will have a geodesic distance reduced by 1
2. the geodesic distance between i' and j' remains constant
3. all the other vertices will have a geodesic distance increased by 1.

Hence, defining $\mathbb{D}_{j'}$ to be the set of descendants of node j',

$$\begin{aligned}\mathcal{R}(j') &= \mathcal{R}(i') - \sum_{i \in \mathbb{D}_{j'}} 1 + \sum_{i' \in [N] \setminus (\mathbb{D}_{j'} \cup \{j'\})} 1 \\ &= \mathcal{R}(i') - |\mathbb{D}_{j'}| + N - (|\mathbb{D}_{j'}| + 1) \\ &= \mathcal{R}(i') + N - 2|\mathbb{D}_{j'}| - 1.\end{aligned}$$

This gives that $\mathcal{R}(j') - \mathcal{R}(i') \leq N$, and

$$\frac{\mathcal{R}(j') - \mathcal{R}(i')}{N} \leq 1. \tag{5.13}$$

We show that vertex j that minimizes L^+_{ij} must be a leaf vertex by contradiction. Suppose j is not a leaf vertex then there exists a child of $j \neq i$. Call the child j' which thus satisfies $\mathcal{R}(i, j') - \mathcal{R}(i, j) = 1$. Hence, Equations (5.11) and (5.12) give

$$L^+_{ij'} - L^+_{ij} = \frac{1}{2}\left(\frac{\mathcal{R}(j')}{N} - \frac{\mathcal{R}(j)}{N} - \mathcal{R}(i, j') + \mathcal{R}(i, j)\right) \tag{5.14}$$

$$\leq 0, \tag{5.15}$$

where the inequality is due to (5.13) for which we let $i' = j$. Hence, we have that $L^+_{ij} \geq L^+_{ij'}$ which is a contradiction.

Then, using Equations (5.11) and (5.12), we have

$$\operatorname*{argmin}_j L^+_{ij} = \operatorname*{argmin}_j \frac{\mathcal{R}(j)}{N} - \mathcal{R}(i, j).$$

Since all leaf vertices have the same $\mathcal{R}(i)$, the leaf vertex that minimizes must be the one with the largest geodesic distance from i. □

Recall that for a tree, the resistance diameter is equal to its geodesic diameter, and hence the vertices that maximize the effective resistance are given by the two leaf vertices with the largest geodesic distance. We therefore proceed by considering i and j to be any of the vertices that maximize the effective resistance, giving

the resistance diameter. Due to fact 3, we have $L_{ii}^+ = L_{jj}^+$. Then, from (5.11), we obtain,

$$\begin{aligned}\frac{1}{2}\mathcal{R}_{diam}(\boldsymbol{L}) &= L_{ii}^+ - L_{ij}^+ \\ &= L_{ii}^+ - \min_k L_{ik}^+ \qquad (5.16)\\ &\geq L_{ii}^+ - \frac{1}{N}\sum_{k=1}^{N} L_{ik}^+ \\ &\geq L_{ii}^+ \qquad (5.17)\end{aligned}$$

where (5.16) comes from the intermediate lemma, and (5.17) comes from the fact that $\sum_{j=1}^N L_{ij}^+ = 0$ for all $i \in [N]$ since $\boldsymbol{L}\mathbf{1} = \mathbf{0}$ for connected graphs. □

The following Lemma is essentially a simplification of the argument in [114, Section 6] for Laplacians,

Lemma 73. *(See [114, Section 6].) If $f \in \mathcal{H}_P \cap \{0,1\}^T$ then*

$$\begin{aligned}\max_{\tau\in[T]} P(t,t) &\leq 2\lceil \log_2 T\rceil \\ \|f\|_P^2 \max_{t\in[T]} P(t,t) &\leq k(f)\lceil \log_2 T\rceil^2 + 2,, \qquad (5.18)\end{aligned}$$

where $k(f) := \sum_{t=1}^{T-1}[f(t) \neq f(t+1)]$.

Proof. First we recall the following standard fact about the graph Laplacian $\boldsymbol{L}$ of an unweighted graph $\mathcal{G} = (V, E)$,

$$\boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u} = \sum_{(i,j)\in E} (u_i - u_j)^2 ,$$

where V is the set of vertices and E is the set of edges in the graph. Call this quantity the *cut* of the labeling $\boldsymbol{u}$. Consider a fully complete binary tree with a depth of $\lceil \log_2 T\rceil + 1$. For simplicity now assume that there are exactly T leaf nodes, i.e., $\log_2 T \in \mathbb{N}$. Assume some natural linear ordering[3] of the leaves. This ordering

[3]Given every three vertices in ordering (a, b, c) we have that $d(a, b) \leq d(a, c)$ where $d(p, q)$ is the path length between p and q.

then defines our *path*. We call each set of vertices at a given depth a "level" and they inherit a natural linear ordering from their children. Suppose that there are n vertices at a given level ℓ, and define $w_i^\ell := u_{v_i^\ell}$, where v_i^ℓ is the ith vertex on level ℓ. The *path-cut* at this level is given by $\sum_{i=1}^{n-1} |w_i^\ell - w_{i+1}^\ell|$.

We now proceed to argue that for a given binary labeling of a path with associated path-cut $k(f)$, we can identify a (real-numbered) labeling of the tree, such that: a. the labeling of the tree leaves is binary and consistent with that of the path and b. the tree has a cut of no more than $\frac{1}{2}k(f)\lceil \log T \rceil$. The construction is as follows: *each parent inherits the average of the labels of its children.* We make two observations about the constructed labeling:

1. The path-cut at a higher level cannot be more than the level below. Consider two adjacent levels with the lower level ℓ having n vertices. Denote the set of odd numbers that is a subset of $[n-1]$ as I_{odd}, and the set of even number that is a subset of $[n-2]$ as I_{even}. Recall that the path-cut of the lower level is $\sum_{i=1}^{n-1} |w_i^\ell - w_{i+1}^\ell|$. This can be bounded as follows:

$$
\begin{aligned}
&\sum_{i=1}^{n-1} |w_i^\ell - w_{i+1}^\ell| \\
&= \sum_{i\in I_{odd}} \left| w_i^\ell - \frac{w_i^\ell + w_{i+1}^\ell}{2} \right| + \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - w_{i+1}^\ell \right| + \sum_{i\in I_{even}} |w_i^\ell - w_{i+1}^\ell| \\
&= \sum_{i\in I_{odd}} \left| w_i^\ell - \frac{w_i^\ell + w_{i+1}^\ell}{2} \right| + \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - w_{i+1}^\ell \right| + \sum_{i\in I_{odd}\setminus\{n-1\}} |w_{i+1}^\ell - w_{i+2}^\ell| \\
&\geq \sum_{i\in I_{odd}} \left| w_i^\ell - \frac{w_i^\ell + w_{i+1}^\ell}{2} \right| + \sum_{i\in I_{odd}\setminus\{n-1\}} \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - w_{i+1}^\ell \right| + |w_{i+1}^\ell - w_{i+2}^\ell| \\
&\geq \sum_{i\in I_{odd}} \left| w_i^\ell - \frac{w_i^\ell + w_{i+1}^\ell}{2} \right| + \sum_{i\in I_{odd}\setminus\{n-1\}} \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - w_{i+2}^\ell \right| \qquad (5.19) \\
&\geq \sum_{i\in I_{odd}\setminus\{1\}} \left| w_i^\ell - \frac{w_i^\ell + w_{i+1}^\ell}{2} \right| + \sum_{i\in I_{odd}\setminus\{n-1\}} \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - w_{i+2}^\ell \right| \\
&= \sum_{i\in I_{odd}\setminus\{n-1\}} \left| w_{i+2}^\ell - \frac{w_{i+2}^\ell + w_{i+3}^\ell}{2} \right| + \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - w_{i+2}^\ell \right| \\
&\geq \sum_{i\in I_{odd}\setminus\{n-1\}} \left| \frac{w_i^\ell + w_{i+1}^\ell}{2} - \frac{w_{i+2}^\ell + w_{i+3}^\ell}{2} \right| \qquad (5.20)
\end{aligned}
$$

where (5.19) and (5.20) follow from $|a-b|+|b-c| \geq |a-c|$ (triangle inequality). Observing that the R.H.S. of (5.20) is the path cut of the upper level, we are then done.

2. If we denote the set of edges between two adjacent levels by $\tilde{E}$, we have that $\sum_{(i,j)\in\tilde{E}}(u_i - u_j)^2$ is at most half the path-cut of the lower level. This can be seen by considering the edges between a given parent i and its two children j and j'. Let us define x as half the path cut due to the children, i.e. $\frac{1}{2}|u_j - u_{j'}|$. Since all labelings are in $[0, 1]$, we have that $x \in [0, 1/2]$. The cut made due to the parent and the children, i.e. $(u_i - u_j)^2 + (u_i - u_{j'})^2$ is then given by $2x^2$. Using the inequality $x - 2x^2 \geq 0$ for $x \in [0, 1/2]$, and applying this to all the parents on the same level as vertex i, we then prove the statement.

Combining the two above observations and recalling that there are $\lceil \log_2 T \rceil + 1$ levels (and therefore $\lceil \log_2 T \rceil$ transitions between the levels), we have that

$$\sum_{(i,j)\in E}(u_i - u_j)^2 \leq \sum_{\ell=1}^{\lceil \log_2 T \rceil} \frac{1}{2} p(\ell) \leq \sum_{k=1}^{\lceil \log_2 T \rceil} \frac{1}{2} k(f),$$

where $p(\ell)$ is the path-cut of the tree at level ℓ, the first inequality is due to observation 2, and the second inequality is due to observation 1. Hence we have shown our premise that the cut is upper bounded by $\frac{1}{2}k(f)\lceil \log_2 T \rceil$. Observe that our premise still holds if there are more than T leaf nodes, as we can treat any additional leaves on the bottom level as being labeled with the last label on their level; thus the cut will not increase. Hence we have shown the following inequality where $\boldsymbol{L}$ is the Laplacian of a fully complete binary tree with N vertices and a path of T leaves labeled by an $f \in \{0, 1\}^{[T]}$.

$$\min_{\boldsymbol{u}\in\boldsymbol{R}^{[N]}:u_i=f(i),i\in[T]} \boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u} \leq \frac{1}{2}k(f)\lceil \log_2 T \rceil . \tag{5.21}$$

We next observe that

$$\mathcal{R}_{\boldsymbol{L}} \leq \lceil \log_2 T \rceil . \tag{5.22}$$

This follows from Lemma 72, where $\mathcal{R}_{\boldsymbol{L}} = \max_{i\in[N]} L^+_{ii}$ is bounded by half the

resistance diameter, which is then just bounded by half the geodesic diameter. Furthermore if $\boldsymbol{L}$ is the Laplacian of a connected graph and $\boldsymbol{L}^\circ := \boldsymbol{L} + \left(\frac{1}{m}\right)\left(\frac{1}{m}\right)^\top \mathcal{R}_{\boldsymbol{L}}^{-1}$ then if $\boldsymbol{u} \in [-1,1]^m$ we have

$$\mathcal{R}_{\boldsymbol{L}^\circ} = 2\mathcal{R}_{\boldsymbol{L}}\,,$$
$$\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u} \le \boldsymbol{u}^\top \boldsymbol{L}\boldsymbol{u} + \frac{1}{\mathcal{R}_{\boldsymbol{L}}}\,.$$

Thus combining the above with (5.21) and (5.22), we have,

$$\mathcal{R}_{\boldsymbol{L}^\circ} \le 2\lceil \log_2 T \rceil\,,$$
$$\min_{\boldsymbol{u}\in\boldsymbol{R}^{[N]}:u_i=f(i),i\in[T]} (\boldsymbol{u}^\top \boldsymbol{L}^\circ \boldsymbol{u})\mathcal{R}_{\boldsymbol{L}^\circ} \le k(f)\lceil \log_2 T\rceil^2 + 2\,,$$

which proves the Lemma. □

Observe that the left hand side in (5.18) is up to constant factors, the normalized margin of f in the sense of Novikoff's Theorem [76]. The construction is somewhat counterintuitive as one may expect that one can use a path graph directly in the construction of the kernel. However, then $\max_{t\in[T]} P(t,t) \in \Theta(T)$ which would lead to a vacuous regret bound. Also one may wonder if one can reduce the term $(\log T)^2$ while maintaining a linear factor in $k(f)$. In fact the term $(\log T)^2$ is known [119, Theorem 6.1] to be required when $k(f) = 1$.

As a straightforward corollary to Lemma 73, we have

Corollary 74. *If $f \in \mathcal{H}_{\tilde{P}} \cap \{0,1\}^T$ then*

$$\|f\|^2_{\tilde{P}} \max_{\tau\in[T]} \tilde{P}(\tau,\tau) \le (k(f) + s(f))\lceil \log_2 T\rceil^2 + 2\,, \tag{5.23}$$

where $\tilde{P} = \tilde{P}^{\ell,T^1,\ldots,T^s}$, $k(f) := \sum_{i=1}^{s}\sum_{t=1}^{T^i-1}[f(\binom{i}{t}) \neq f(\binom{i}{t+1})]$ and $s(f) := \sum_{i=1}^{s-1}[f(\binom{i}{T^i}) \neq f(\binom{i+1}{1})]$.

Proof. Since each task is laid out contiguously along the bottom layer, we pay the path-cut for each task individually and we pay $s(f)$ for the intertask boundaries. □

Proof of Theorem 69

We first recall some of the notation introduced earlier in the section. The *block expansion* matrices are defined as $\mathcal{B}^{m,d} := \{\boldsymbol{R} \subset \{0,1\}^{m\times d} : \|\boldsymbol{R}_i\| = 1 \text{ for } i \in [m], \text{rank}(\boldsymbol{R}) = d\}$. The class of (k,ℓ)-binary-biclustered matrices is defined as $\mathbb{B}^{m,n}_{k,\ell} = \{\boldsymbol{U} = \boldsymbol{R}\boldsymbol{U}^*\boldsymbol{C}^\top \in \{-1,1\}^{m\times n} : \boldsymbol{U}^* \in \{-1,1\}^{k\times\ell}, \boldsymbol{R} \in \mathcal{B}^{m,k}, \boldsymbol{C} \in \mathcal{B}^{n,\ell}\}$. Next, we recall Theorem 69 and provide a proof.

Theorem 69. *The expected regret of Algorithm 4 with upper estimates, $k \geq k(\boldsymbol{h}^*)$, $m \geq |m(\boldsymbol{h}^*)|$,*

$$\hat{C} \geq C(\boldsymbol{h}^*) := \lambda^2 \left(\sum_{h\in m(\boldsymbol{h}^*)} \|h\|_K^2 \, X_K^2 + 2(s+k-1)m\lceil \log_2 T\rceil^2 + 2m^2 \right),$$

$\hat{X}_K^2 \geq \max_{\tau\in[T]} K(x^\tau, x^\tau)$, and learning rate $\eta = \sqrt{\frac{\hat{C}\log(2T)}{2Tm\lambda^2}}$ is bounded by

$$\sum_{i=1}^{s}\sum_{t=1}^{T^i} \mathbb{E}[\mathcal{L}_{01}(y_t^i, \hat{y}_t^i)] - \frac{1+\lambda^2}{2}\mathcal{L}_{01}(y_t^i, h_t^i(x_t^i)) \leq 3.5\sqrt{2\hat{C}\,T\log(2T)} \tag{5.24}$$

with received instance sequence $\boldsymbol{x} \in \mathcal{X}^T$ for any $\boldsymbol{h}^* \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)^T}$ where $\lambda \geq 1$.

Proof. Algorithm 4 is the same as Algorithm 3 except for some redefinitions in the notation. For convenience we recall Algorithm 3 below[4].

Algorithm 3.

Parameters: Learning rate: $0 < \eta$ quasi-dimension estimate: $1 \leq \widehat{\mathcal{D}}$, margin estimate: $0 < \gamma \leq 1$ and side-information kernels $\mathcal{M}^+ : \mathcal{I}\times\mathcal{I} \to \Re$, $\mathcal{N}^+ : \mathcal{J}\times\mathcal{J} \to \Re$, with $\mathcal{R}_\mathcal{M} := \max_{i\in\mathcal{I}} \mathcal{M}^+(i,i)$ and $\mathcal{R}_\mathcal{N} := \max_{j\in\mathcal{J}} \mathcal{N}^+(j,j)$, and maximum distinct rows m and columns n, where $m+n \geq 3$.

Initialization: $\mathbb{M} \leftarrow \emptyset, \mathbb{U} \leftarrow \emptyset, \mathcal{I}^1 \leftarrow \emptyset, \mathcal{J}^1 \leftarrow \emptyset$.

For $t = 1, \ldots, T$

- Receive pair $(i_t, j_t) \in \mathcal{I}\times\mathcal{J}$.

[4]Since we are only concerned with the regret bound, we have set the parameter **NON-CONSERVATIVE** = 1 in our restating of the algorithm.

- Define

$$(\boldsymbol{M}^t)^+ := (\mathcal{M}^+(i_r, i_s))_{r,s\in\mathcal{I}^t\cup\{i_t\}}\,; \quad (\boldsymbol{N}^t)^+ := (\mathcal{N}^+(j_r, j_s))_{r,s\in\mathcal{J}^t\cup\{j_t\}}\,,$$

$$\tilde{\boldsymbol{X}}^t(s) := \left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\boldsymbol{e}^{i_s}}{\sqrt{2\mathcal{R}_{\mathcal{M}}}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\boldsymbol{e}^{j_s}}{\sqrt{2\mathcal{R}_{\mathcal{N}}}}\right]\left[\frac{(\sqrt{(\boldsymbol{M}^t)^+})\boldsymbol{e}^{i_s}}{\sqrt{2\mathcal{R}_{\mathcal{M}}}}; \frac{(\sqrt{(\boldsymbol{N}^t)^+})\boldsymbol{e}^{j_s}}{\sqrt{2\mathcal{R}_{\mathcal{N}}}}\right]^\top,$$

$$\tilde{\boldsymbol{W}}^t \leftarrow \exp\left(\log\left(\frac{\widehat{\mathcal{D}}}{m+n}\right)\boldsymbol{I}^{|\mathcal{I}^t|+|\mathcal{J}^t|+2} + \sum_{s\in\mathbb{U}}\eta y_s\tilde{\boldsymbol{X}}^t(s)\right).$$

- Predict

$$\hat{y}_t \leftarrow \mathrm{tr}\left(\tilde{\boldsymbol{W}}^t\tilde{\boldsymbol{X}}^t(t)\right)\,;\ \bar{y}_t \leftarrow \mathrm{sign}(\hat{y}_t)\,.$$

- Receive label $y_t \in \{-1, 1\}$.
- If $y_t \neq \hat{y}_t$ then $\mathbb{M} \leftarrow \mathbb{M} \cup \{t\}$.
- If $y_t\bar{y}_t < \gamma$ then

$$\mathbb{U} \leftarrow \mathbb{U} \cup \{t\}\,,\ \ \mathcal{I}^{t+1} \leftarrow \mathcal{I}^t \cup \{i_t\},\ \text{and}\ \mathcal{J}^{t+1} \leftarrow \mathcal{J}^t \cup \{j_t\}\,.$$

- Else $\mathcal{I}^{t+1} \leftarrow \mathcal{I}^t$ and $\mathcal{J}^{t+1} \leftarrow \mathcal{J}^t$.

The following table summarizes the notational changes between the two algorithms.

Description	Algorithm 3	Algorithm 4
Row space	$\mathcal{I}$	$\mathcal{X}$
Column space	$\mathcal{J}$	$[T]$
Row kernel	$\mathcal{M}^+$	K
Column kernel	$\mathcal{N}^+$	$P := \tilde{P}^{\ell,T^1,\ldots,T^s}$
Row squared radius	$\mathcal{R}_{\mathcal{M}}$	$\hat{X}_K^2$
Column squared radius	$\mathcal{R}_{\mathcal{N}}$	$\hat{X}_P^2$
Margin estimate	γ^{-2}	$m\lambda^2$
Complexity Estimate	$\widehat{\mathcal{D}}\gamma^{-2}$	$\hat{C}$
Dimensions [5]	m, n	T, T
Time	t	τ
Instance	(i_t, j_t)	(x^τ, τ)

We now recall the following regret bound for Algorithm 1, which also holds for Algorithm 3, due to Proposition 6.

[5]Note T is an upper bound known in advance for the number of rows. We will use p to denote the number of distinct x values seen over the trials of the algorithm.

Theorem 1 *The expected mistakes of Algorithm 1, parameters* $\widehat{\mathcal{D}} \geq \mathcal{D}^{\gamma}_{M,N}(\boldsymbol{U})$*,* $\eta = \sqrt{\frac{\widehat{\mathcal{D}}\log(m+n)}{2T}}$*, are bounded by*

$$\mathbb{E}[|\mathbb{M}|] \leq \frac{\left(1 + \max_{i,j}|U_{ij}|\right)}{2} \sum_{t\in\mathbb{M}} [y_t \neq \operatorname{sign}(U_{i_t j_t})] + \frac{3.5}{\gamma}\sqrt{\widehat{\mathcal{D}}\log(m+n)T} \tag{5.25}$$

for all $\boldsymbol{U} \in ([-\infty,-1]\cup[1,\infty])^{m\times n}$ *with* $\|\boldsymbol{U}\|_{max} \leq 1/\gamma$*.*

We introduce the following notation: the set $\mathcal{X}^{\text{fin}} := \cup_{\tau\in[T]}\{x^\tau\}$, the matrix $\boldsymbol{H} := (h^\tau(x) : x \in \mathcal{X}^{\text{fin}}, \tau \in [T])$, $\boldsymbol{H}^* := (h(x))_{x\in\mathcal{X}^{\text{fin}}, h\in m(\boldsymbol{h}^*)}$ and $\boldsymbol{C} := ([h^\tau = h])_{\tau\in[T], h\in m(\boldsymbol{h}^*)}$ (note $\boldsymbol{C} \in \mathcal{B}^{T,m}$), $\boldsymbol{D} \in \mathfrak{R}^{p\times p}$ is a diagonal matrix with entries $D_{ii} = \frac{1}{\gamma\|\boldsymbol{H}_i^*\|}$ for $i \in [p]$, $\boldsymbol{U} := \boldsymbol{D}\boldsymbol{H}$, the matrices $\bar{\boldsymbol{K}} = [(K(x,x') : x,x' \in \mathcal{X}^{\text{fin}})]^{-1}$, $\bar{\boldsymbol{P}} = [(\tilde{P}(\tau,\upsilon) : \tau,\upsilon \in [T])]^{-1}$, and $\lambda := \max_{i,j}|\boldsymbol{H}_{ij}|$. Observe that $\boldsymbol{H} = \boldsymbol{H}^*\boldsymbol{C}^\top$.

Initially we note that we very trivially extend the algorithm and thus its analysis in so far as we use the upper bounds and $\hat{X}_K^2 \geq \mathcal{R}_{\bar{\boldsymbol{K}}}$ and $\hat{X}_P^2 \geq \mathcal{R}_{\bar{\boldsymbol{P}}}$.

It now remains that in order to complete the reduction of Theorem 69 to Theorem 3, we need to demonstrate the following four inequalities:

$$\|\boldsymbol{U}\|_{\max} \leq \sqrt{m\lambda^2} \tag{5.26}$$

$$\max_{i,j}|\boldsymbol{U}_{ij}| \leq \lambda^2 \tag{5.27}$$

$$1 \leq \min_{i,j}|\boldsymbol{U}_{ij}| \tag{5.28}$$

$$\mathcal{D}^{1/\sqrt{m\lambda^2}}_{\bar{\boldsymbol{K}},\bar{\boldsymbol{P}}}(\boldsymbol{U}) \leq \frac{1}{m\lambda^2} C(\boldsymbol{h}^*)\,. \tag{5.29}$$

First we show (5.26). Recalling the definition of the max-norm, we have

$$\|\boldsymbol{U}\|_{\max} = \min_{\boldsymbol{P}\boldsymbol{Q}^\top=\boldsymbol{U}} \left(\max_i \|\boldsymbol{P}_i\| \max_j \|\boldsymbol{Q}_j\|\right)$$

Set $\boldsymbol{P}' := \boldsymbol{D}\boldsymbol{H}^*$, and $\boldsymbol{Q}' := \boldsymbol{C}$ and observe that $(\boldsymbol{P}', \boldsymbol{Q}')$ is in the feasible set of the minimization, giving

$$\|\boldsymbol{U}\|_{\max} \leq \max_{i\in[|\mathcal{X}^{\text{fin}}|]} \|(\boldsymbol{D}\boldsymbol{H}^*)_i\| \max_{j\in[T]} \|\boldsymbol{C}_j\|$$

$$
\begin{aligned}
&= \max_{i\in[|\mathcal{X}^{\text{fin}}|]} \left(\frac{1}{\gamma\|\boldsymbol{H}_i^*\|}\|\boldsymbol{H}_i^*\|\right) \max_{j\in[T]} \|\boldsymbol{C}_j\| \\
&= \frac{1}{\gamma} \\
&= \sqrt{m\lambda^2}
\end{aligned}
$$

Next we show (5.27) and (5.28). Defining $\boldsymbol{U}^* := \boldsymbol{D}\boldsymbol{H}^*$, we have $\boldsymbol{U} = \boldsymbol{U}^*\boldsymbol{C}^\top$, so that all the entries that appear in $\boldsymbol{U}$ must also exist in $\boldsymbol{U}^*$ and vice versa. We therefore have

$$
\begin{aligned}
\max_{i,j} |\boldsymbol{U}_{ij}| &= \max_{i,j} |\boldsymbol{U}^*_{ij}| \\
&= \max_{i,j} |(\boldsymbol{D}\boldsymbol{H}^*)_{ij}| \\
&= \max_{i,j} \frac{|\boldsymbol{H}^*_{ij}|}{\gamma\|\boldsymbol{H}_i^*\|}
\end{aligned}
$$

Using $\frac{1}{\gamma} = \sqrt{m\lambda^2}$, $|\boldsymbol{H}^*_{ij}| \leq \lambda$, and $\frac{1}{\|\boldsymbol{H}_i^*\|} \leq \frac{1}{\sqrt{m}}$ for all i and j, then gives (5.27). Similarly, we have

$$
\min_{i,j} |\boldsymbol{U}_{ij}| = \min_{i,j} \frac{|\boldsymbol{H}^*_{ij}|}{\gamma\|\boldsymbol{H}_i^*\|}
$$

Using $\frac{1}{\gamma} = \sqrt{m\lambda^2}$, $|\boldsymbol{H}^*_{ij}| \geq 1$ and $\frac{1}{\|\boldsymbol{H}_i^*\|} \geq \frac{1}{\sqrt{m\lambda^2}}$ for all i and j gives (5.28).

We now show (5.29). We recall the following useful equality (see e.g. [82, Proposition 12.32]),

$$
\boldsymbol{u}^\top \boldsymbol{K}^{-1}\boldsymbol{u} = \min_{f\in\mathcal{H}_K : f(x)=u_x : x\in X} \|f\|_K^2 \ . \tag{5.30}
$$

where $\boldsymbol{K} = (K(x,x'))_{x,x'\in X}$, $\boldsymbol{u} \in \mathfrak{R}^X$ and $\boldsymbol{K}$ is invertible and K is a kernel. By Theorem 71 we have

$$
\mathcal{D}^{1/\sqrt{m\lambda^2}}_{\bar{\boldsymbol{K}},\bar{\boldsymbol{P}}}(\boldsymbol{U}) \leq \frac{1}{m}\operatorname{tr}((\boldsymbol{H}^*)^\top \bar{\boldsymbol{K}}\boldsymbol{H}^*)\hat{X}_K^2 + \operatorname{tr}(\boldsymbol{C}^\top\bar{\boldsymbol{P}}\boldsymbol{C})\hat{X}_P^2
$$

where we recall that $\boldsymbol{H} = \boldsymbol{H}^*\boldsymbol{C}^\top$ with $\boldsymbol{H}^* := (h(x))_{x\in\mathcal{X}^{\text{fin}}, h\in m(\boldsymbol{h}^*)}$ and $\boldsymbol{C} := ([h^\tau = h])_{\tau\in[T], h\in m(\boldsymbol{h}^*)}$ (note $\boldsymbol{C} \in \mathcal{B}^{T,m}$).

Simplifying and using (5.30) we have,

$$\mathcal{D}^{1/\sqrt{m\lambda^2}}_{\bar{\boldsymbol{K}},\bar{\boldsymbol{P}}}(\boldsymbol{U}) \leq \frac{1}{m}\sum_{h\in m(\boldsymbol{h}^*)} \|h\|^2_K \hat{X}^2_K + \mathrm{tr}(\boldsymbol{C}^\top \bar{\boldsymbol{P}}\boldsymbol{C})\hat{X}^2_P\,. \tag{5.31}$$

From (5.30) we have,

$$\mathrm{tr}(\boldsymbol{C}^\top \bar{\boldsymbol{P}}\,\boldsymbol{C}) = \sum_{h\in m(\boldsymbol{h}^*)} \boldsymbol{c}_h^\top \bar{\boldsymbol{P}}\boldsymbol{c}_h = \sum_{h\in m(\boldsymbol{h}^*)} \|f_h\|^2_{\tilde{P}} \tag{5.32}$$

where $\boldsymbol{c}_h$ is the column vector formed by taking the h^{th} column of $\boldsymbol{C}$. The vector $\boldsymbol{c}_h \in \{0,1\}^T$ indicates if hypothesis h is "active" on trial τ, i.e., $c^\tau_h = [h^\tau = h]$. Next we define $f_h(\tau) := c^\tau_h$ for $\tau = 1,\cdots,T$. Recalling $\tau \equiv {}^{\ell^\tau}_{\sigma(\tau)}$, we also have $f_h(\tau) \equiv f_h({}^{\ell^\tau}_{\sigma(\tau)})$.

From (5.32) and Corollary 74 we have,

$$\begin{aligned}
\mathrm{tr}(\boldsymbol{C}^\top \bar{\boldsymbol{P}}\boldsymbol{C})\hat{X}^2_P &= \sum_{h\in m(\boldsymbol{h}^*)} \|f_h\|_{\tilde{P}}\,\hat{X}^2_P && (5.33)\\
&\leq \sum_{h\in m(\boldsymbol{h}^*)} \Big(k(f_h) + s(f_h))\lceil\log_2 T\rceil^2 + 2\Big)\\
&\leq \sum_{h\in m(\boldsymbol{h}^*)} (k(f_h) + s(f_h))\lceil\log_2 T\rceil^2 + 2m(\boldsymbol{h}^*)\\
&\leq \sum_{h\in m(\boldsymbol{h}^*)} (k(f_h) + s(f_h))\lceil\log_2 T\rceil^2 + 2m\\
&\leq 2(s+k-1)\lceil\log_2 T\rceil^2 + 2m && (5.34)
\end{aligned}$$

where

$$k(f) = \sum_{i=1}^{s}\sum_{t=1}^{T^i-1}[f({}^i_t) \neq f({}^i_{t+1})]\,,\; s(f) = \sum_{i=1}^{s-1}[f({}^i_{T^i}) \neq f({}^{i+1}_1)]\,,$$

and where (5.34) comes from using $\sum_{h\in m(\boldsymbol{h}^*)} k(f_h) = k(\boldsymbol{h}^*) \leq 2k$ and $\sum_{h\in m(\boldsymbol{h}^*)} s(h) \leq 2(s-1)$, where the factors of two are due to each switch of f_h on successive time steps as well as intertask boundaries being counted twice.

Substituting (5.34) into (5.31), we have

$$\mathcal{D}^{1/\sqrt{m\lambda^2}}_{\bar{\boldsymbol{K}},\bar{\boldsymbol{P}}}(\boldsymbol{U}) \leq \frac{1}{m} \sum_{h \in m(\boldsymbol{h}^*)} \|h\|^2_K \hat{X}^2_K + 2(s+k-1)\lceil \log_2 T \rceil^2 + 2m\,,$$

This demonstrates (5.29) thus completing the reduction. □

5.6.2 Proof Sketch of Proposition 70

First we recall and then give a proof sketch of Proposition 70.

Proposition 70. *For any (randomized) algorithm and any $s, k, m, \Gamma \in \mathbb{N}$, with $k + s \geq m > 1$ and $\Gamma \geq m \log_2 m$, there exists a kernel K and a $T_0 \in \mathbb{N}$ such that for every $T \geq T_0$:*

$$\sum_{\tau=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y^\tau, \hat{y}^\tau)] - \mathcal{L}_{01}(y^\tau, h^\tau(x^\tau)) \in \Omega\left(\sqrt{(\Gamma + s\log m + k \log m)\, T}\right),$$

for some multitask sequence $(x^1, y^1), \ldots, (x^T, y^T) \in (\mathcal{X} \times \{-1, 1\})^T$ and some $\boldsymbol{h}^* \in [\mathcal{H}_K^{(\boldsymbol{x},1)}]^T$ such that $m \geq |m(\boldsymbol{h}^*)|$, $k \geq k(\boldsymbol{h}^*)$, $\sum_{h \in m(\boldsymbol{h}^*)} \|h\|^2_K X^2_K \geq |m(\boldsymbol{h}^*)| \log_2 m$, where $X^2_K = \max_{\tau \in [T]} K(x^\tau, x^\tau)$.

Proof Sketch. We recall the following online learning terminology. A sequence of examples $(x_1, y_1), \ldots, (x_T, y_T)$ is *realizable* with respect to a hypothesis class $\mathcal{H}$ if there exists an $h \in \mathcal{H}$, such that $\sum_{t=1}^{T} \mathcal{L}_{01}(y_t, h(x_t)) = 0$. The *optimal mistake bound* (Ldim($\mathcal{H}$)) with respect to a hypothesis class $\mathcal{H}$ also known as the *Littlestone dimension* [16, 120] is, informally speaking, the minimum over all deterministic learning algorithms, of the maximum over all realizable example sequences of the number of mistaken predictions.

We will apply the following useful result [120, Lemma 14] which we quote below for convenience,

Lemma 14 (Lower Bound). *Let $\mathcal{H}$ be any hypothesis class with a finite Ldim($\mathcal{H}$). For any (possibly randomized) algorithm, there exists a*

sequence $(x_1, y_1), \ldots, (x_T, y_T)$ *such that*

$$\mathbb{E} \sum_{t=1}^{T} \mathcal{L}_{01}(y_t, \hat{y}_t) - \min_{h \in \mathcal{H}} \mathcal{L}_{01}(y_t, h(x_t)) \geq \sqrt{\frac{\mathrm{Ldim}(\mathcal{H})T}{8}} .$$

In essence, this allows one to go from a lower bound on mistakes in the *realizable* case to a lower bound in the non-realizable case. However, Lemma 14 only applies directly to the standard single-task model. To circumvent this, we recall as discussed in Section 5.2, that the switching multitask model may be reduced to the single-task model with a domain $\mathcal{X}' = \mathcal{X} \times [T] \times [s]$ and hypothesis class $\mathcal{H}'$. Therefore a lower bound in the switching multitask model with respect to $\mathcal{H}$ implies a lower bound in the single-task non-switching case for $\mathcal{H}'$ via the reduction. There are some slight technical issues over the fact that "time" is now part of the domain $\mathcal{X}'$ and thus e.g., valid example sequences cannot be permuted. We gloss over these issues in this proof sketch noting that they do not in fact impact our arguments. The argument proceeds by demonstrating that there exists for any $s, k, m, \Gamma \in \mathbb{N}$, a kernel K and a realizable multitask sequence $(x^1, y^1), \ldots, (x^T, y^T)$ for which

$$\sum_{\tau=1}^{T} \mathcal{L}_{01}(y^\tau, \hat{y}^\tau) \in \Omega\left(\Gamma + s \log m + k \log m\right) , \tag{5.35}$$

where $\boldsymbol{x} \in \mathcal{X}^T$, $X_K^2 = \max_{\tau \in [T]} K(x^\tau, x^\tau)$, $\Gamma \geq \sum_{h \in m(\boldsymbol{h}^*)} \|h\|_K^2 X_K^2 \geq m \log_2 m$, $k \geq k(\boldsymbol{h}^*)$, $m \geq |m(\boldsymbol{h}^*)|$ and $k + s \geq m > 1$. After demonstrating that there exists such an example sequence we can apply [120, Lemma 14] to demonstrate the proposition. Since the lower bound is in the form $\Omega(P+Q+R)$ which is equivalent to $\Omega(\max(P, Q, R))$, we may treat P, Q and R, independently to prove the bound. Before we treat the individual cases, we give a straightforward result for a simplistic hypothesis class.

Define $\mathcal{X}_d := [d]$ and $\mathcal{H}_d := \{-1, 1\}^d$ (i.e., the set of functions that map $[d] \to \{-1, 1\}$). Observe that $\mathrm{Ldim}(\mathcal{H}_d) = d$, as an algorithm can force a mistake for every component and then no more. Also, observe that if we define a kernel $K_d(x, x') := 2[x = x'] - 1$ over the domain $\mathcal{X}_d$ that $\mathcal{H}_d = \mathcal{H}_{K_d}^{([d])}$, $\max_{x \in [d]} K_d(x, x) = 1$ and that

$\|h\|_{K_d}^2 = d$ for all $h \in \mathcal{H}_d$. Finally, note that if $m = |\mathcal{H}_d|$ then $\sum_{h\in\mathcal{H}_d} \|h\|_{K_d}^2 X_{K_d}^2 = m \log_2 m$.

We proceed by sketching an adversary for each of the three cases.

1. Case Γ is the max.

   To force Γ mistakes, we choose $K = K_d$ and set $d = \Gamma/m$ and without loss of generality assume that d is an integer and recall that $k + s \geq m$. Since $\mathrm{Ldim}(\mathcal{H}_d) = d$, an adversary may force d mistakes within a single task in the first d trials. This strategy may be repeated k more times within a single task thus forcing $(k + 1)d$ mistakes. If $k + 1 \geq m$, we are done. Otherwise, the constraint $k + s \geq m$ implies that we may force d mistakes per task in $m - (k + 1)$ other tasks. Thus after md trials, $md = \Gamma$ mistakes have been forced while maintaining the condition $m \geq |m(\boldsymbol{h}^*)|$.

2. Case $k \log_2 m$ is the max.

   Set $d = \log_2 m$ and without loss of generality assume d is positive integer. Using $\mathcal{H}_d$ we force kd mistakes by first forcing d mistakes within a single task then "switching" $k - 1$ times forcing $kd = k \log_2 m$ mistakes, while maintaining the conditions $m \geq |m(\boldsymbol{h}^*)|$ and $k \geq k(\boldsymbol{h}^*)$.

3. Case $s \log_2 m$ is the max.

   Same instance as the above case, except we force d mistakes per task.

□

5.6.3 Proofs and Details for Section 5.2

For the reader's convenience, we collect some standard well-known online learning results or minor extensions thereof in this subsection.

Details for MW Bound

The algorithm and analysis corresponds essentially to the classic weighted majority algorithm introduced in [19]. We first define some of the notation to be used in the section. We define the component-wise multiplication as $\boldsymbol{x} \odot \boldsymbol{w} := (x_1 w_1, \ldots, x_n w_n)$.

If $f : \mathfrak{R} \to \mathfrak{R}$ and $\boldsymbol{x} \in \mathfrak{R}^n$ then $f(\boldsymbol{x}) := (f(x_1), \ldots, f(x_n))$. We denote the probability simplex as $\Delta_{\mathcal{H}} := \{h \in [0,1]^{\mathcal{H}}\} \cap \{h : \sum_{h \in \mathcal{H}} = 1\}$ and set $\Delta_n := \Delta_{[n]}$. If $\boldsymbol{v} \in \Delta_{\mathcal{H}}$ then $h \sim \boldsymbol{v}$ denotes that h is a random sample from the probability vector $\boldsymbol{v}$ over the set $\mathcal{H}$. We denote $|\mathcal{H}_{\text{fin}}|$ as n. We now introduce the MW algorithm and give the corresponding regret.

Algorithm 5 MW Algorithm

Parameters: Learning rate η; finite hypothesis set $\{h^1, \ldots, h^n\} = \mathcal{H}_{\text{fin}} \subset \{-1,1\}^{\mathcal{X}}$
Initialization: Initialize $\boldsymbol{v}_1 = \frac{1}{n}\mathbf{1}^n$
For $t = 1, \ldots, T$

- Receive instance $\boldsymbol{x}_t \in \mathcal{X}$.
- Set $\boldsymbol{h}_t = (h^1(\boldsymbol{x}_t) \ldots h^n(\boldsymbol{x}_t)) \in \{-1,1\}^n$.
- Predict

$$i_t \sim v_t \,;\; \hat{y}_t \leftarrow h_t^{i_t} \,.$$

- Receive label $y_t \in \{-1,1\}$.
- Update

$$\begin{aligned} \boldsymbol{\ell}_t &\leftarrow \frac{1}{2}|\boldsymbol{h}_t - \hat{y}_t\mathbf{1}| \\ \boldsymbol{w}_{t+1} &\leftarrow \boldsymbol{w}_t \odot \exp(-\eta\boldsymbol{\ell}_t) \\ \boldsymbol{v}_{t+1} &\leftarrow \frac{\boldsymbol{w}_{t+1}}{\sum_{i=1}^n w_{t+1,i}} \end{aligned}$$

Theorem 75. *For Algorithm 5, setting* $\eta = \sqrt{(2\log n)/T}$

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \mathcal{L}_{01}(y_t, h(x_t)) \leq \sqrt{2\log(n)T} \tag{5.36}$$

for any $h \in \mathcal{H}_{\text{fin}}$.

Proof. Recalling that $\boldsymbol{\ell}_t = \frac{|\boldsymbol{h}_t - \hat{y}_t\mathbf{1}|}{2}$, we have that $\boldsymbol{v}_t \cdot \boldsymbol{\ell}_t = \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)]$ and that $\boldsymbol{e}^i \cdot \boldsymbol{\ell}_t = \mathcal{L}_{01}(y_t, h^i(x_t))$. In what follows, we will therefore bound $\boldsymbol{v}_t \cdot \boldsymbol{\ell}_t - \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t$. We first prove the following "progress versus regret" inequality.

$$\boldsymbol{v}_t \cdot \boldsymbol{\ell}_t - \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t \leq \frac{1}{\eta}\left(d(\boldsymbol{e}^i, \boldsymbol{v}_t) - d(\boldsymbol{e}^i, \boldsymbol{v}_{t+1})\right) + \frac{\eta}{2}\sum_{i=1}^{n} v_{t,i}\ell_{t,i}^2. \tag{5.37}$$

where $d(\boldsymbol{u}, \boldsymbol{v}) := \sum_{i=1}^n u_i \log(\frac{u_i}{v_i})$ is the relative entropy for $\boldsymbol{u}, \boldsymbol{v} \in \Delta_n$. Let $Z_t :=$

$\sum_{i=1}^{n} v_{t,i} \exp(-\eta \ell_{t,i})$. Observe that from the algorithm

$$\begin{aligned} d(\boldsymbol{e}^i, \boldsymbol{v}_t) - d(\boldsymbol{e}^i, \boldsymbol{v}_{t+1}) &= \sum_{j=1}^{n} e^i_j \log \frac{v_{t+1,j}}{v_{t,j}} \\ &= -\eta \sum_{j=1}^{n} e^i_j \ell_{t,j} - \log Z_t \\ &= -\eta \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t - \log \sum_{i=1}^{n} v_{t,i} \exp(-\eta \ell_{t,i}) \\ &\geq -\eta \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t - \log \sum_{i=1}^{n} v_{t,i} (1 - \eta \ell_{t,i} + \frac{1}{2} \eta^2 \ell^2_{t,i}) \qquad (5.38) \\ &= -\eta \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t - \log (1 - \eta \boldsymbol{v}_t \cdot \boldsymbol{\ell}_t + \frac{1}{2} \eta^2 \sum_{i=1}^{n} v_{t,i} \ell^2_{t,i}) \\ &\geq \eta (\boldsymbol{v}_t \cdot \boldsymbol{\ell}_t - \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t) - \frac{1}{2} \eta^2 \sum_{i=1}^{n} v_{t,i} \ell^2_{t,i} \qquad (5.39) \end{aligned}$$

using inequalities $e^{-x} \leq 1 - x + \frac{x^2}{2}$ for $x \geq 0$ and $\log(1 + x) \leq x$ for (5.38) and (5.39) respectively.

Summing over t and rearranging we have

$$\begin{aligned} \sum_{t=1}^{m} \left(\boldsymbol{v}_t \cdot \boldsymbol{\ell}_t - \boldsymbol{e}^i \cdot \boldsymbol{\ell}_t \right) &\leq \frac{1}{\eta} \left(d(\boldsymbol{e}^i, \boldsymbol{v}_1) - d(\boldsymbol{e}^i, \boldsymbol{v}_{m+1}) \right) + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{n} v_{t,i} \ell^2_{t,i} \\ &\leq \frac{\log n}{\eta} + \frac{\eta}{2} \sum_{t=1}^{T} \sum_{i=1}^{n} v_{t,i} \ell^2_{t,i} \qquad (5.40) \end{aligned}$$

where (5.40) comes from noting that $d(\boldsymbol{u}, \boldsymbol{v}_1) \leq \log n$, $-d(\boldsymbol{u}, \boldsymbol{v}_{m+1}) \leq 0$, and $\sum_{t=1}^{T} \sum_{i=1}^{n} v_{t,i} \ell^2_{t,i} \leq T$. Finally we substitute the value of η and obtain the theorem. □

Proof for Online Gradient Descent Regret Bound

In this section, we will prove expected regret bounds for online gradient descent [76] for both the switching and non-switching cases. The proofs are adapted from the material in [12, 89, 85, 121]. Recall that we wish to prove the following for the non-switching case:

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \frac{1 + \lambda}{2} \mathcal{L}_{01}(y_t, \text{sign}(h(x_t))) \in O\left(\sqrt{\|h\|^2_K X^2_K T} \right) \qquad (5.41)$$

for all $h \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ and $X_K^2 := \max_{t\in[T]} K(x_t, x_t)$. For the switching case, we wish to prove

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \frac{1+\lambda}{2}\mathcal{L}_{01}(y_t, \operatorname{sign}(h(x_t))) \in O\left(\sqrt{k \max_t \|h_t\|_K^2 X_K^2 T}\right) \quad (5.42)$$

for all $h \in \mathcal{H}_K^{(\boldsymbol{x},\lambda)}$ and $X_K^2 := \max_{t\in[T]} K(x_t, x_t)$. For simplicity, we prove for an arbitrary inner product $\langle\cdot,\cdot\rangle$ space with induced norm $\|\cdot\|$. The RKHS setting reduces to this setting by identifying $\boldsymbol{x} := K(x,\cdot)$, $\boldsymbol{u} := h$, and $\langle \boldsymbol{u}, \boldsymbol{x}\rangle := h(x)$.

Algorithm 6 Randomized Constrained Online Gradient Descent Algorithm

Parameters: Learning rate η, radius R

Initialization: Initialize $\boldsymbol{w}_1 = \boldsymbol{0}$

For $t = 1, \ldots, T$

- Receive vector $\boldsymbol{x}_t \in \mathfrak{R}^d$.
- Predict

$$Y_t \sim \textsc{Uniform}(-1, 1)\,;\ \bar{y}_t \leftarrow \langle \boldsymbol{w}_t, \boldsymbol{x}_t\rangle\,;\ \hat{y}_t \leftarrow \operatorname{sign}(\bar{y}_t - Y_t)\,.$$

- Receive label $y_t \in \{-1, 1\}$.
- If $\bar{y}_t y_t \le 1$ then

$$\boldsymbol{w}_t^m \leftarrow \boldsymbol{w}_t + \eta y_t \boldsymbol{x}_t$$
$$\boldsymbol{w}_{t+1} \leftarrow P_R(\boldsymbol{w}_t^m)$$

- Else $\boldsymbol{w}_t^m \leftarrow \boldsymbol{w}_t\,;\ \ \boldsymbol{w}_{t+1} \leftarrow \boldsymbol{w}_t$

In the following, we define the hinge loss (with margin 1) $\mathcal{L}_{\text{hi}}(y_1, y_2) = [1 - y_1 y_2]_+$ for $y_1, y_2 \in \mathfrak{R}$. We define $\boldsymbol{z}_t := -y_t \boldsymbol{x}_t [1 - y_t \langle \boldsymbol{w}_t, \boldsymbol{x}_t\rangle \ge 0] \in \nabla_{\boldsymbol{w}} \mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{w}, \boldsymbol{x}_t\rangle)$, where $\boldsymbol{w}_t$, $\boldsymbol{x}_t$ and y_t are as defined in Algorithm 6. We define $P_R(\boldsymbol{w})$ to be the projection into the ball with radius R, so that

$$P_R(\boldsymbol{w}) = \begin{cases} \boldsymbol{w} & \text{if } \|\boldsymbol{w}\| \le R \\ R\frac{\boldsymbol{w}}{\|\boldsymbol{w}\|} & \text{otherwise}\,. \end{cases}$$

We also present a lemma, used as a starting point for both the switching and non-switching proofs.

Lemma 76. *For Algorithm 6 and any $\boldsymbol{u}$ lying in the convex set $\{\boldsymbol{u} : \|\boldsymbol{u}\| \leq R\}$,*

$$\langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{z}_t \rangle \leq \frac{1}{2\eta} \left(\|\boldsymbol{w}_t - \boldsymbol{u}\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}\|^2 + \eta^2 \|\boldsymbol{z}_t\|^2 \right)$$

Proof. Using the update rule of the algorithm, we have

$$\begin{aligned} \left\| \boldsymbol{w}_t^m - \boldsymbol{u} \right\|^2 &= \|\boldsymbol{w}_t - \eta \boldsymbol{z}_t - \boldsymbol{u}\|^2 \\ &= \|\boldsymbol{w}_t - \boldsymbol{u}\|^2 - 2\eta \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{z}_t \rangle + \eta^2 \|\boldsymbol{z}_t\|^2 \end{aligned}$$

Next note that $\|\boldsymbol{w}_{t+1} - \boldsymbol{u}\|^2 \leq \left\| \boldsymbol{w}_t^m - \boldsymbol{u} \right\|^2$ since $\boldsymbol{w}_{t+1}$ and $\boldsymbol{u}$ both live in the convex set $\{\boldsymbol{u} : \|\boldsymbol{u}\| \leq R\}$, giving

$$\|\boldsymbol{w}_{t+1} - \boldsymbol{u}\|^2 \leq \|\boldsymbol{w}_t - \boldsymbol{u}\|^2 - 2\eta \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{z}_t \rangle + \eta^2 \|\boldsymbol{z}_t\|^2 .$$

Rearranging then results in the lemma. □

Non-switching bound

Lemma 77. *For Algorithm 6, given $X = \max_t \|\boldsymbol{x}_t\|$, $\|\boldsymbol{u}\| \leq U$ and $\eta = \frac{U}{X\sqrt{T}}$ we have that*

$$\sum_{t=1}^{T} \mathcal{L}_{hi}(y_t, \bar{y}_t) - \mathcal{L}_{hi}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t \rangle) \leq \sqrt{U^2 X^2 T}$$

for any vector $\boldsymbol{u}$.

Proof. Using the convexity of the hinge loss (with respect to its second argument), we have

$$\mathcal{L}_{\text{hi}}(y_t, \bar{y}_t) - \mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t \rangle) \leq \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{z}_t \rangle.$$

We may therefore proceed by bounding $\sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{z}_t \rangle$. Starting with Lemma 76 and summing over t, we have

$$\begin{aligned} \sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}, \boldsymbol{z}_t \rangle &\leq \frac{1}{2\eta} \left(\|\boldsymbol{w}_1 - \boldsymbol{u}\|^2 - \|\boldsymbol{w}_{T+1} - \boldsymbol{u}\|^2 + \eta^2 \sum_{t=1}^{T} \|\boldsymbol{z}_t\|^2 \right) \\ &\leq \frac{1}{2\eta} \left(\|\boldsymbol{u}\|^2 + \eta^2 \sum_{t=1}^{T} \|\boldsymbol{z}_t\|^2 \right) \end{aligned} \tag{5.43}$$

$$
\begin{aligned}
&= \frac{1}{2\eta}\|\boldsymbol{u}\|^2 + \frac{\eta}{2}\sum_{t=1}^{T}\|\boldsymbol{x}_t\|^2\,[1 - y_t\langle \boldsymbol{w}_t, \boldsymbol{x}_t\rangle \geq 0] \\
&\leq \frac{1}{2\eta}\|\boldsymbol{u}\|^2 + \frac{\eta}{2}\sum_{t=1}^{T}\|\boldsymbol{x}_t\|^2 \\
&\leq \frac{1}{2\eta}U^2 + \frac{\eta}{2}X^2T \\
&= \sqrt{U^2X^2T}
\end{aligned}
$$

where Equation (5.43) results from the definition of $\boldsymbol{w}_1$. □

Theorem 78. *For Algorithm 6, given* $X = \max_t \|\boldsymbol{x}_t\|$, $\|\boldsymbol{u}\| \leq U$, $\eta = \frac{U}{X\sqrt{T}}$,

$$
\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \frac{1+\lambda}{2}\,\mathcal{L}_{01}(y_t, \operatorname{sign}(\langle \boldsymbol{u}_t, \boldsymbol{x}_t\rangle)) \leq \frac{1}{2}\sqrt{U^2X^2T}\,,
$$

for any vector $\boldsymbol{u}$ such that $1 \leq |\langle \boldsymbol{u}, \boldsymbol{x}_t\rangle| \leq \lambda$ for $t = 1, \ldots, T$.

Proof. The bound follows from Lemma 77, where we bound the hinge loss terms as follows:

1. $\sum_{t=1}^{T}\mathcal{L}_{\text{hi}}(y_t, \bar{y}_t) \geq 2\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)]$ (see e.g. Lemma 23)

2. $\mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t\rangle) \leq (1+\lambda)\,\mathcal{L}_{01}(y_t, \operatorname{sign}(\langle \boldsymbol{u}_t, \boldsymbol{x}_t\rangle))$.

The second inequality can be proven as follows. We recall that $\mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t\rangle) = [1 - y_t\langle \boldsymbol{u}, \boldsymbol{x}_t\rangle]_+$. In the case that $y_t\langle \boldsymbol{u}, \boldsymbol{x}_t\rangle > 0$, we have that $\mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t\rangle) = 0$ since $|\langle \boldsymbol{u}, \boldsymbol{x}_t\rangle| \geq 1$. Otherwise, we have $\mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t\rangle) = [1 + |\langle \boldsymbol{u}, \boldsymbol{x}_t\rangle|]_+ \leq 1 + \lambda$. Combining the two cases, we have $\mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t\rangle) \leq (1+\lambda)\,\mathcal{L}_{01}(y_t, \operatorname{sign}(\langle \boldsymbol{u}_t, \boldsymbol{x}_t\rangle))$.

□

The bound for the non-switching case in (5.41) then follows by setting $U = \|\boldsymbol{u}\|$.

Switching bound

Lemma 79. *For Algorithm 6, given* $X = \max_t \|\boldsymbol{x}_t\|$, $\{\boldsymbol{u}_1, \dots \boldsymbol{u}_T\} \subset \{\boldsymbol{u} : \|\boldsymbol{u}\| \leq R\}$, $\eta = \frac{U}{X\sqrt{T}}$ *and* $\sqrt{\|\boldsymbol{u}_T\|^2 + 2R\sum_{t=1}^{T-1} \|\boldsymbol{u}_{t+1} - \boldsymbol{u}_t\|} \leq U$, *we have that*

$$\sum_{t=1}^{T} \mathcal{L}_{hi}(y_t, \bar{y}_t) - \mathcal{L}_{hi}(y_t, \langle \boldsymbol{u}_t, \boldsymbol{x}_t \rangle) \leq \sqrt{U^2 X^2 T}.$$

Proof. Using the convexity of the hinge loss (with respect to its second argument), we have

$$\mathcal{L}_{\text{hi}}(y_t, \bar{y}_t) - \mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}_t, \boldsymbol{x}_t \rangle) \leq \langle \boldsymbol{w}_t - \boldsymbol{u}_t, \boldsymbol{z}_t \rangle.$$

We may therefore proceed by bounding $\sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}_t, \boldsymbol{z}_t \rangle$. Starting with Lemma 76 and summing over t, we have

$$\sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}_t, \boldsymbol{z}_t \rangle \leq \frac{1}{2\eta} \sum_{t=1}^{T} \left(\|\boldsymbol{w}_t - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 + \eta^2 \|\boldsymbol{z}_t\|^2 \right) \tag{5.44}$$

To transform the right hand side of the above equation into a telescoping sum, we add and subtract the term $A_t = \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{t+1}\|^2$, giving

$$\begin{aligned}
\sum_{t=1}^{T} \|\boldsymbol{w}_t - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 &= \sum_{t=1}^{T} \|\boldsymbol{w}_t - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{t+1}\|^2 - A_t \\
&= \|\boldsymbol{u}_1\|^2 - \|\boldsymbol{w}_{T+1} - \boldsymbol{u}_{T+1}\|^2 - \sum_{t=1}^{T} (\|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{t+1}\|^2) \\
&= \|\boldsymbol{u}_1\|^2 - \|\boldsymbol{w}_{T+1} - \boldsymbol{u}_T\|^2 - \sum_{t=1}^{T-1} (\|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{t+1}\|^2) \qquad (5.45) \\
&\leq \|\boldsymbol{u}_1\|^2 - \sum_{t=1}^{T-1} (\|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{t+1}\|^2), \qquad (5.46)
\end{aligned}$$

where Equation (5.45) comes from evaluating $t = T$ in the summation.

Computing the sum, we obtain

$$\begin{aligned}\sum_{t=1}^{T-1} \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_t\|^2 - \|\boldsymbol{w}_{t+1} - \boldsymbol{u}_{t+1}\|^2 &= \sum_{t=1}^{T-1} \|\boldsymbol{u}_t\|^2 - \|\boldsymbol{u}_{t+1}\|^2 - 2\langle \boldsymbol{w}_{t+1}, (\boldsymbol{u}_t - \boldsymbol{u}_{t+1})\rangle \\ &\geq \sum_{t=1}^{T-1} \|\boldsymbol{u}_t\|^2 - \|\boldsymbol{u}_{t+1}\|^2 - 2\,\|\boldsymbol{w}_{t+1}\|\,\|\boldsymbol{u}_t - \boldsymbol{u}_{t+1}\| \\ &\geq \|\boldsymbol{u}_1\|^2 - \|\boldsymbol{u}_T\|^2 - 2R \sum_{t=1}^{T-1} \|\boldsymbol{u}_t - \boldsymbol{u}_{t+1}\| \qquad (5.47)\end{aligned}$$

where Equation (5.47) comes from $\|\boldsymbol{w}_{t+1}\| \leq R$, a consequence of the projection step. Substituting this back into Equations (5.44) and (5.46), we then obtain

$$\begin{aligned}\sum_{t=1}^{T} \langle \boldsymbol{w}_t - \boldsymbol{u}_t, \boldsymbol{z}_t\rangle &\leq \frac{1}{2\eta}\left(\|\boldsymbol{u}_T\|^2 + 2R \sum_{t=1}^{T-1} \|\boldsymbol{u}_t - \boldsymbol{u}_{t+1}\| + \sum_{t=1}^{T} \eta^2 \|\boldsymbol{z}_t\|^2\right) \\ &\leq \frac{1}{2\eta} U^2 + \frac{\eta}{2} X^2 T. \\ &= \sqrt{U^2 X^2 T},\end{aligned}$$

where the second inequality comes from the definitions of $\boldsymbol{z}_t$, U and X, and the equality comes from the definition of η. □

Theorem 80. *For Algorithm 6, given* $X = \max_t \|\boldsymbol{x}_t\|$, $\{\boldsymbol{u}_1, \dots \boldsymbol{u}_T\} \subset \{\boldsymbol{u} : \|\boldsymbol{u}\| \leq R\}$, *and* $\sqrt{\|\boldsymbol{u}_T\|^2 + 2R \sum_{t=1}^{T-1} \|\boldsymbol{u}_{t+1} - \boldsymbol{u}_t\|} \leq U$, *and* $\eta = \frac{U}{X\sqrt{T}}$ *we have that*

$$\sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)] - \frac{1+\lambda}{2} \mathcal{L}_{01}(y_t, \operatorname{sign}(\langle \boldsymbol{u}_t, \boldsymbol{x}_t\rangle)) \leq \frac{1}{2}\sqrt{U^2 X^2 T}\,,$$

for any sequence of vectors $\boldsymbol{u}_1, \dots \boldsymbol{u}_T$ such that $1 \leq |\langle \boldsymbol{u}_t, \boldsymbol{x}_t\rangle| \leq \lambda$ for $t = 1, \dots, T$.

Proof. The bound follows from Lemma 79, where we bound the hinge loss terms as follows:

1. $\sum_{t=1}^{T} \mathcal{L}_{\text{hi}}(y_t, \bar{y}_t) \geq 2 \sum_{t=1}^{T} \mathbb{E}[\mathcal{L}_{01}(y_t, \hat{y}_t)]$ (see e.g. Lemma 23)

2. $\mathcal{L}_{\text{hi}}(y_t, \langle \boldsymbol{u}, \boldsymbol{x}_t\rangle) \leq (1+\lambda)\, \mathcal{L}_{01}(y_t, \operatorname{sign}(\langle \boldsymbol{u}_t, \boldsymbol{x}_t\rangle))$ (see the proof for Theorem 78 for more details).

□

The bound for the switching case then follows from Theorem 80 by setting $R = \max_t \|\boldsymbol{u}_t\|$, and $U = \sqrt{(4k+1)\max_t \|\boldsymbol{u}_t\|^2}$, noting that

$$\begin{aligned}\|\boldsymbol{u}_T\|^2 + 2R\sum_{t=1}^{T-1}\|\boldsymbol{u}_{t+1} - \boldsymbol{u}_t\| &\leq \|\boldsymbol{u}_T\|^2 + 2\max_t \|\boldsymbol{u}_t\|\,(2k\max_t \|\boldsymbol{u}_t\|)\\ &= \|\boldsymbol{u}_T\|^2 + 4k\max_t \|\boldsymbol{u}_t\|^2\\ &\leq (4k+1)\max_t \|\boldsymbol{u}_t\|^2\\ &= U^2.\end{aligned}$$

This gives us a regret bound of

$$\sum_{t=1}^{T}\mathbb{E}[\mathcal{L}_{01}(y_t,\hat{y}_t)] - \frac{1+\lambda}{2}\mathcal{L}_{01}(y_t,\langle\boldsymbol{u}_t,\boldsymbol{x}_t\rangle) \leq \frac{1}{2}\sqrt{(4k+1)\max_t \|\boldsymbol{u}_t\|^2 X^2 T}\,.$$

Using the inequality $\sqrt{a+b} \leq \sqrt{a} + \sqrt{b}$ for positive scalars a and b, we then obtain the desired scaling.

Chapter 6

Conclusion

In Chapter 3, we provide efficient algorithms for online matrix completion with side information. We consider two possible settings for how the side information may be revealed. In the transductive setting, the side information of all the rows and columns is provided as two positive semi-definite matrices, whereas in the inductive setting, we are given two kernels, and the side information is revealed in an online fashion. The algorithms that we provide are instances of the MEG and MGD algorithms. For the MEG algorithm, we prove mistake bounds of the form $\tilde{O}(\frac{\mathcal{D}}{\gamma^2})$, where $\mathcal{D}$ measures the quality of the side information, and γ^2 is the margin complexity when exactly tuned. The MGD algorithm has an inferior mistake bound of $O(\frac{\mathcal{D}^2}{\gamma^2})$, but a superior time complexity. The prediction of the MGD algorithm can easily be shown to have a dual form; depending on the embedding, it is equivalent to the kernel perceptron prediction with the kernel being the product or sum squared of the row and column kernels. We observe in the discussion of Chapter 4 that the MGD algorithm can be naturally extended to the tensor case, whereas this remains an open problem for the MEG algorithm.

In Chapter 4, we apply our bounds on the hypothesis class of biclustered matrices. For these matrices, our best mistake bounds in the transductive setting is of $\tilde{O}(k\ell)$, which is tight up to logarithmic factors. In the inductive setting however, our best mistake bounds are $\tilde{O}(\min(k, \ell) \max(k^2, \ell^2))$. It remains to be seen whether this gap can be resolved. In Chapter 5, we introduce the novel setting of online multi-

task learning with long-term memory. We frame this problem as a special case of completing a biclustered matrix, and propose to solve this using our inductive algorithm for matrix completion up to a few redefinitions. The applications that we have considered so far are based on considering the hypothesis class of biclustered matrices. However, the bounds that we prove in Chapter 3 are completely general, and other research directions include applying our bounds to other hypothesis classes of matrices, which could give rise to other applications.

In addition to our theoretical results, we perform experiments where we apply a sketching method on the transductive algorithm in Chapter 4 and decay methods in Chapter 5. Although these heuristics to reduce the time complexity seem to work to varying degrees, theoretical guarantees are still lacking. Having interpretable bounds for these heuristics would lend weight to large scale applications, for which the otherwise cubic and quartic runtimes may become prohibitive. The experiments that we include in the thesis are done on synthetic data. To further demonstrate the applicability of the algorithms, experiments on real world data can be considered in the future.

Bibliography

[1] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of the KDD Cup Workshop 2007*, pages 3–6, New York, August 2007. ACM.

[2] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to Weave an Information tapestry. *Communications of the ACM*, 35(12):61–70, dec 1992.

[3] Jason D Lee, Ben Recht, Nathan Srebro, Joel Tropp, and Russ R Salakhutdinov. Practical large-scale optimization for max-norm regularization. In J. D. Lafferty, C. K. I. Williams, J. Shawe-Taylor, R. S. Zemel, and A. Culotta, editors, *Advances in Neural Information Processing Systems 23*, pages 1297–1305. Curran Associates, Inc., 2010.

[4] 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, 2005.

[5] F Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

[6] Albert B.J. Novikoff. On convergence proofs on perceptrons. *Proceedings of the Symposium on the Mathematical Theory of Automata*, 12:615–622, 1962.

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

[8] Mark Herbster, Stephen Pasteris, and Lisa Tse. Online multitask learning with long-term memory. *Neural Information Processing Systems*, (33), 2020.

[9] N. Linial, S. Mendelson, G. Schechtman, and A. Shraibman. Complexity measures of sign matrices. *Combinatorica*, 27(4):439–463, 2007.

[10] M. Herbster and M. Pontil. Prediction on a graph with a perceptron. In *Advances in Neural Information Processing Systems 19*, pages 577–584, 2006.

[11] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[12] S. Shalev-Shwartz. Online Learning and Online Convex Optimization. *Foundations and Trends® in Machine Learning*, 4(2):107–194, 2011.

[13] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[14] J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. *IEEE Transactions on Signal Processing*, 52:2165–2176, 2004.

[15] Nello Cristianini, John Shawe-Taylor, et al. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[16] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, April 1988.

[17] J M Barzdin and R V Freivald. On the prediction of general recursive functions. *Soviet Math. Doklady*, 13:1224–1228, 1972.

[18] Dana Angluin. Queries and Concept Learning. *Machine Learning*, 2(4):319–342, 1988.

[19] Nick Littlestone and Manfred K. Warmuth. The weighted majority algorithm. *Inf. Comput.*, 108(2):212–261, February 1994.

[20] Jyrki Kivinen and Manfred K Warmuth. Averaging expert predictions. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 1572, pages 153–167, 1999.

[21] David Haussler, Jyrki Kivinen, and Manfred K. Warmuth. Tight worst-case loss bounds for predicting with expert advice. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 1995.

[22] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119?139, August 1997.

[23] Jyrki Kivinen and Manfred K Warmuth. Exponentiated Gradient versus Gradient Descent for Linear Predictors. *Information and Computation*, 132(1):1–63, 1997.

[24] S. Arora and S. Kale. A combinatorial, primal-dual approach to semidefinite programs. In *Proceedings of the 29th Annual ACM Symposium on Theory of Computing*, pages 227–236, 2007.

[25] E. Hazan, S. Kale, and S. Shalev-Shwartz. Near-optimal algorithms for online matrix prediction. In *Proceedings of the 23rd Annual Conference on Learning Theory*, volume 23:38.1-38.13, 2012.

[26] M.K. Warmuth. Winnowing subspaces. In *Proceedings of the 24th International Conference on Machine Learning*, pages 999–1006, 2007.

[27] S. A. Goldman, R. L. Rivest, and R. E. Schapire. Learning binary relations and total orders. *SIAM J. Comput.*, 22(5), 1993.

[28] Pratik Biswas, Tzu-Chen Lian, Ta-Chung Wang, and Yinyu Ye. Semidefinite programming based algorithms for sensor network localization. *ACM Transactions on Sensor Networks (TOSN)*, 2(2):188–220, 2006.

[29] Ralph Schmidt. Multiple emitter location and signal parameter estimation. *IEEE transactions on antennas and propagation*, 34(3):276–280, 1986.

[30] Mehran Mesbahi and George P Papavassilopoulos. On the rank minimization problem over a positive semidefinite linear matrix inequality. *IEEE Transactions on Automatic Control*, 42(2):239–243, 1997.

[31] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: a factorization method. *International journal of computer vision*, 9(2):137–154, 1992.

[32] S. A. Goldman and M. K. Warmuth. Learning binary relations using weighted majority voting. In *Proceedings of the 6th Annual Conference on Computational Learning Theory*, pages 453–462, 1993.

[33] N. Cesa-Bianchi and O. Shamir. Efficient online learning via randomized rounding. In *Advances in Neural Information Processing Systems 24*, pages 343–351, 2011.

[34] M. Herbster, S. Pasteris, and M. Pontil. Mistake bounds for binary matrix completion. In *Advances in Neural Information Processing Systems 29*, pages 3954–3962. 2016.

[35] Ken Ichiro Moridomi, Kohei Hatano, Eiji Takimoto, and Koji Tsuda. Online matrix prediction for sparse loss matrices. *Journal of Machine Learning Research*, 39(2014):250–265, 2014.

[36] J. Nie, W. Kotłowski, and M. K. Warmuth. Online PCA with optimal regrets. In *Proceedings of the 24th International Conference on Algorithmic Learning Theory*, pages 98–112, 2013.

[37] Dima Kuzmin and Manfred K. Warmuth. Online kernel PCA with entropic matrix updates. *ACM International Conference Proceeding Series*, 227:465–472, 2007.

[38] C. Gentile, M. Herbster, and S. Pasteris. Online similarity prediction of networked data from known and unknown graphs. In *Proceedings of the 26th Annual Conference on Learning Theory*, 2013.

[39] M. Herbster, S. Pasteris, and S. Pontil. Predicting a switching sequence of graph labelings. *Journal of Machine Learning Research*, 16:2003–2022, 2015.

[40] Matthew Brand. Fast online svd revisions for lightweight recommender systems. In *Proceedings of the 2003 SIAM international conference on data mining*, pages 37–46. SIAM, 2003.

[41] Laura Balzano, Robert Nowak, and Benjamin Recht. Online identification and tracking of subspaces from highly incomplete information. *2010 48th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2010*, pages 704–711, jun 2010.

[42] Xin Luo, Yunni Xia, and Qingsheng Zhu. Incremental Collaborative Filtering recommender based on Regularized Matrix Factorization. *Knowledge-Based Systems*, 27:271–280, mar 2012.

[43] Akshay Krishnamurthy and Aarti Singh. Low-rank matrix and tensor completion via adaptive sampling. In *Advances in Neural Information Processing Systems*, 2013.

[44] Charanpal Dhanjal, Romaric Gaudel, and Stéphan Clémençon. Online matrix completion through nuclear norm regularisation. In *SIAM International Conference on Data Mining 2014, SDM 2014*, volume 2, pages 623–631, 2014.

[45] Brian Lois and Namrata Vaswani. Online matrix completion and online robust PCA. In *IEEE International Symposium on Information Theory - Proceedings*, volume 2015-June, pages 1826–1830, 2015.

[46] Se-Young Yun, Marc Lelarge, and Alexandre Proutiere. Streaming, memory limited matrix completion with noise. *arXiv preprint arXiv:1504.03156*, 2015.

[47] Chi Jin, Sham M Kakade, and Praneeth Netrapalli. Provable efficient online matrix completion via non-convex stochastic gradient descent. *Advances in Neural Information Processing Systems*, 29, 2016.

[48] Jicong Fan and Madeleine Udell. Online high rank matrix completion. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2019-June, pages 8682–8690, 2019.

[49] Jaya Kawale, Hung Bui, Branislav Kveton, Long Tran Thanh, and Sanjay Chawla. Efficient Thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems*, volume 2015-Janua, pages 1297–1305, 2015.

[50] M. Fazel, H. Hindi, and S. P. Boyd. A rank minimization heuristic with application to minimum orders system approximation. *Proceedings of the American Control Conference*, 2001.

[51] N. Srebro, J. D. M. Rennie, and T. S. Jaakkola. Maximum-margin matrix factorization. *Advances in Neural Information Processing Systems 17*, 2005.

[52] Ruslan Salakhutdinov and Andriy Mnih. Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems 20 - Proceedings of the 2007 Conference*, 2009.

[53] N. Srebro and A. Shraibman. Rank, trace-norm and max-norm. In *Proceedings of the 18th Annual Conference on Learning Theory*, pages 545–560, 2005.

[54] E. J. Candes and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, Col. 9:717–772, 2008.

[55] E. J. Candes and T. Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Trans. Inform. Theory*, Vol 56(5):2053–2080, 2009.

[56] Nathan Srebro and Russ R Salakhutdinov. Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. *Advances in neural information processing systems*, 23, 2010.

[57] Rina Foygel, Ruslan Salakhutdinov, Ohad Shamir, and Nathan Srebro. Learning with the Weighted Trace-norm under Arbitrary Sampling Distributions. *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011*, jun 2011.

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

[59] Raghunandan H. Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from a few entries. *IEEE Transactions on Information Theory*, 2010.

[60] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the Annual ACM Symposium on Theory of Computing*, 2013.

[61] Ruoyu Sun and Zhi Quan Luo. Guaranteed Matrix Completion via Non-Convex Factorization. *IEEE Transactions on Information Theory*, 2016.

[62] Prateek Jain and Inderjit S Dhillon. Provable Inductive Matrix Completion. *arXiv preprint arXiv:1306.0626*, 2013.

[63] M Xu, R Jin, and Z. H. Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in Neural Information Processing Systems*, 2013.

[64] X. Zhang, S. S. Du, and Q. Gu. Fast and Sample Efficient Inductive Matrix Completion via Multi-Phase Procrustes Flow. In *Proceedings of Machine Learning Research*, 2018.

[65] J. Abernethy, F. Bach, T. Evgeniou, and J. Vert. Low-rank matrix factorization with attributes. In *ArXiv preprint ArXiv: cs/0611124*, 2006.

[66] V. Kalofolias, X. Bresson, M. Bronstein, and P. Vandergheynst. Matrix completion on graphs. Technical report, EPFL, 2014.

[67] N. Rao, P. Yu, H.-F.; Ravikumar, and I. Dhillon. Collaborative Filtering with Graph Information: Consistency and Scalable Methods. In *Advances in Neural Information Processing Systems*, 2015.

[68] S. Ben-David, N. Eiron, and H. U. Simon. Limitations of learning via embeddings in euclidean half spaces. *Journal of Machine Learning Research*, 3:441–461, 2003.

[69] S. Sabato, S. Shalev-Shwartz, N. Srebro, Daniel J. Hsu, and T. Zhang. Learning sparse low-threshold linear classifiers. *Journal of Machine Learning Research*, 16:1275–1304, 2015.

[70] M. K. Warmuth, W. Kotłowski, and S. Zhou. Kernelization of matrix updates, when and how? In *Algorithmic Learning Theory*, pages 350–364, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[71] R. Bhatia. *Matrix Analysis*. Springer Verlag, New York, 1997.

[72] J. A. Hartigan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.

[73] C. Gao, Y. Lu, Z. Ma, and H. H. Zhou. Optimal estimation and completion of matrices with biclustering structures. *Journal of Machine Learning Research*, 17:161:1–161:29, 2016.

[74] E. Candès and B. Recht. Exact matrix completion via convex optimization. *Commun. ACM*, 55(6):111–119, June 2012.

[75] Robert Ganian, Iyad A. Kanj, Sebastian Ordyniak, and Stefan Szeider. Parameterized algorithms for the matrix completion problem. In *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, pages 1642–1651, 2018.

[76] A.B. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, pages 615–622, 1962.

[77] M. Belkin and P. Niyogi. Semi-supervised learning on riemannian manifolds. *Machine Learning*, 56:209–239, 2004.

[78] Xiaojin Zhu and Andrew B. Goldberg. Introduction to semi-supervised learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 2009.

[79] M. Herbster, M. Pontil, and L. Wainer. Online learning over graphs. In *Proceedings of the 22nd International Conference on Machine Learning*, pages 305–312, 2005.

[80] Yair Carmon, John C. Duchi, Aaron Sidford, and Kevin Tian. A rank-1 sketch for matrix multiplicative weights. In Alina Beygelzimer and Daniel Hsu, editors, *Conference on Learning Theory, COLT 2019, 25-28 June 2019, Phoenix, AZ, USA*, volume 99 of *Proceedings of Machine Learning Research*, pages 589–623. PMLR, 2019.

[81] Awad H. Al-Mohy and Nicholas J. Higham. Computing the action of the matrix exponential, with an application to exponential integrators. *SIAM Journal on Scientific Computing*, 33(2):488–511, 2011.

[82] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

[83] O. Bousquet and M.K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3:363–396, 2003.

[84] N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

[85] M. Herbster and M.K. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, 2001.

[86] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(6):121–164, 2012.

[87] Nicolò Cesa-Bianchi, Yoav Freund, David Haussler, David P. Helmbold, Robert E. Schapire, and Manfred K. Warmuth. How to use expert advice. *J. ACM*, 44(3):427?485, May 1997.

[88] Volodimir G. Vovk. Aggregating strategies. In *Proceedings of the Third Annual Workshop on Computational Learning Theory*, COLT '90, pages 371–386, 1990.

[89] M. Herbster and M.K. Warmuth. Tracking the best expert. *Machine Learning*, 32(2):151–178, 1998.

[90] A. György, T. Linder, and G. Lugosi. Tracking the best of many experts. In *Proceedings 18th Annual Conference on Learning Theory*, pages 204–216, 2005.

[91] Wouter M. Koolen and Tim van Erven. Freezing and sleeping: Tracking experts that learn by evolving past posteriors, 2010.

[92] N. Cesa-Bianchi, P. Gaillard, G. Lugosi, and G. Stoltz. Mirror descent meets fixed share (and feels no regret). In *Advances in Neural Information Processing Systems 24*, pages 989–997, 2012.

[93] A. György, T. Linder, and G. Lugosi. Efficient tracking of large classes of experts. *IEEE Transactions on Information Theory*, 58(11):6709–6725, Nov 2012.

[94] D. Adamskiy, M.K. Warmuth, and W.M. Koolen. Putting bayes to sleep. In *Advances in Neural Information Processing Systems 25*, pages 135–143, 2012.

[95] D. Adamskiy, W. M. Koolen, A. Chernov, and V. Vovk. A closer look at adaptive regret. In *Proceedings of the 23rd International Conference on Algorithmic Learning Theory*, ALT'12, pages 290–304, 2012.

[96] A. Daniely, A. Gonen, and S. Shalev-Shwartz. Strongly adaptive online learning. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 1405–1411, 2015.

[97] Jaouad Mourtada and Odalric-Ambrym Maillard. Efficient tracking of a growing number of experts. In *Proceedings of the 28th International Conference on Algorithmic Learning Theory (ALT)*, volume 76 of *Proceedings of Machine Learning Research*, pages 517–539, 2017.

[98] Maria-Florina Balcan, Travis Dick, and Dravyansh Sharma. Online optimization of piecewise lipschitz functions in changing environments. *CoRR*, abs/1907.09137, 2019.

[99] Kai Zheng, Haipeng Luo, Ilias Diakonikolas, and Liwei Wang. Equipping experts/bandits with long-term memory. In *Advances in Neural Information Processing Systems 32*, pages 5929–5939, 2019.

[100] Michael Mccloskey and Neil J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *The Psychology of Learning and Motivation*, 24:104–169, 1989.

[101] Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128 – 135, 1999.

[102] Giovanni Cavallanti, Nicolò Cesa-Bianchi, and Claudio Gentile. Tracking the best hyperplane with a simple budget perceptron. *Mach. Learn.*, 69(2-3):143–167, 2007.

[103] Jonathan Baxter. Learning internal representations. In *Proceedings of the Eighth Annual Conference on Computational Learning Theory*, COLT ?95, page 311?320, New York, NY, USA, 1995. Association for Computing Machinery.

[104] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[105] Theodoros Evgeniou and Massimiliano Pontil. Regularized multi–task learning. In *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 109–117, New York, NY, USA, 2004. Association for Computing Machinery.

[106] J. Abernethy, P. Bartlett, and A. Rakhlin. Multitask learning with expert advice. In *Proceedings 20th Annual Conference on Learning Theory*, pages 484–498, 2007.

[107] Alekh Agarwal, Alexander Rakhlin, and Peter Bartlett. Matrix regularization techniques for online multitask learning. Technical Report UCB/EECS-2008-138, EECS Department, University of California, Berkeley, Oct 2008.

[108] S. Avishek, R. Piyush, H. Daumé III, and S. Venkatasubramanian. Online learning of multiple tasks and their relationships. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, pages 643–651, 2011.

[109] Alexander Rakhlin, Jacob D. Abernethy, and Peter L. Bartlett. Online discovery of similarity mappings. In Zoubin Ghahramani, editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML*

2007), Corvallis, Oregon, USA, June 20-24, 2007, volume 227 of *ACM International Conference Proceeding Series*, pages 767–774. ACM, 2007.

[110] Gábor Lugosi, Omiros Papaspiliopoulos, and Gilles Stoltz. Online multi-task learning with hard constraints. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009.

[111] O. Dekel, P.M. Long, and Y. Singer. Online learning of multiple tasks with a shared loss. *Journal of Machine Learning Research*, 8(10):2233–2264, 2007.

[112] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 1:2901–2934, 2010.

[113] Christoph Hirnschall, Adish Singla, Sebastian Tschiatschek, and Andreas Krause. Coordinated online learning with applications to learning user preferences. *arXiv preprint arXiv:1702.02849*, 2017.

[114] M. Herbster, G. Lever, and M. Pontil. Online prediction on large diameter graphs. In *Advances in Neural Information Processing Systems 21*, pages 649–656, 2008.

[115] Ofer Dekel, Shai Shalev-shwartz, and Yoram Singer. The forgetron: A kernel-based perceptron on a fixed budget. In Y. Weiss, B. Schölkopf, and J. C. Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 259–266. MIT Press, 2006.

[116] Francesco Orabona and Dávid Pál. Coin betting and parameter-free online learning. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS16, pages 577–585, Red Hook, NY, USA, 2016. Curran Associates Inc.

[117] Douglas Klein and Milan Randic. Resistance distance. *Journal of Mathematical Chemistry*, 12:81–95, 12 1993.

[118] M. Herbster, M. Pontil, and S. Rojas-Galeano. Fast prediction on a tree. In *Advances in Neural Information Processing Systems*, pages 657–664, 2009.

[119] J. Forster, N. Schmitt, and H.U. Simon. Estimating the optimal margins of embeddings in euclidean half spaces. In *Proceedings Computational Learning Theory*, pages 402–415, 2001.

[120] Shai Ben-David, Dávid Pál, and Shai Shalev-Shwartz. Agnostic online learning. In *COLT 2009 - The 22nd Conference on Learning Theory, Montreal, Quebec, Canada, June 18-21, 2009*, 2009.

[121] Martin Zinkevich. Online Convex Programming and Generalized Infinitesimal Gradient Ascent. In *Proceedings, Twentieth International Conference on Machine Learning*, volume 2, pages 928–935, 2003.