Efficient Rank Aggregation from Pairwise Comparisons

Tao Jin

A Dissertation Presented to
the Faculty of the School of Engineering and Applied Science
University of Virginia

In Partial Fulfillment
of the requirements for the Degree
Doctor of Philosophy (Computer Science)

Abstract

Rank aggregation is a widely applicable task in various domains, including voting, gaming, and recommendation systems. It involves combining pairwise or listwise comparisons to generate a unified ranking. This topic has a long history and is still relevant today: it dates back to democratic voting systems in Ancient Greece; it was modeled in psychology experiments in the last century, and it is the cornerstone of reinforcement learning with human feedback (RLHF) in large language model (LLM) fine-tuning to produce high quality output aligned with human values. In the era of big data, with an abundance of available data, there is a growing need for efficient and accurate analysis to uncover hidden knowledge.

Motivated by the inefficiencies during usage and acquisition of data from multiple sources with different levels of quality during rank aggregation, this work aims to address the challenges of efficiently and accurately dealing with heterogeneous data sources in multiple scenarios.

First of all we propose a novel model that originates from the Random Utility Model (RUM) to take account of the heterogeneity of data sources. Next, we devised an active ranking algorithm that works not only for the proposed model but also for a wider class of models that in the family of Strong Stochastic Transitivity (SST) models.

Noted by the inefficiency of the active ranking algorithm above for Weak Stochastic Transitivity (WST) models that are more prevalent in real-world scenarios, we further propose a new variant of the active ranking algorithm that is able to handle WST models. Following this work, we provide the heterogeneous variant of this algorithm that also efficiently aggregates from multiple data sources setting.

Active ranking techniques aim to minimize the number of samples needed to generate an aggregated ranking by strategically selecting data based on existing information and rankings. If the objective of the algorithm is both to rank items and to collect rewards such as on online shopping platforms where the seller is interested in both collecting the revenue and figuring out the best seller, the ranking problem can be formulated as a dueling bandits problem. This work further covers this case by considering several generalized linear models that contains variable data source quality. We first fill the void of lack of Borda score bandits in the field by proposing a new variant of the dueling bandits algorithm that is able to handle Borda score. This work is further extended to the variance-aware Borda score dueling bandits that is able to handle the heterogeneous data sources setting. The proposed methods achieve nearly optimal regret for this class of problems.

Acknowledgments

First and foremost, immeasurable gratitude is extended to my advisor, Professor Farzad Farnoud, for his invaluable guidance and unwavering support throughout my Ph.D. journey. He will always be my role model. The discussions we shared deepened our understanding of problems and led to natural insights. I deeply admire his knowledge and insight, which I may never fully match. His profound knowledge and remarkable ability to connect different concepts have been truly inspiring.

Heartfelt appreciation goes to Professor Quanquan Gu. Although not my formal advisor, he has generously offered his time and expertise over the years for research discussions. Through these interactions, invaluable behind-the-scenes understanding of research problems was gained, enabling more independent progress in my work.

Sincere thanks are extended to my collaborators, Yue Wu, Qiwei Di, and Professor Pan Xu. Despite the geographical distance, a strong collaborative relationship was maintained through countless virtual meetings. Their availability and engagement in problem-solving discussions have been invaluable to my research journey.

My research group peers - Hao Lou, Yuanyuan Tang, Kallie Whritenour, Yuting Li, Sarvin Motamen and Haoxuan Luo - deserve special recognition for their peer support and the positive academic environment we shared.

The friendship and support of Bingchen Gong, Xuejing Huang, Yueyue Song, Chang Xin, Mengjing Chen, Haodong Zhou, Zeyuan Liao, Juan Zhu, and Shinya Li are profoundly valued.

Finally, deepest gratitude is expressed to my parents and family for their unconditional love and support, without which this journey would not have been possible.

Contents

Abstract . . . 1
Acknowledgments . . . 2
List of Figures . . . 7
List of Tables . . . 8

1 Introduction **9**
1.1 Wide Range of Rank Aggregation Applications and Approaches . . . 9
1.1.1 Static Aggregation . . . 9
1.1.2 Active Ranking . . . 10
1.1.3 Dueling Bandits with Side Information . . . 10
1.2 Preliminaries . . . 10
1.2.1 Rank Aggregation Models . . . 11
1.3 Random Utility Models . . . 11
1.4 Stochastic Transitivity Models . . . 12
1.5 Score-Based Ranking . . . 13
1.6 Summary of Notations . . . 13
1.7 Thesis Outline . . . 14
1.7.1 Heterogeneous Random Utility Model . . . 15
1.7.2 Adaptive Sampling of Heterogeneous Users in Active Ranking . . . 15
1.7.3 Active Ranking under Weak Stochastic Transitivity Assumption . . . 16
1.7.4 Heterogeneous Active Ranking under WST Assumptions . . . 16
1.7.5 Score-Based Contextual Dueling Bandits . . . 17
1.7.6 Variance-Aware Contextual Dueling Bandits . . . 17

I Efficient Ranking under Strong Stochastic Transitivity Assumption **19**

2 Heterogeneous Random Utility Models **20**
2.1 Introduction . . . 20
2.2 Related Work . . . 20
2.3 Modeling Heterogeneous Ranking Data . . . 21
2.3.1 The Heterogeneous Random Utility Model . . . 21
2.3.2 Rank Aggregation via Maximum Likelihood Estimation . . . 22
2.4 Theoretical Analysis . . . 24
2.4.1 Implications of Specific Models . . . 25
2.4.2 Proof of the Generic Model . . . 26
2.4.3 Proofs of Specific Examples . . . 28
2.4.4 Proofs of Technical Lemmas . . . 32
2.5 Experiments . . . 37
2.5.1 Experimental Results on Synthetic Data . . . 37
2.5.2 Experimental Results on Real-World Data . . . 39
2.5.3 Analysis on regularization effects . . . 39

3 Heterogeneous Active Ranking under Strong Stochastic Assumption **45**
3.1 Introduction . . . 45
3.2 Related Work . . . 45
3.3 Preliminaries and Problem Setup . . . 46
3.3.1 Ranking from Noisy Pairwise Comparisons . . . 46
3.3.2 Iterative Insertion Ranking with a Single User . . . 46
3.4 Adaptive Sampling and User Elimination . . . 47
3.4.1 A Two-stage Algorithm as Baseline . . . 48
3.5 Theoretical Analysis . . . 52
3.5.1 Sample Complexity of the Proposed Algorithm . . . 52
3.5.2 Sample Complexity Gap Analysis . . . 53
3.5.3 Discussion on the Sample Complexity Gap and the Optimality of the Proposed Algorithm 56
3.6 Experiments . . . 57
3.6.1 Synthetic Experiment . . . 57
3.6.2 Real-world Experiment . . . 58

II Efficient Ranking under Weak Stochastic Transitivity Assumption **60**

4 Active Ranking under Weak Stochastic Transitivity **61**
4.1 Introduction . . . 61
4.2 Problem Setup . . . 62
4.3 Proposed Algorithm . . . 62
4.3.1 A Sample-efficient Variant of `Probe-Rank` . . . 65
4.4 Theoretical Analysis . . . 67
4.4.1 Upper Bound on the Sample Complexity . . . 67
4.4.2 Lower Bound on the Sample Complexity . . . 70
4.5 Experiments . . . 72
4.5.1 Detailed Experiments . . . 74

5 Heterogeneous Active Ranking under Weak Stochastic Transitivity **78**
5.1 Introduction . . . 78
5.2 Related work . . . 78
5.3 Problem Setup and Preliminaries . . . 79
5.3.1 Harndness Factor for Ranking two items . . . 80
5.4 Heterogeneous Ranking Algorithm under WST Condition . . . 80
5.5 Theoretical Analysis . . . 82
5.5.1 Upper Bound of the Sample Complexity . . . 82
5.5.2 Lower Bound of the Sample Complexity for Multiple Oracles . . . 86
5.6 Experiments . . . 87
5.6.1 Improved Algorithm for Practical Use . . . 87

III Efficient Rank Aggregation in Contextual Dueling Bandits **89**

6 Contextual Borda Dueling Bandits **90**
6.1 Introduction . . . 90
6.2 Related Work . . . 91
6.3 Problem Setup and Preliminaries . . . 92
6.3.1 Assumptions . . . 93
6.3.2 Existing Results for Structured Contexts . . . 93
6.4 Proposed Algorithm for Generalized Contextual Dueling Bandits . . . 94
6.5 Proposed Algorithm for Adversarial Contextual Dueling Bandit . . . 96
6.5.1 Algorithm Description . . . 96
6.6 Construction of Hardness Cases . . . 96

6.7 Experiments . . . 97
6.7.1 Simulated Study: Generated Hard Case . . . 98
6.7.2 Real-world Data Experiments . . . 98
6.7.3 Additional Information for Experiments . . . 100

7 Variance-Aware Contextual Dueling Bandits **102**
7.1 Introduction . . . 102
7.2 Related Work . . . 102
7.3 Problem Setup . . . 103
7.4 Algorithm . . . 104
7.4.1 Overview of the Algorithm . . . 104
7.4.2 Regularized MLE . . . 104
7.4.3 Multi-layer Structure with Variance-Aware Confidence Radius . . . 106
7.4.4 Symmetric Arm Selection . . . 106
7.5 Variance-aware Regret Bound . . . 109
7.5.1 Proof Sketch of Theorem 7.5.1 . . . 110
7.5.2 Proof of Theorem 7.5.1 . . . 111
7.6 Experiments . . . 113
7.6.1 Additional Experiment on Real-world Data . . . 114
7.6.2 Comparison with Prior Works . . . 114
7.7 Proof of Lemmas . . . 115
7.7.1 Proof of Lemma 7.5.7 . . . 115
7.7.2 Proof of Lemma 7.5.8 . . . 116
7.7.3 Proof of Lemma 7.5.9 . . . 117
7.7.4 Proof of Lemma 7.5.10 . . . 119
7.7.5 Proof of Lemma 7.5.11 . . . 119
7.7.6 Auxiliary Lemmas . . . 120

IV Conclusion **122**

List of Figures

2.1 The effect of γ_u on the probability of error for a BTL comparison in which items have scores 0 and 1. In particular, for large negative values of γ_u, the user is accurate (with a high level of expertise) but adversarial. . . . 22
2.2 Evolution of estimation errors vs. number of iterations t for HBTL model (a-b) and HTCV model (c-d). The plots show the convergence behavior of both score vector $\boldsymbol{s}^*$ and accuracy parameters $\boldsymbol{\gamma}^*$ estimation. . . . 38

3.1 Sample complexities v.s. number of items for all algorithms. (a) (b) and (c) are different heterogeneous user settings where the accuracy of two group of users differs. . . . 58
3.2 Sample complexities v.s. number of items for all algorithms. (a) (b) and (c) are different settings where the number of users differs. The accuracy of two groups of users are $\gamma_A = 0.5$, $\gamma_B = 2.5$. . . . 59

4.1 An illustration of the steps by Probe-Ranking, assuming true ranking as $1 \succ 2 \succ 3 \succ 4$. . . . 65
4.2 Comparison of sample complexities of `Probe-Rank` and `IIR` under various settings. In each subfigure, Δ_d is fixed while the number of items varies. . . . 73
4.3 Relationship between n and gap Δ_d . . . 73
4.4 Ablation study on the dependence of the sample complexity on the probability gap Δ_d. . . . 74
4.5 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the WST setting. In each subfigure, Δ_d is fixed while the number of items varies. . . . 75
4.6 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the SST setting. In each subfigure, Δ_d is fixed while the number of items varies. . . . 75
4.7 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the NON-SST setting. In each subfigure, Δ_d is fixed while the number of items varies. . . . 75
4.8 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the `ADJ-ASYM` setting. In each subfigure, Δ_d and α are fixed while the number of items varies. . . . 76
4.9 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the WST setting. In each subfigure, n is fixed while Δ_d varies. . . . 76
4.10 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the SST setting. In each subfigure, n is fixed while Δ_d varies. . . . 76
4.11 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the NON-SST setting. In each subfigure, n is fixed while Δ_d varies. . . . 77
4.12 Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the `ADJ-ASYM` setting. In each subfigure, n and α are fixed while Δ_d varies. . . . 77

5.1 Sample complexities of ranking $N \in \{2, 4, 8, 16, 32, 64\}$ items with $M - 1$ oracles have low accuracy and one oracle has high accuracy. . . . 88

6.1 Illustration of the hard-to-learn preference probability matrix $\{p^{\boldsymbol{\theta}}_{i,j}\}_{i\in[K],j\in[K]}$. There are $K = 2^{d+1}$ items in total. The first 2^d items are "good" items with higher Borda scores, and the last 2^d items are "bad" items. The upper right block $\{p_{i,j}\}_{i<2^d,j\geq 2^d}$ is defined as shown in the blue bubble. The lower left block satisfies $p_{i,j} = 1 - p_{j,i}$. For any $\boldsymbol{\theta}$, there exist one and only best item i such that $\mathbf{bit}(i) = \mathbf{sign}(\boldsymbol{\theta})$. . . . 97

6.2 The regret of the proposed algorithms (BETC-GLM, BEXP3) and the baseline algorithms (UCB-Borda, DEXP3, ETC-Borda). 98
6.3 The performance of BETC under different choices of error tolerance ϵ, compared with BEXP3. We examined BETC with ϵ, 2ϵ, 4ϵ, 8ϵ where $\epsilon = d^{1/6}T^{-1/3}$. 99
6.4 EventTime 99
6.5 The regret of the proposed algorithm (BETC-GLM,BEXP3) and the baseline algorithms (UCB-Borda, DEXP3, ETC-Borda). 99

7.1 Experiments showing regret performance in various settings. 114
7.2 Regret comparison between `VACDB` and `MaxInP` on a real-world dataset. 115

List of Tables

1.1 Thesis Overview 14

2.1 Kendall's tau correlation for different method under Gumbel noise. Group A users all have the accuracy level γ_A and Group B users all have the accuracy level γ_B. α represents the portion of all possible pairwise comparisons each annotator labeled in the simulation. The **bold** number highlights the highest performance and the underlined number indicates a tie. 40

2.2 Kendall's tau correlation for different methods under noise from the normal distribution. Group A users all have the accuracy level γ_A and Group B users all have the accuracy level γ_B. α represents the portion of all possible pairwise comparisons each annotator labeled in the simulation. The **bold** number highlights the highest performance and the underlined number indicates a tie. 41

2.3 Kendall's tau correlation for different methods under noise from the Gumbel distribution when a third of the users are *adversarial*. The **bold** number highlights the highest performance and the underlined number indicates a tie. 42

2.4 Kendall tau correlation for different methods under noise from the normal distribution when a third of the users are *adversarial*. The **bold** number highlights the highest performance and the underlined number indicates a tie. 43

2.5 Ground truth for "Country Population" dataset. 44

2.6 Performance of ranking algorithms on real-world dataset. The **bold** number highlights the highest performance. 44

2.7 Performance of ranking algorithms for the "Reading Level" dataset with different regularization parameters. The **bold** number highlights the highest performance. 44

2.8 Performance of ranking algorithms for the "Country Population" dataset with different regularization parameters. The **bold** number highlights the highest performance. 44

3.1 Experiments on Country Population with 15 items and 25 users. 59

4.1 δ-correct algorithms for exact ranking with sample complexity guarantee under WST assumption. 62

5.1 A comparison among the related works and the proposed method. The table is divided into two major sections. The upper section mainly shows the sample complexities of three related algorithms under the *SST* condition. The lower section of the table shows the result under the *WST* condition. 79

Chapter 1

Introduction

1.1 Wide Range of Rank Aggregation Applications and Approaches

Rank aggregation is the task of recovering the order of a set of objects from pairwise comparisons, partial rankings, or full rankings provided by users or experts. This approach offers several advantages over traditional rating systems: comparisons are more natural for humans to make and provide more consistent results since they don't rely on arbitrary scales. Ranked data can be collected both explicitly through user queries and passively through observation of user behavior, such as product purchases, search engine clicks, or streaming service choices.

The applications of rank aggregation span diverse domains, from classical social choice theory (de Borda, 1781) to modern applications in information retrieval (Dwork et al., 2001), recommendation systems (Baltrunas et al., 2010; Piech et al., 2013), and bioinformatics (Aerts et al., 2006; Kim et al., 2015). In recommendation systems, rank aggregation combines various factors like purchase history and browsing behavior into unified product rankings. In gene expression analysis, it helps identify genes most relevant to specific diseases by combining rankings from different statistical methods. Search engines use rank aggregation to merge results from different ranking algorithms and user click patterns. In sports, it helps determine team rankings by combining results from multiple matches and tournaments. Political voting systems (Caplin and Nalebuff, 1991; Conitzer and Sandholm, 2005) use rank aggregation to combine individual voter preferences into a collective decision. In clinical trials, it helps rank treatment effectiveness by combining results from different evaluation metrics and expert assessments. Additional applications include ranking players in online gaming systems (Herbrich et al., 2006; Minka et al., 2018).

The data available for rank aggregation typically consists of pairwise comparisons or partial rankings of item subsets. The goal is to combine these into a unified ranking across all items, subject to predefined assumptions or scoring systems. This data is often collected from multiple users or annotators who may have varying levels of expertise or precision in their assessments.

This work focuses on improvement on three main aspects of rank aggregation, each addressing different use cases and objectives:

1. Static Aggregation: Inferring rankings from previously collected preferential data.
2. Active Ranking: Optimizing data collection by actively selecting which comparisons to sample based on existing data.
3. Dueling Bandits: Balancing exploration and exploitation while actively ranking items to maximize rewards.

1.1.1 Static Aggregation

The task of aggregating pairwise comparisons is to infer a ranking from those comparisons from distinct pairs. It is similar to learning a ranking from noisy pairwise comparisons, and it has been studied in many works including Hunter (2004); Braverman and Mossel (2008); Negahban et al. (2012); Wauthier et al. (2013).

Instead of assuming the same probability for all comparisons, it is natural to assume that the comparison of similar items is more likely to be noisy than those items that are distinctly different. This intuition is reflected in the *random utility model* (RUM) or *stochastic utility model* (SUM). It includes models known as *the Thurstone model* (Thurstone, 1927) and *Bradley-Terry-Luce* (BTL) model (Bradley and Terry, 1952; Luce, 1959; Hunter, 2004), where each item has a true score, and data sources provide rankings of subsets of items by comparing approximate versions of these scores corrupted by additive noise.

To aggregate the comparisons for RUM, a maximum likelihood estimator (MLE) for the score (utility) of the item given the observed pairwise comparison data for BTL models (Hunter, 2004). Alternatively, Rank Centrality proposed by Negahban et al. (2012) is an iterative method with a random walk interpretation which perform the inference more efficiently than the MLE. And it is shown that it performs as well as the MLE and it provided non-asymptotic performance guarantees. Chen and Suh (2015) extend this work to identify the top-k candidates efficiently.

1.1.2 Active Ranking

The methods described in the previous section are passive and only apply to the already collected data. A possible improvement to increase data collection efficiency is to actively request the data required to perform inference. In contrast to passive algorithms, active algorithms leverage assumptions embedded in the models to identify the most informative pairs to query, thus reducing the sample complexity.

Ailon (2012) proposes an active learning approach that assumes no transitivity in pairwise ordering with nearly optimal loss from pairwise comparisons to obtain an *approximate* aggregated ranking. Starting from Maystre and Grossglauser (2017), active ranking methods from noisy comparisons have seem similar design that resembles "Merge Sort" or "Binary Search Insertion Sort" for deterministic scenario. This works provides an *approximate* ranking of the items of interest. This trend is further explored by Ren et al. (2019) who provided an analysis for a distribution agnostic active ranking scheme called the `Iterative-Insertion-Ranking` (`IIR`) algorithm for the *exact* ranking problem. It maintains a preference tree and performs ranking by inserting items one after another. During the insertion process, every item is to be compared with increasingly similar items to determine its placement in the ranking.

1.1.3 Dueling Bandits with Side Information

Dueling Bandits is derived from the Multi-Armed Bandits (MAB) problem (Lattimore and Szepesvári, 2020), which is an interactive process where in each round, an agent chooses an arm to pull and receives a noisy reward as feedback. For MAB, the feedback is usually numerical. However, compared to preferential feedback, numerical feedback is more difficult to gauge and prone to errors in many real-world applications. This motivates *Dueling Bandits* (Yue and Joachims, 2009), where the agent repeatedly pulls two arms at a time and is provided with feedback being the binary outcome of "duels" between the two arms.

In many real-world applications, side information is available for each option that needs to be ranked. This side information can provide valuable insights into the ordering of the options. For example, in e-commerce, an item's category and other attributes can influence user preferences. Similarly, in the movie industry, factors such as genre, plot, directors, and actors can play a role in decision-making. To address such scenarios, contextual bandit algorithms have been developed. These algorithms leverage the side information provided to the agent and assume that rewards have a linear structure. Various algorithms (Filippi et al., 2010; Abbasi-Yadkori et al., 2011; Li et al., 2017; Jun et al., 2017) have been proposed to utilize this contextual information.

1.2 Preliminaries

Assume there are N items to be ranked, and there are M sources that can provide preferential feedback, i.e., a given source returns the preferred item when being presented with two items after comparison. Specifically, an "item" can be something that is subject to ordering according to quality, popularity, usefulness, skills, etc. For instances, those items appears when we are ranking movies, political candidates, clinical drugs under trial, sports teams, respectively. And a "source" is the provider of the feedback, such as crowd-source worker, expert in a certain field, voter, observation method in scientific experiments, etc.

We use i, j, k to index into the set of items $[N] := \{1, 2, \cdots, N\}$, and use u, v to index into the set of information sources $[M]$. And for each comparison, we use $r \in \{0, 1\}$ to denote the response of it, where $r = 1$ denotes the first item in the pair is preferred in the response provided by the source.

It is assumed that the feedback r of from a comparion pair (i, j) is a random variable that has an expected value of $p_{i,j}$: $\Pr(r = 1) = \Pr(i \succ j) = p_{i,j}$. A *probability matrix* is defined as $P := \{p_{i,j}\}$ if there is only a single source, or it is treated as a single one. If it is assumed that for different sources of comparisons such probabilities are different, then we define a set of *probability matrices* as $P^{(u)} := \{p_{i,j}^{(u)}\}, u \in [M]$, where the superscript u denotes the source.

Define $\pi(\cdot) : [N] \to [N]$ as a mapping that maps from the position in the ranking to the actual item. And $\sigma(\cdot) : [N] \to [N]$ as the position of item in the ranking. The inverse of them satisfies $\pi(i) = \sigma^{-1}(i)$ or $\sigma(i) = \pi^{-1}(i)$. Under this definition, a ranking of N items can be written as $\pi(1) \succ \pi(2) \cdots \succ \pi(N)$ is equivalent to $\sigma^{-1}(1) \succ \sigma^{-1}(2) \cdots \succ \sigma^{-1}(N)$.

We use normal letters to denote scalars, lowercase bold letters to denote vectors, and uppercase bold letters to denote matrices. If each item in the pair being compared brings in additional side information, we denote it as $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. In this case, we use two vectors to denote the items that are compared instead of the index numbers. If the index $i \in [N]$ is given, we also use $\mathbf{x}_i \in \mathbb{R}^d$ to denote its feature vector. For a vector $\mathbf{x}$, $\|\mathbf{x}\|$ denotes its ℓ_2-norm. The weighted ℓ_2-norm associated with a positive-definite matrix $\mathbf{A}$ is defined as $\|\mathbf{x}\|_{\mathbf{A}} = \sqrt{\mathbf{x}^\top \mathbf{A} \mathbf{x}}$. The minimum eigenvalue of a matrix $\mathbf{A}$ is written as $\lambda_{\min}(\mathbf{A})$. We use $\mathbf{A} \succeq \mathbf{B}$ to denote that the matrix $\mathbf{A} - \mathbf{B}$ is positive semi-definite.

1.2.1 Rank Aggregation Models

In real-world scenarios, there are multiple approaches to model and aggregate rankings. These approaches can be broadly categorized into three main classes, each with progressively less restrictive assumptions:

1. **Random Utility Models**: These models assume that each item has an underlying latent score or utility. The probability of one item being preferred over another depends on the difference between their scores. Items with higher scores are ranked higher. This is the most structured approach, as it assumes a clear relationship between item scores and preferences.

2. **Stochastic Transitivity Models**: These models focus on the transitivity property of preferences without assuming underlying scores. If item A is preferred to B, and B is preferred to C, then A should be preferred to C. This approach is more flexible than utility models but still maintains some structure in the preference relationships.

3. **Score-Based Models**: These models define rankings based on aggregate scores derived from pairwise comparison probabilities. The ranking is determined by how well each item performs against others, without assuming any underlying structure or transitivity. This is the most flexible approach, as it makes minimal assumptions about the nature of preferences.

1.3 Random Utility Models

In aggregating rankings, the raw data is often noisy and inconsistent. One approach to arrive at a single ranking is to assume a generative model for the data whose parameters include a true score for each of the items. This approach is known as the *Random Utility Model* (RUM) or *Stochastic Utility Model* (SUM), which is classified as a utility theory model in ranking (Fishburn et al., 1979; Tversky and Kahneman, 1981).

The fundamental assumption in RUM is that each item $i \in [N]$ has a latent score (utility) s_i, and the rank corresponds to the magnitude order of these utilities. When comparing items, users provide rankings by comparing approximate versions of these scores corrupted by additive noise. This intuition reflects that comparisons between similar items are more likely to be noisy than those between distinctly different items.

Consider a set of N items with score vector $\boldsymbol{s} = (s_1, \cdots, s_N)^\top$. These items are evaluated by a set of M independent data sources. For each item i, a data source estimates an empirical score as:

$$z_i = s_i + \epsilon_i, \tag{1.3.1}$$

where ϵ_i is random noise introduced during evaluation. This coarse estimate z_i is implicit and cannot be directly observed. Instead, data sources only produce rankings by sorting these scores. For a subset of items $\{i_1, \cdots, i_h\} \subseteq [N]$, where $2 \leq h \leq N$, we have:

$$\Pr\left(\pi(1) \succ \pi(2) \succ \cdots \succ \pi(h)\right) = \Pr\left(z_{\pi(1)} > z_{\pi(2)} > \cdots > z_{\pi(h)}\right), \tag{1.3.2}$$

where $i \succ j$ indicates that i is preferred to j and $\{\pi(1), \cdots, \pi(h)\}$ is a permutation of $\{i_1, \cdots, i_h\}$ indicating the ranking as we defined in the previous section. Each comparison produces a new score estimate, which are commonly assumed to be i.i.d. (Braverman and Mossel, 2008; Negahban et al., 2012; Wauthier et al., 2013).

The distribution of ϵ_i determines the link function and leads to several widely used models. If ϵ_i follows a Gumbel distribution, it corresponds to the BTL model(Bradley and Terry, 1952), while a Gaussian distribution leads to the Thurstone model (case V) (Thurstone, 1927).

For pairwise comparisons, the probabilistic model assumes that for an observation $i \succ j$:

$$\Pr(i \succ j) = \mu(s_i - s_j) \tag{1.3.3}$$

where $\mu(\cdot)$ is a link function that is symmetric about $(0, \frac{1}{2})$, increasing monotonically, and satisfies $\mu(x) + \mu(-x) = 1$. The logistic function $\sigma(x) = (1 + \exp(-x))^{-1}$ is a popular choice that leads to the BTL model.

The absolute difference of $p_{i,j}$ with $\frac{1}{2}$ represents the noisiness of the collected samples. To simplify the discussion, we define the following terms as the *gap* of a comparison: $\Delta_{i,j} := |p_{i,j} - \frac{1}{2}|$, and superscript with (u) to indicate data sources if applicable: $\Delta_{i,j}^{(u)} := |p_{i,j}^{(u)} - \frac{1}{2}|$.

Linear Stochastic Utility Models When items have associated feature vectors, we can extend RUM to Linear Stochastic Utility Models (Bengs et al., 2022). Here, it is assumed that $s_i = \mathbf{x}_i^\top \boldsymbol{\theta}^*$, where $\boldsymbol{\theta}^* \in \mathbb{R}^d$ is an unknown global parameter that has to be estimated.

For a pair of items with feature vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have:

$$\mathbb{P}(r = 1) = \mathbb{P}(\text{item with } \mathbf{x} \text{ is preferred over item with } \mathbf{y}) = \mu((\mathbf{x} - \mathbf{y})^\top \boldsymbol{\theta}^*). \tag{1.3.4}$$

Various estimation and aggregation algorithms have been developed for RUM and its special cases (Hunter, 2004; Guiver and Snelson, 2009; Hajek et al., 2014; Chen and Suh, 2015; Vojnovic and Yun, 2016; Negahban et al., 2016). In this work, we mainly focus on the Maximum Likelihood Estimation (MLE) methods.

1.4 Stochastic Transitivity Models

In RUM, the model satisfies the *Strong Stochastic Transitivity* (SST) assumption (Feige et al., 1994; Mohajer et al., 2017; Falahatgar et al., 2017a, 2018; Ren et al., 2018, 2019; Saha and Gopalan, 2019). SST requires items that have closer ranks to be more difficult to compare, i.e. if $i \succ j \succ k$, then $p_{i,k} \geq \max(p_{i,j}, p_{j,k}) > \frac{1}{2}$. However, the SST assumption can be too strong for scenarios where relative noisiness of the comparison outcome is not related to ranking. For instance, in sports, match outcomes are usually affected by team tactics. Team k may play a tactic that counters team i, resulting in a higher winning rate against team i compared with team j. Furthermore, items usually have multidimensional features and people may compare different pairs based on different features. A close pair in the overall ranking is thus not necessarily harder to compare than a pair that has a large gap. For example, when comparing cars, people might compare a given pair based on their interior design and another pair based on performance. As another example, in an experiment with games of chance with different probabilities of winning and payoffs Tversky (1969), it was observed that "people chose between adjacent gambles according to the payoff and between the more extreme gambles according to probability, or expected value." A sensible relaxed version of this assumption waives the original requirement and instead assumes that the ranking of items aligns with the probabilities. It only requires that $p_{i,j} > \frac{1}{2}$ when i is preferred to j, i.e., $i \succ j$. This is called the Weak Stochastic Transitivity (WST) assumption and implies that if $i \succ j$ and $j \succ k$ then $i \succ k$.

We formally state these two assumptions as below.

Assumption 1.4.1 (Weak Stochastic Transitivity)**.** For a given data source u, for any pair of items (i, j) and (j, k), if $p^u_{i,j} \geq \frac{1}{2}$ and $p^u_{j,k} \geq \frac{1}{2}$ then $p^u_{i,k} \geq \frac{1}{2}$.

WST implies that if $i \succ j$ and $j \succ k$, then $i \succ k$, which eliminates the possible cyclic order dependency and guarantees that an exact order of items can be inferred.

Assumption 1.4.2 (Strong Stochastic Transitivity)**.** For a given data source u, for any pair of items (i, j) and (j, k), if $p^u_{i,j} \geq \frac{1}{2}$ and $p^u_{j,k} \geq \frac{1}{2}$, then $p^u_{i,k} \geq \max\{p^u_{i,j}, p^u_{j,k}\} \geq \frac{1}{2}$.

1.5 Score-Based Ranking

Although WST can be considered a natural and reasonably weak assumption, there are situations where WST does not hold as an ordering over items may not exist or, if it does, all comparison probabilities are not necessarily consistent with that ranking. So another line of research is to allow comparison probabilities $p_{i,j}$ to take any value in $(0, 1)$ as long as $p_{i,j} + p_{j,i} = 1$.

In such scenarios, rankings can be defined and derived based on various criteria defined beforehand. Some popular choices for such scores are the Borda score (Heckel et al., 2019; Katariya et al., 2018; Shah and Wainwright, 2017) and the Copeland score (Busa-Fekete et al., 2013; Zoghi et al., 2015).

The Borda score is defined as

$$B(i) := \frac{1}{N-1} \sum_{j \in [N], j \neq i} p_{i,j}. \tag{1.5.1}$$

It is essentially an average of all possibilities of the item i wins the other items. The Copeland score is defined as

$$C(i) := \frac{1}{N-1} \sum_{j \in [N], j \neq i} \mathbb{1}\{p_{i,j} > 1/2\}. \tag{1.5.2}$$

A Copeland winner is the item that beats the most number of other items. It can be viewed as a "thresholded" version of Borda winner.

1.6 Summary of Notations

For clarity, we summarize the key notations used throughout this work:

Asymptotic Notation. We use standard asymptotic notations including $O(\cdot)$, $\Omega(\cdot)$, and $\Theta(\cdot)$ in their usual sense. The notations $\widetilde{O}(\cdot)$, $\widetilde{\Omega}(\cdot)$, and $\widetilde{\Theta}(\cdot)$ denote their corresponding weaker forms that hide logarithmic factors.

General Notation. We use lowercase letters for scalars, lowercase bold letters for vectors, and uppercase bold letters for matrices. For a vector $\mathbf{x}$, $\|\mathbf{x}\|$ denotes its Euclidean norm. For a positive integer N, we define $[N] := \{1, 2, \ldots, N\}$.

- Sets and Indices:
  - $[N]$: Set of items to be ranked
  - $[M]$: Set of information sources: could be users, oracles, judges, or annotators
  - $i, j, k \in [N]$: Indices for items
  - $u, v \in [M]$: Indices for information sources
- Probabilities and Preferences:
  - $p_{i,j}$: Probability that item i is preferred over item j
  - $p^{(u)}_{i,j}$: Probability that item i is preferred over item j by source u
  - $r \in \{0, 1\}$: Binary response from a comparison

- $P = \{p_{i,j}\}$: Probability matrix for single source
- $P^{(u)} = \{p_{i,j}^{(u)}\}$: Probability matrix for source u

- Rankings and Mappings:
  - $\pi : [N] \to [N]$: Mapping from rank position to item index
  - $\pi^{-1} : [N] \to [N]$: Inverse mapping from item index to rank position
  - $\sigma^{-1} : [N] \to [N]$: Mapping from rank position to item index
  - $\sigma : [N] \to [N]$: Inverse mapping from item index to rank position
  - $\succ$: Preference relation ($i \succ j$ means item i is preferred over j)

- Feature Vectors and Parameters:
  - $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$: Feature vectors for items
  - $\mathbf{x}_i \in \mathbb{R}^d$: Feature vector for item i
  - $\boldsymbol{\theta}^* \in \mathbb{R}^d$: Unknown parameter vector

- Comparison Metrics:
  - $\Delta_{i,j} = |p_{i,j} - \frac{1}{2}|$: Gap between comparison probability and random guess
  - $\Delta_{i,j}^{(u)} = |p_{i,j}^{(u)} - \frac{1}{2}|$: Gap for source u
  - $\mu(\cdot)$: Link function
  - $\sigma(x) = (1 + \exp(-x))^{-1}$: Logistic function[1]

1.7 Thesis Outline

In this thesis, we present our research on extending rank aggregation models to effectively handle multiple information sources with varying levels of accuracy. We develop novel methods to address active learning challenges under different stochastic transitivity assumptions, both in single-source and heterogeneous-source settings. Additionally, we explore how incorporating side information can improve ranking efficiency, particularly when optimizing for the Borda score in dueling bandits setting.

Table 1.1 summarizes the contributions of the thesis and compares it with existing works. In next sections, we will give a brief discussion of our contributions in order.

Model or Assumption	Objective	Homogeneous Data Source	Heterogeneous Data Source
RUM (Section 1.3)	Static Aggregation (Section 1.1.1)	Negahban et al. (2016)	Chapter 2
SST (Section 1.4)	Active Ranking (Section 1.1.2)	Ren et al. (2019)	Chapter 3
WST (Section 1.4)	Active Ranking (Section 1.1.2)	Chapter 4	Chapter 5
Score-based (Section 1.5)	Dueling Bandits (Section 1.1.3)	Chapter 6	Chapter 7

Table 1.1: Thesis Overview

The following paragraph summarizes the author's contributions and corresponding publications for each chapter of this thesis. In Chapter 2, the author was responsible for the problem formulation, algorithm implementation, and experiments, which resulted in a co-first author publication (Jin et al., 2020). For Chapter 3, the author took charge of the problem formulation, initial proof development, implementation, and experiments, leading to another co-first author publication (Wu et al., 2022). In Chapter 4, the author contributed algorithm implementation, experiments, and developed an improved efficiency version, resulting

[1]This may have clashing notation with the σ in Section 1.4. However, the context should make it clear which one is being referred to.

in a second author publication (Lou et al., 2022). Chapter 5 saw the author responsible for the problem formulation, upper bound proof development, implementation, and experiments, culminating in a co-first author publication (Jin et al., 2025) that is currently under review. For Chapter 6, the author handled the problem formulation, implementation, and experiments, resulting in a second author publication (Wu et al., 2024). Finally, in Chapter 7, the author was in charge of the problem formulation, initial proof development, implementation, and experiments, leading to a co-first author publication (Di et al., 2024).

1.7.1 Heterogeneous Random Utility Model

Traditional data models described in Sections 1.1.1 and 1.3, can only treat information gathered from multiple sources as a single one. This limitation is partially addressed using multiple methods that can handle heterogeneous populations of users with varying levels of expertise under multiple problem settings.

In the *Heterogeneous Random Utility Model* (HRUM) proposed an aggregation method for preference data that has already been acquired. This method can take into account the accuracy levels of different comparison providers. By allowing different noise distributions, the proposed model expands the generality of original framework of RUM, and as such, extends the Bradley-Terry-Luce/Plackett-Luce (BTL/PL) model for pairwise comparisons to heterogeneous populations of information sources. Under this framework, the rank aggregation algorithm is based on alternating gradient descent to simultaneously estimate the underlying item scores and accuracy levels of different sources from noisy pairwise comparisons.

Specifically, it is assumed that each information source has a different probability of making mistakes in evaluating items, i.e., the evaluation noise of source u is controlled by a scaling factor $\gamma_u > 0$. The proposed model is then represented as follows:

$$z_i^u = s_i + \epsilon_i/\gamma_u. \tag{1.7.1}$$

Based on the estimated scores z_i^u of each information source for each item, the probability of a certain ranking pair of items provided by source u is again given by Eq. (1.3.2). With the larger the γ_u, the more accurate for the source. This extension actually applies to both pairwise comparisons and rankings, though pairwise comparisons are the main focus of this work.

This work theoretically shows that the proposed algorithm produced an estimate that converges to unknown scores $\{s_i\}_{i\in[N]}$ and the accuracy factors $\{\gamma_u\}_{u\in[M]}$ at a locally linear rate up to a tight statistical error under mild conditions. For models with specific noise distributions such as the For their extension of BTL models, it is proved statistical errors in the order of $O(N^2\log(MN^2)/(MT))$. When $M = 1$, the statistical error matches the error bound in the state-of-the-art work for single source model (Negahban et al., 2016).

1.7.2 Adaptive Sampling of Heterogeneous Users in Active Ranking

In the context of heterogeneous rank aggregation problems, sources often have varying levels of accuracy when comparing pairs of items. This diversity in source accuracy suggests that a uniform querying strategy may not be optimal. While in the previous section, the problem of when the dataset is static how to efficiently infer the aggregated rank is addressed.

If we have the liberty to actively choose the pair to compare, thus opens up the possibility that can further save the data acquisition cost. To address this challenge, we proposed an active sampling strategy based on source elimination. This strategy estimates the ranking of items using noisy pairwise comparisons from multiple sources and improves the average accuracy of the sources by maintaining an active set. Specifically, a short history of source responses is maintained for a set of comparisons. When the inferred rank of these comparisons is estimated to be true with a high confidence, it is then used to calculate a reward based on the recorded responses. Then an upper confidence bound (UCB)-style elimination process is performed to remove inaccurate sources from active source set. Experiments on both synthetic and real-world datasets demonstrate that the adaptive sampling algorithm based on source elimination is much more sample efficient than baseline algorithms and can sometimes reach the performance of an source algorithm.

To reduce the analysis complexity of the algorithm, we assume that for each pair (i, j) for the same source u, the comparison probability is the same across all possible pairs, which means $\Delta_{i,j}^{(u)} := \Delta_u, \forall(i, j) \in$

$[N] \times [N]$. Then the best source can be defined as $u^* = \arg\max \Delta_{u^*}$. Additionally, we define a shorthand to reduce clutter:

$$F(x) = x^{-2}(\log\log x^{-1} + \log(N/\delta)). \tag{1.7.2}$$

Given the problem setup in Section 1.2 to rank items with sources. Let $\mathcal{C}$ denotes the sample complexity when the algorithm stops. It is proved that if the best source is known beforehand, the sample complexity is as follows:

$$\mathcal{C}_{u^*} = \Theta\big(N\Delta_{u^*}^{-2}\big(\log\log\Delta_{u^*}^{-1} + \log(N/\delta)\big)\big) = \Theta(NF(\Delta_{u^*})). \tag{1.7.3}$$

Define $\bar{\Delta} = \frac{1}{M}\sum_{u\in[M]}\Delta_u$ to be the average accuracy of all sources which corresponds to an algorithm that samples each information source uniformly at random. In this case, the sample complexity is:

$$\mathcal{C}_{\text{ave}} = \Theta(NF(\bar{\Delta}_0)) \tag{1.7.4}$$

Their proposed algorithm is able to achieve a result as follows:

$$\mathcal{C}_{\text{alg}} = \Theta(NF(\Delta_{u^*})) + o(N\big(F(\bar{\Delta}_0) - F(\Delta_{u^*})\big)) + o\big(N\big). \tag{1.7.5}$$

The last two terms are negligible when compared with the first term. Therefore, it can perform comparably efficiently as if the best source were known while enjoying an advantage over the naive algorithm with sample complexity $\mathcal{C}_{\text{ave}}$.

1.7.3 Active Ranking under Weak Stochastic Transitivity Assumption

Recovering the full ranking of N items under a more general setting, where only WST holds, while SST is not assumed to hold has not been studied in the field. A δ-correct algorithm, `Probe-Rank` proposed in this work actively infers the ranking from noisy pairwise comparisons. A sample complexity upper bound for `Probe-Rank` is proven as:

$$\widetilde{O}\bigg(N\sum_{i=2}^{N}\Delta_{\sigma(i),\sigma(i-1)}^{-2}\bigg), \tag{1.7.6}$$

which only depends on the preference probabilities between items that are adjacent in the true ranking. This improves the sample complexity of `Iterative-Insertion-Ranking` (IIR) (Ren et al., 2019) that depend on the preference probabilities for all pairs of items. The sample complexity of IIR is as follows, where $\Delta_i = \min_{j\in[N],i\neq j}\Delta_{i,j}$:

$$O\bigg(\sum_{i\in[N]}\Delta_i^{-2}\Big(\log\log\big(\Delta_i^{-1}\big) + \log(N/\delta)\Big)\bigg) \tag{1.7.7}$$

In extreme cases, there could be several item pairs that have comparison probability close to $1/2$ thus making the gap Δ_i close to zero, which in turn renders this bound vacuous. `Probe-Rank` thus mitigates this issue by only having the dependency on the adjacent items in final rankings.

1.7.4 Heterogeneous Active Ranking under WST Assumptions

We have discussed the algorithms for aggregating preferential data from a single or homogeneous source under the RUM assumption, the SST assumption, and the WST assumption. However, in today's world where a vast amount of data is involved, it is more desirable to use active or adaptive algorithms to reduce the cost of data acquisition.

Given the heterogeneous nature of the data sources, it is important to explore active ranking techniques that can handle such diversity. In previous sections, the active ranking problem under the SST assumption

was studied. However, their assumption about the pairwise comparison probabilities are uniform across each pair is quite restrictive and may not hold in real-world scenarios.

When comparing two arbitrary items i and j, denote the gap as $\Delta_{i,j}^{(u)} = |p_{i,j}^{(u)} - 1/2|$, it is easy to see a trivial solution is to construct an "average" source from a pool of sources by randomly sampling one source and query the pair (i, j).

Define the average gap as: $\bar{\Delta}_{i,j} := \frac{1}{M}\sum_{u=1}^{M} \Delta_{i,j}^{(u)}$, which represents a trivial aggregation of all sources by randomly sampling one source and querying the pair (i, j), any single-source algorithm can work under the multi-source setting as if the one source has accuracy gap $\bar{\Delta}_{i,j}$. We also introduce the average hardness as

$$\bar{H}_{i,j} := \frac{1}{(\bar{\Delta}_{i,j})^2}.$$

Next, we define

$$H_{i,j} := \frac{M}{\sum_{u=1}^{M}(\Delta_{i,j}^{(u)})^2}$$

as the hardness factor of a given pair (i, j) by deploying the technique provided by Saad et al. (2023). These hardness factors are directly proportional to the sample complexity. It is straightforward to prove that $H_{i,j} \leq \bar{H}_{i,j}$ by Cauchy-Schwartz inequality. As the sample complexity is directly proportional to the hardness factor, we can improve the efficiency of the algorithms for WST.

We propose to develop a new algorithm which has a bi-level design: at the higher level, it actively allocates comparison budgets to all undetermined pairs until the full ranking is recovered; at the lower level, it attempts to compare the pair of items and selects the more accurate sources simultaneously using the method that can achieve a better rate.

1.7.5 Score-Based Contextual Dueling Bandits

The optimal algorithm that matches the lower bound for the contextual bandits with RUM assumption has been well studied (Saha, 2021; Bengs et al., 2022). As we discussed before, the RUM assumption is quite restrictive. To remove the restriction, we can directly consider the score-based methods using Copeland score or Borda score. Copeland score targeted bandit without contextual information has been studied in the dueling bandits by Zoghi et al. (2015). And the optimality under the *Borda score* criteria has been adopted by several previous works (Jamieson et al., 2015; Falahatgar et al., 2017a; Heckel et al., 2018). In a work that studied the problem of regret minimization for adversarial dueling bandits (Saha et al., 2021). They proved a T-round Borda regret upper bound $\widetilde{O}(K^{1/3}T^{2/3})$. And a matching $\Omega(K^{1/3}T^{2/3})$ lower bound for stationary dueling bandits using Borda regret, where K is the number of arms. This result implies that for multi-armed stochastic dueling bandit, the minimum Borda regret must be of the order $\Omega(K^{1/3}T^{2/3})$.

To solve the problem when the set of arms are changing or even infinite, we proposed to take contextual information into account when the preference probabilities depend on a linear function of d-dimensional feature vectors. We proved upper and lower bounds on the Borda regret which is determined by d instead of K when $K \gg d$, or it can even be infinite. The regret lower bound is of order $\Omega(d^{2/3}T^{2/3})$ for the Borda regret minimization problem, where d is the dimension of contextual vectors and T is the time horizon. An explore-then-commit type algorithm for the stochastic setting is proposed, which has a nearly matching regret upper bound $\widetilde{O}(d^{2/3}T^{2/3})$. Empirical evaluations on both synthetic data and a simulated real-world environment conducted corroborated the theoretical analysis.

1.7.6 Variance-Aware Contextual Dueling Bandits

Existing dueling bandits algorithms do not consider the uncertainty of the pairwise comparison between dueling arms and suffer from an $\widetilde{O}(d\sqrt{T})$ regret, where d is the dimension of the context and T is the number of rounds. In an information-theoretic perspective, greater uncertainty or variance suggests a higher level of difficulty and lower information gain. We formulate this problem under the contextual dueling bandits framework, where the binary comparison of dueling arms is generated from a generalized linear model (GLM). We propose a new SupLinUCB-type algorithm that enjoys computational efficiency and a

variance-aware regret bound $\widetilde{O}\big(d\sqrt{\sum_{t=1}^{T}\sigma_t^2}+d\big)$, where σ_t is the variance of the pairwise comparison in round t, d is the dimension of the context vectors, and T is the time horizon. Our regret bound naturally aligns with the intuitive expectation — in scenarios where the comparison is deterministic, the algorithm only suffers from an $\widetilde{O}(d)$ regret. We perform empirical experiments on synthetic data to confirm the advantage of our method over previous variance-agnostic algorithms.

Part I

Efficient Ranking under Strong Stochastic Transitivity Assumption

Chapter 2

Heterogeneous Random Utility Models

2.1 Introduction

Conventional models of ranked data and aggregation algorithms that rely on them make the assumption that the data is either produced by a single user (data source)[1] or from a set of users (data sources) that are similar. In real-world datasets, however, users that provide the raw data are usually diverse with different levels of familiarity with the objects of interest, thus providing data that is not uniformly reliable and should not have equal influence on the final result. This is of particular importance in applications such as aggregating expert opinions for decision-making and aggregating annotations provided by workers in crowd sourcing settings.

We study the problem of rank aggregation for heterogeneous populations of users. We present a generalization of Random Utility Model (RUM), called the *Heterogeneous Random Utility Model* (HRUM), which allows users with different noise levels, as well as a certain class of adversarial users. Unlike previous efforts on rank aggregation for heterogeneous populations such as Chen et al. (2013); Kumar and Lease (2011), the proposed model maintains the generality of Thurstone's framework and thus also extends its special cases such as BTL and PL models. We evaluate the performance of the method using simulated data for different noise distributions. We also demonstrate that the proposed aggregation algorithm outperforms the state-of-the-art method for real datasets on evaluating the difficulty of English text and comparing the population of a set of countries.

2.2 Related Work

When restricted to comparing pairs of items, Thurstone's model reduces to the BTL model (Zermelo, 1929; Bradley and Terry, 1952; Luce, 1959; Hunter, 2004) if the noise follows the Gumbel distribution, and to the Thurstone Case V (TCV) model (Thurstone, 1927) if the noise is normally distributed. Recently, Negahban et al. (2012) proposed Rank Centrality, an iterative method with a random walk interpretation and showed that it performs as well as the maximum likelihood (ML) solution (Zermelo, 1929; Hunter, 2004) for BTL models and provided non asymptotic performance guarantees. Chen and Suh (2015) studied identifying the top-K candidates under the BTL model and its sample complexity.

Thurstone's model can also be used to describe data from comparisons of multiple items. Hajek et al. (2014) provided an upper bound on the error of the ML estimator and studied its optimality when data consists of partial rankings (as opposed to pairwise comparisons) under the PL model. Yu (2000) studied order statistics under the normal noise distribution with consideration of item confusion covariance and user perception shift in a Bayesian model. Weng and Lin (2011) proposed a Bayesian approximation method for game player ranking with results from two-team matches. Guiver and Snelson (2009) studied the ranking

[1] In this chapter, we use the term user to refer to any entity that provides ranked data. In specific applications other terms may be more appropriate, such as voter, expert, judge, worker, and annotator.

aggregation problem with partial ranking (PL model) in a Bayesian framework. However, due to the nature of Bayesian method, above mentioned work provided few theoretical analysis. Vojnovic and Yun (2016) studied the parameter estimation problem for Thurstone models where first choices among a set of alternatives are observed. Raman and Joachims (2014, 2015) proposed the peer grading methods for solving a similar problem as ours, while the generative models to aggregate partial rankings and pairwise comparisons are completely different. Very recently, Zhao et al. (2018) proposed the k-RUM model which assumes that the rank distribution has a mixture of k RUM components. They also provided the analyses of identifiability and efficiency of this model.

Almost all aforementioned works assume that all the data is provided by a single user or that all users have the same accuracy. However, this assumption is rarely satisfied in real-world datasets. The accuracy levels of different users are considered in Kumar and Lease (2011), which assumes that each user is correct with a certain probability and studies the problem via simulation methods such as naive Bayes and majority voting. In their pioneering work, Chen et al. (2013) studied rank aggregation in a crowd-sourcing environment for pairwise comparisons, modeled via the BTL or TCV model, where noisy BTL comparisons are assumed to be further corrupted. They are flipped with a probability that depends on the identity of the worker. The k-RUM model proposed by Zhao et al. (2018) considered a mixture of ranking distributions, without using extra information on who contributed the comparison, it may suffer from common mixture model issues.

2.3 Modeling Heterogeneous Ranking Data

2.3.1 The Heterogeneous Random Utility Model

In real-world applications, users often have different levels of expertise and some may even be adversarial. Therefore, it is natural for us to propose an extension of the Thurstone's model presented above, referred to as the *Heterogeneous Random Utility Model* (HRUM), which has the flexibility to reflect the different levels of expertise of different users. Specifically, we assume that each user has a different level of making mistakes in evaluating items, i.e., the evaluation noise of user u is controlled by a scaling factor $\gamma_u > 0$.

Throughout this chapter, we use n to denote the number of items and m to denote the number of users. The vector $\mathbf{s} = (s_1, \ldots, s_n)$ represents the true scores of all items, and $\gamma = (\gamma_1, \ldots, \gamma_m)$ represents the expertise levels of all users. The proposed model is then represented as follows:

$$z_i^u = s_i + \epsilon_i/\gamma_u. \tag{2.3.1}$$

Based on the estimated scores of each user for each item, the probability of a certain ranking of h items provided by user u is again given by (1.3.2). While this extension actually applies to both pairwise comparisons and multi-item orderings, we mainly focus on pairwise comparisons in this chapter.

When two items i and j are compared by user u, we denote by Y_{ij}^u the random variable representing the result,

$$Y_{ij}^u = \begin{cases} 1 & \text{if } i \succ j; \\ 0 & \text{if } i \prec j. \end{cases} \tag{2.3.2}$$

Let F denote the CDF of $\epsilon_j - \epsilon_i$, where ϵ_i and ϵ_j are two i.i.d. random variables. For the result Y_{ij}^u of comparison of i and j by user u, we have

$$\Pr(Y_{ij}^u = 1; s_i, s_j, \gamma_u) = \Pr(\epsilon_j - \epsilon_i < \gamma_u(s_i - s_j)) = F\left(\gamma_u(s_i - s_j)\right). \tag{2.3.3}$$

It is clear that the larger the value of γ_u, the more accurate the user is, since large $\gamma_u > 0$ increases the probability of preferring an item with higher score to one with lower score.

We now consider several special cases arising from specific noise distributions. First, if ϵ_i follows a Gumbel distribution with mean 0 and scale parameter 1, then we obtain the following *Heterogeneous BTL* (HBTL) model:

$$\log \Pr(Y_{ij}^u = 1; s_i, s_j, \gamma_u) = \log \frac{e^{\gamma_u s_i}}{e^{\gamma_u s_i} + e^{\gamma_u s_j}} = -\log(1 + \exp(-\gamma_u(s_i - s_j))), \tag{2.3.4}$$

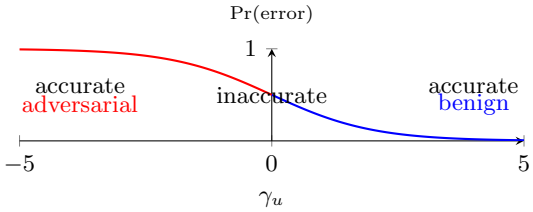


Figure 2.1: The effect of γ_u on the probability of error for a BTL comparison in which items have scores 0 and 1. In particular, for large negative values of γ_u, the user is accurate (with a high level of expertise) but adversarial.

which follows from the fact that the difference between two independent Gumbel random variables has the logistic distribution. We note that setting $\gamma_u = 1$ recovers the traditional BTL model (Bradley and Terry, 1952).

If ϵ_i follows the standard normal distribution, we obtain the following *Heterogeneous Thurstone Case V* (HTCV) model:

$$\log \Pr(Y_{ij}^u = 1; s_i, s_j, \gamma_u) = \log \Phi\left(\frac{\gamma_u(s_i - s_j)}{\sqrt{2}}\right), \tag{2.3.5}$$

where Φ is the CDF of the standard normal distribution. Again, when $\gamma_u = 1$, this reduces to Thurstone's Case V (TCV) model for pairwise comparisons (Thurstone, 1927).

Adversarial users: Under our heterogeneous framework, we can also model a certain class of adversarial users, whose goal is to make the estimated ranking be the opposite of the true ranking, so that, for example, an inferior item is ranked higher than the alternatives. We assume for adversarial users, the score of item i is $C - s_i$, for some constant C. Changing s_i to $C - s_i$ in (2.3.3) is equivalent to assuming the user has a negative accuracy γ_u. In this way, the accuracy of the user is determined by the magnitude $|\gamma_u|$ and its trustworthiness by $\text{sign}(\gamma_u)$, as illustrated in Figure 2.1. When adversarial users are present, this will facilitate optimizing the loss function, since instead of solving the combinatorial optimization problem of deciding which users are adversarial, we simply optimize the value of γ_u for each user.

One relevant work to ours is the CrowdBT algorithm proposed by Chen et al. (2013), where they also explored the accuracy level of different users in learning a global ranking. In particular, they assume that each user has a probability η_u of making mistakes in comparing items i and j:

$$\Pr(Y_{ij}^u = 1; s_i, s_j, \eta_u) = \eta_u \Pr(i \succ j) + (1 - \eta_u) \Pr(j \succ i), \tag{2.3.6}$$

where $\Pr(i \succ j)$ and $\Pr(j \succ i)$ follow the BTL model. This translates to introducing a parameter in the likelihood function to quantify the reliability of each pairwise comparison. This parameterization, however, deviates from the additive noise in Thurstonian models defined as in (1.3.1) such as BTL and Thurstone's Case V. Specifically, the Thurstonian model explains the noise observed in pairwise comparisons as resulting from the additive noise in estimating the latent item scores. Therefore, the natural extension of Thurstonian models to a heterogeneous population of users is to allow different noise levels for different users, as was done in (2.3.1). As a result, CrowdBT cannot be easily extended to settings where more than two items are compared at a time. In contrast, the model proposed here is capable to describe such generalizations of Thurstonian models, such as the PL model.

2.3.2 Rank Aggregation via Maximum Likelihood Estimation

In this section, we define the pairwise comparison loss function for the population of users and propose an efficient and effective optimization algorithm to minimize it. We denote by $\boldsymbol{Y}^u$ the matrix containing all pairwise comparisons Y_{ij}^u of user u on items i and j. The entries of $\boldsymbol{Y}^u$ are 0/1/?, where ? indicates that

the pair was not compared by the user. Furthermore, let $\mathcal{D}_u$ denote the set of all pairs (i, j) compared by user u. We define the loss function for each user u as

$$\begin{aligned}\mathcal{L}_u\left(\boldsymbol{s}, \gamma_u; \boldsymbol{Y}^u\right) &= -\frac{1}{k_u} \sum_{(i,j)\in\mathcal{D}_u} \log \Pr(Y_{ij}^u = 1 | s_i, s_j, \gamma_u) \\ &= -\frac{1}{k_u} \sum_{(i,j)\in\mathcal{D}_u} \log F\left(\gamma_u(s_i - s_j)\right),\end{aligned}$$

where $k_u = |\mathcal{D}_u|$ is the number of comparisons by user u. Then, the total loss function for m users is

$$\mathcal{L}\left(\boldsymbol{s}, \boldsymbol{\gamma}; \boldsymbol{Y}\right) = \frac{1}{m} \sum_{u=1}^{m} \mathcal{L}_u\left(\boldsymbol{s}, \gamma_u; \boldsymbol{Y}^u\right), \tag{2.3.7}$$

where $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_m)^\top$ and $\boldsymbol{Y} = (\boldsymbol{Y}^1, \ldots, \boldsymbol{Y}^m)$. We denote the unknown true score vector as $\boldsymbol{s}^*$ and the true accuracy vector as $\boldsymbol{\gamma}^*$. Given observation $\boldsymbol{Y}$, our goal is to recover $\boldsymbol{s}^*$ and $\boldsymbol{\gamma}^*$ via minimizing the loss function in (2.3.7). To ensure the identifiability of $\boldsymbol{s}^*$, we follow Negahban et al. (2016) to assume that $\mathbf{1}^\top \boldsymbol{s}^* = \sum_{i=1}^n s_i^* = 0$, where $\mathbf{1} \in \mathbb{R}^n$ is the all one vector. The following proposition shows that the loss function $\mathcal{L}$ is convex in $\boldsymbol{s}$ and in $\boldsymbol{\gamma}$ separately if the PDF of ϵ_i is log-concave.

Proposition 2.3.1. If the distribution of the noise ϵ_i in (2.3.1) is log-concave, then the loss function $\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}; \boldsymbol{Y})$ given in (2.3.7) is convex in $\boldsymbol{s}$, and in $\boldsymbol{\gamma}$ respectively.

The log-concave family includes many well-known distributions such as normal, exponential, Gumbel, gamma and beta distributions. In particular, the noise distributions used in BTL and Thurstone's Case V (TCV) models fall into this category. Although the loss function $\mathcal{L}$ is non convex with respect to the joint variable $(\boldsymbol{s}, \boldsymbol{\gamma})$, Proposition 2.3.1 inspires us to perform alternating gradient descent (Jain et al., 2013) on $\boldsymbol{s}$ and $\boldsymbol{\gamma}$ to minimize the loss function. As is shown in Algorithm 2.1, we alternating perform gradient descent update on $\boldsymbol{s}$ (or $\boldsymbol{\gamma}$) while fixing $\boldsymbol{\gamma}$ (or $\boldsymbol{s}$) at each iteration. In addition to the alternating gradient descent steps, we shift $\boldsymbol{s}^{(t)}$ in Line 4 of Algorithm 2.1 such that $\mathbf{1}^\top \boldsymbol{s}^{(t)} = 0$ to avoid the aforementioned identifiability issue of $\boldsymbol{s}^*$. After T iterations, given the output $\boldsymbol{s}^{(T)}$, the estimated ranking of the items is obtained by sorting $\{s_1^{(T)}, \ldots, s_n^{(T)}\}$ in descending order (item with the highest score in $\boldsymbol{s}^{(T)}$ is the most preferred).

Algorithm 2.1 HRUM with Alternating Gradient Descent

1: **input:** learning rates $\eta_1, \eta_2 > 0$, initial points $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ satisfying $\|\boldsymbol{s}^{(0)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(0)} - \boldsymbol{\gamma}^*\|_2^2 \le r$, number of iteration T, comparison results by users $\boldsymbol{Y}$.
2: **for** $t = 0, \ldots, T-1$ **do**
3: $\quad \widetilde{\boldsymbol{s}}^{(t+1)} = \boldsymbol{s}^{(t)} - \eta_1 \nabla_{\boldsymbol{s}} \mathcal{L}\big(\boldsymbol{s}^{(t)}, \boldsymbol{\gamma}^{(t)}; \boldsymbol{Y}\big)$
4: $\quad \boldsymbol{s}^{(t+1)} = (\mathbf{I} - \mathbf{1}\mathbf{1}^\top/n)\widetilde{\boldsymbol{s}}^{(t+1)}$
5: $\quad \boldsymbol{\gamma}^{(t+1)} = \boldsymbol{\gamma}^{(t)} - \eta_2 \nabla_{\boldsymbol{\gamma}} \mathcal{L}\big(\boldsymbol{s}^{(t)}, \boldsymbol{\gamma}^{(t)}; \boldsymbol{Y}\big)$
6: **end for**
7: **output:** $\boldsymbol{s}^{(T)}$, $\boldsymbol{\gamma}^{(T)}$.

As we will show in the next section, the convergence of Algorithm 2.1 to the optimal points $\boldsymbol{s}^*$ and $\boldsymbol{\gamma}^*$ is guaranteed if an initialization such that $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ are close to the unknown parameters is available. In practice, to initialize $\boldsymbol{s}$, we can use the solution provided by the rank centrality algorithm (Negahban et al., 2012) or start from uniform or random scores. In this chapter, we initialize $\boldsymbol{s}$ and $\boldsymbol{\gamma}$, as $\boldsymbol{s}^{(0)} = \mathbf{1}$ and $\boldsymbol{\gamma}^{(0)} = \mathbf{1}$. We note that multiplying $\boldsymbol{s}$ or $\boldsymbol{\gamma}$ by a negative constant does not alter the loss but reverses the estimated ranking. Implicit in our initialization is the assumption that the majority of the users are trustworthy and thus have positive γ. When data is sparse, there may be subsets of items that are not compared directly or indirectly. In such cases, regularization may be necessary, which is discussed in further detail in Section 2.5.

2.4 Theoretical Analysis

In this section, we provide the convergence analysis of Algorithm 2.1 for the general loss function defined in (2.3.7). Without loss of generality, we assume the number of observations $k_u = k$ for all users $u \in [m]$ throughout our analysis. Since there's no specific requirement on the noise distributions in the general HRUM model, to derive the linear convergence rate, we need the following conditions on the loss function $\mathcal{L}$, which are standard in the literature of alternating minimization (Jain et al., 2013; Zhu et al., 2017; Xu et al., 2017b,a; Zhang et al., 2018; Chen et al., 2018). Note that all these conditions can actually be verified once we specify the noise distribution in specific models. We provide the justifications of these conditions in the appendix.

Condition 2.4.1 (Strong Convexity)**.** $\mathcal{L}$ is μ_1-strongly convex with respect to $\boldsymbol{s} \in \mathbb{R}^n$ and μ_2-strongly convex with respect to $\boldsymbol{\gamma} \in \mathbb{R}^m$. In particular, there is a constant $\mu_1 > 0$ such that for all $\boldsymbol{s}, \boldsymbol{s}' \in \mathbb{R}^n$,

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) \geq \mathcal{L}(\boldsymbol{s}', \boldsymbol{\gamma}) + \langle \nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}', \boldsymbol{\gamma}), \boldsymbol{s} - \boldsymbol{s}' \rangle + \mu_1/2 \|\boldsymbol{s} - \boldsymbol{s}'\|_2^2.$$

And there is a constant $\mu_2 > 0$ such that for all $\boldsymbol{\gamma}, \boldsymbol{\gamma}' \in \mathbb{R}^m$, it holds

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) \geq \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}') + \langle \nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}'), \boldsymbol{\gamma} - \boldsymbol{\gamma}' \rangle + \mu_2/2 \|\boldsymbol{\gamma} - \boldsymbol{\gamma}'\|_2^2.$$

Condition 2.4.2 (Smoothness)**.** $\mathcal{L}$ is L_1-smooth with respect to $\boldsymbol{s} \in \mathbb{R}^n$ and L_2-smooth with respect to $\boldsymbol{\gamma} \in \mathbb{R}^m$. In particular, there is a constant $L_1 > 0$ such that for all $\boldsymbol{s}, \boldsymbol{s}' \in \mathbb{R}^n$, it holds

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) \leq \mathcal{L}(\boldsymbol{s}', \boldsymbol{\gamma}) + \langle \nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}', \boldsymbol{\gamma}), \boldsymbol{s} - \boldsymbol{s}' \rangle + L_1/2 \|\boldsymbol{s} - \boldsymbol{s}'\|_2^2.$$

And there is a constant $L_2 > 0$ such that for all $\boldsymbol{\gamma}, \boldsymbol{\gamma}' \in \mathbb{R}^m$, it holds

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) \leq \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}') + \langle \nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}'), \boldsymbol{\gamma} - \boldsymbol{\gamma}' \rangle + L_2/2 \|\boldsymbol{\gamma} - \boldsymbol{\gamma}'\|_2^2.$$

The next condition is a variant of the usual Lipschitz gradient condition. It is worth noting that the gradient is derived with respect to $\boldsymbol{s}$ (or $\boldsymbol{\gamma}$), while the upper bound is the difference of $\boldsymbol{\gamma}$ (or $\boldsymbol{s}$). This condition is commonly imposed and verified in the analysis of expectation-maximization algorithms (Wang et al., 2015) and alternating minimization (Jain et al., 2013).

Condition 2.4.3 (First-order Stability)**.** There are constants $M_1, M_2 > 0$ such that $\mathcal{L}$ satisfies

$$\begin{aligned} \|\nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}')\|_2 &\leq M_1 \|\boldsymbol{\gamma} - \boldsymbol{\gamma}'\|_2, \\ \|\nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}', \boldsymbol{\gamma})\|_2 &\leq M_2 \|\boldsymbol{s} - \boldsymbol{s}'\|_2, \end{aligned}$$

for all $\boldsymbol{s}, \boldsymbol{s}' \in \mathbb{R}^n$ and $\boldsymbol{\gamma}, \boldsymbol{\gamma}' \in \mathbb{R}^m$.

Note that the loss function in (2.3.7) is defined based on finitely many samples of observations. The next condition shows how close the gradient of the sample loss function is to the expected loss function.

Condition 2.4.4. Denote $\bar{\mathcal{L}}$ as the expected loss, where the expectation of $\mathcal{L}$ is taken over the random choice of the comparison pairs and the observation $\boldsymbol{Y}$. With probability at least $1 - 1/n$, we have

$$\begin{aligned} \|\nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{s}} \bar{\mathcal{L}}(\boldsymbol{s}, \boldsymbol{\gamma})\|_2 &\leq \epsilon_1(k, n), \\ \|\nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{\gamma}} \bar{\mathcal{L}}(\boldsymbol{s}, \boldsymbol{\gamma})\|_2 &\leq \epsilon_2(k, n), \end{aligned}$$

where n is the number of items and k is the number of observations for each user. In addition, $\epsilon_1(k, n)$ and $\epsilon_2(k, n)$ will go to zero when sample size k goes to infinity.

$\epsilon_1(k, n)$ and $\epsilon_2(k, n)$ in Condition 2.4.4 are also called the statistical errors (Wang et al., 2015; Xu et al., 2017a) between the sample version gradient and the expected (population) gradient.

Now we deliver our main theory on the linear convergence of Algorithm 2.1 for general HRUM models. Full proofs can be found in the appendix.

Theorem 2.4.5. For a general HRUM model, assume Conditions 2.4.1, 2.4.2, 2.4.3 and 2.4.4 hold and that $M_1, M_2 \leq \sqrt{\mu_1\mu_2}/4$. Denote that $\|\boldsymbol{s}^*\|_\infty = s_{\max}$ and $\|\boldsymbol{\gamma}^*\|_\infty = \gamma_{\max}$. Suppose the initialization guarantees that $\|\boldsymbol{s}^{(0)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(0)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2$,where $r = \min\{\mu_1/(2M_1), \mu_2/(2M_2)\}$. If we set the step size $\eta_1 = \eta_2 = \mu/(12(L^2 + M^2))$, where $L = \max\{L_1, L_2\}$, $\mu = \min\{\mu_1, \mu_2\}$ and $M = \max\{M_1, M_2\}$, then the output of Algorithm 2.1 satisfies

$$\|\boldsymbol{s}^{(T)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(T)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2\rho^T + \frac{\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2}{\mu^2}$$

with probability at least $1 - 1/n$, where the contraction parameter is $\rho = 1 - \mu^2/(48(L^2 + M^2))$.

Remark 2.4.6. Theorem 2.4.5 establishes the linear convergence of Algorithm 2.1 when the initial points are close to the unknown parameters. The first term on the right-hand side is called the optimization error, which goes to zero as iteration number t goes to infinity. The second term is called the statistical error of the HRUM model, which goes to zero when sample size mk goes to infinity. Hence, the estimation error of our proposed algorithm converges to the order of $O((\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2)/\mu^2)$ after $t = O(\log((\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2)/\mu^2 r^2)/\log\rho)$ iterations.

Note that the results in Theorem 2.4.5 hold for any general HRUM models with Algorithm 2.1 as a solver. In particular, if we run the alternating gradient descent algorithm on the HBTL and HTCV models proposed in Section 2.3, we will also obtain linear convergence rate to the true parameters up to a statistical error in the order of $O(n^2\log(mn^2)/(mk))$, which matches the state-of-the-art statistical error for such models (Negahban et al., 2016). We provide the implications of Theorem 2.4.5 on specific models in the following section.

2.4.1 Implications of Specific Models

Our Theorem 2.4.5 is for general HRUM models that satisfy Conditions 2.4.1, 2.4.2, 2.4.3 and 2.4.4. In this subsection, we will show that the linear convergence rate of Algorithm 2.1 can also be attained for specific models without assuming theses conditions when the random noise ϵ_i in (2.3.1) follows the Gumbel distribution and the Gaussian distribution respectively.

Heterogeneous BTL model

We first consider the model with Gumbel noise. Specifically, $\{\epsilon_i\}_{i=1,\ldots,n}$ follow the Gumbel distribution with mean 0 and scale parameter 1. Then we obtain the HBTL model defined in (2.3.4). The following corollary states the convergence result of Algorithm 2.1 for HBTL models.

Corollary 2.4.7. Consider the HBTL model in (2.3.4) and assume the sample size $k \geq n^2\log(mn)/m^2$. Let $\|\boldsymbol{s}^*\|_\infty = s_{\max}$, $\max_u|\gamma^{*u}| = \gamma_{\max}$ and $\min_u|\gamma^{*u}| = \gamma_{\min}$. Assume $\gamma_{\max}s_{\max} = C_0$ for a constant $C_0 \geq 1/2$ and

$$s_{\max} \leq \frac{\sqrt{m}\|\boldsymbol{s}^*\|_2}{n} \cdot \frac{\gamma_{\min}e^{5C_0}}{32\sqrt{2}\gamma_{\max}(1 + e^{5C_0})^2}.$$

Suppose the initialization points $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ satisfy that $\|\boldsymbol{s}^{(0)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(0)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2$, where $r = \min\{\|\boldsymbol{s}^*\|_2/2, \gamma_{\min}/2, s_{\max}, \sqrt{\gamma_{\max}s_{\max}}\}$. If we set the step size small enough such that

$$\eta_1 = \eta_2 < \frac{mne^{5C_0}\Gamma_1^2}{6(1 + e^{5C_0})^2(m\Gamma_2^4 + 32n^2C_0^2)},$$

where $\Gamma_1 = \min\{\gamma_{\min}/2, \|\boldsymbol{s}^*\|_2\}$ and $\Gamma_2 = \max\{2\gamma_{\max}, 2\|\boldsymbol{s}^*\|_2\}$, then the output of Algorithm 2.1 satisfies

$$\|\boldsymbol{s}^{(T)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(T)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2\rho^T + \frac{\Lambda n^2\log(4mn^2)}{mk}$$

with probability at least $1 - 1/n$, where $\rho = 1 - \eta(\mu - 6\eta(\Gamma_2^4/n^2 + 32C_0^2/m))/2$ and Λ is a constant which only which depends on $C_0, \gamma_{\max}$ and Γ_1.

Remark 2.4.8. According to Corollary 2.4.7, when the initial points $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ lie in a small neighborhood of the unknown parameter $\boldsymbol{s}^*, \boldsymbol{\gamma}^*$, the proposed algorithm converges linearly fast to a term in the order of $O(n^2 \log(mn^2)/(mk))$, which is called the statistical error of the HBTL model. Note that when $m = 1$, the statistical error reduces to $O(n^2 \log(n)/k)$, which matches the state-of-the-art estimation error bound for single user BTL model (Negahban et al., 2016). In addition, we assumed that $\|\boldsymbol{s}^*\|_\infty \lesssim O(\sqrt{m}/n\|\boldsymbol{s}^*\|_2)$ in order to derive the linear convergence of Algorithm 2.1. When m is in the same order of n, the requirement reduces to $\|\boldsymbol{s}^*\|_\infty \lesssim O(\|\boldsymbol{s}^*\|_2/\sqrt{n})$. This assumption is similar to the spikiness assumption in Agarwal et al. (2012); Negahban and Wainwright (2012), which ensures that there are not too many items that have zero or nearly zero scores.

Heterogeneous Thurstone Case V model

Now we consider the HRUM model with Gaussian noise. Assume that $\{\epsilon_i\}_{i=1,\dots,n}$ are i.i.d. from $N(0,1)$. Then the general HRUM model becomes HTCV model defined in (2.3.5), which generalizes the single user TCV model (Thurstone, 1927). Before we present the convergence results of Algorithm 2.1 for this model, we first remark some notations of the normal distribution to simplify the presentation. In particular, let $\Phi(x)$ be the CDF of standard normal distribution. We define $H(x) = (\Phi'(x)^2 - \Phi(x)\Phi''(x))/\Phi(x)^2$, which can be verified to be a monotonically decreasing function.

Corollary 2.4.9. Consider the HTCV model in (2.3.5) and assume the sample size $k \geq n^2 \log(mn)/m^2$. $s_{\max}$, $\gamma_{\max}$, $\gamma_{\min}$ and C_0 are defined the same as in Corollary 2.4.7. Assume $s_{\max}$ satisfies

$$s_{\max} \leq \frac{\sqrt{m}\|\boldsymbol{s}^*\|_2}{n} \cdot \frac{\gamma_{\min} H(5C_0)}{30\gamma_{\max}(\Phi(-5C_0)^{-1} + H(-5C_0))}.$$

Suppose the initialization points $\boldsymbol{s}^{(0)}$ and $\boldsymbol{\gamma}^{(0)}$ satisfy that $\|\boldsymbol{s}^{(0)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(0)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2$, where $r = \min\{\|\boldsymbol{s}^*\|_2/2, \gamma_{\min}/2, s_{\max}, \sqrt{\gamma_{\max} s_{\max}}\}$. If we set the step size

$$\eta_1 = \eta_2 < \frac{mn\Gamma_1^2 H(5C_0)}{6(m\Gamma_2^4 + 50n^2C_0^2)H(-5C_0)^2},$$

where $\Gamma_1 = \min\{\gamma_{\min}/2, \|\boldsymbol{s}^*\|_2\}$ and $\Gamma_2 = \max\{2\gamma_{\max}, 2\|\boldsymbol{s}^*\|_2\}$, then the output of Algorithm 2.1 satisfies

$$\|\boldsymbol{s}^{(T)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(T)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2\rho^T + \frac{\Lambda' n^2 \log(4mn^2)}{mk}$$

with probability at least $1 - 1/n$, where $\rho = 1 - \eta(\mu - 6\eta(\Gamma_2^4/n^2 + 32C_0^2/m))/2$ and Λ' is a constant which only depends on $C_0, \gamma_{\max}$ and Γ_1.

Remark 2.4.10. Corollary 2.4.9 suggests that under suitable initialization, Algorithm 2.1 enjoys a linear convergence rate when the random noise follows the standard normal distribution. The statistical error for the HTCV model is in the order of $O(n^2 \log(mn^2)/(mk))$. We again need the 'spikiness' assumption on the unknown score vector $\boldsymbol{s}^*$ in order to ensure the algorithm to find the true parameter. The results are almost the same as those of the HBTL model presented in Corollary 2.4.7 except that the constants in the HTCV model depends on the normal CDF Φ and its first and second derivatives.

2.4.2 Proof of the Generic Model

In this section, we provide the proof of Theorem 2.4.5 for general HSUM.

Proof of Theorem 2.4.5. According to the update in Algorithm 2.1 and the fact that $\mathbf{1}^\top \boldsymbol{s}^* = 0$, we have

$$\begin{aligned}
\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^*\|_2^2 &= \|(\mathbf{I} - \mathbf{1}\mathbf{1}^\top/n)(\widetilde{\boldsymbol{s}}^{(t+1)} - \boldsymbol{s}^*)\|_2^2 \\
&\leq \|\widetilde{\boldsymbol{s}}^{(t+1)} - \boldsymbol{s}^*\|_2^2 \\
&= \|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \eta_1^2\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)}, \boldsymbol{\gamma}^{(t)})\|_2^2 - 2\eta_1\langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)}, \boldsymbol{\gamma}^{(t)}), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle,
\end{aligned}$$

where the inequality comes from the fact that $\|\mathbf{I} - \mathbf{1}\mathbf{1}^\top/n\|_2 \leq 1$. We first bound the second term on the right hand side above

$$\begin{aligned}\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^{(t)})\|_2^2 &\leq 3\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^{(t)}) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^*)\|_2^2 + 3\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^*) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*)\|_2^2 \\ &\quad + 3\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*) - \nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*)\|_2^2 \\ &\leq 3M_1^2\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 + 3L_1^2\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + 3\epsilon_1(k,n)^2,\end{aligned}$$

where the first inequality is due to $\nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*) = \mathbf{0}$ and the second inequality is due to Conditions 2.4.2, 2.4.3, and 2.4.4. Now we bound the inner product term. Note that

$$\begin{aligned}&\langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^{(t)}), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle \\ &= \langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^{(t)}) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^{(t)}), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle + \langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^{(t)}) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle \\ &\quad + \langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*) - \nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle.\end{aligned}$$

By strong convexity (Condition 2.4.1) of $\mathcal{L}$ we have

$$\langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^{(t)}) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^{(t)}), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle \geq \mu_1\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2. \tag{2.4.1}$$

Applying Young's inequality and Condition 2.4.3, we obtain

$$\begin{aligned}|\langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^{(t)}) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle| &\leq \|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^{(t)}) - \nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*)\|_2 \cdot \|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2 \\ &\leq \frac{\alpha M_1^2}{2}\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 + \frac{1}{2\alpha}\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2,\end{aligned} \tag{2.4.2}$$

where $\alpha > 0$ is an arbitrarily chosen constant. In addition, by Condition 2.4.4 and Young's inequality we have

$$\begin{aligned}|\langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*) - \nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle| &\leq \|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*) - \nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s}^*,\boldsymbol{\gamma}^*)\|_2 \cdot \|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2 \\ &\leq \frac{1}{2\mu_1}\epsilon_1(k,n)^2 + \frac{\mu_1}{2}\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2.\end{aligned} \tag{2.4.3}$$

Combining (2.4.1), (2.4.2) and (2.4.3), we have

$$\langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}^{(t)},\boldsymbol{\gamma}^{(t)}), \boldsymbol{s}^{(t)} - \boldsymbol{s}^*\rangle \geq \frac{\mu_1\alpha - 1}{2\alpha}\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 - \frac{\alpha M_1^2}{2}\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 - \frac{1}{2\mu_1}\epsilon_1(k,n)^2.$$

Therefore, we have

$$\begin{aligned}\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^*\|_2^2 \leq \left(1 + 3L_1^2\eta_1^2 - \eta_1\left(\mu_1 - \frac{1}{\alpha}\right)\right)\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + M_1^2(3\eta_1^2 + \alpha\eta_1)\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 \\ + (3\eta_1^2 + \eta_1/\mu_1)\epsilon_1(k,n)^2.\end{aligned} \tag{2.4.4}$$

Similarly, we can bound $\|\boldsymbol{\gamma}^{(t+1)} - \boldsymbol{\gamma}^*\|_2^2$ as follows

$$\begin{aligned}\|\boldsymbol{\gamma}^{(t+1)} - \boldsymbol{\gamma}^*\|_2^2 \leq \left(1 + 3L_2^2\eta_2^2 - \eta_2\left(\mu_2 - \frac{1}{\beta}\right)\right)\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 + M_2^2(3\eta_2^2 + \beta\eta_2)\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 \\ + (3\eta_2^2 + \eta_2/\mu_2)\epsilon_2(k,n)^2,\end{aligned} \tag{2.4.5}$$

where $\beta > 0$ are arbitrarily chosen constants. In particular, set $\alpha = \mu_2/(4M_1^2)$, $\beta = \mu_1/(4M_2^2)$ and $\eta_1 = \eta_2 = \eta$. When $M_1, M_2 \leq \sqrt{\mu_1\mu_2}/4$, we have

$$\begin{aligned}\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t+1)} - \boldsymbol{\gamma}^*\|_2^2 &\leq (1 + 3(L_1^2 + M_2^2)\eta^2 - \mu_1\eta/2)\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 \\ &\quad + (1 + 3(L_2^2 + M_1^2)\eta_2^2 - \mu_2\eta/2)\|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 \\ &\quad + (3\eta^2 + \eta/\mu_1)\epsilon_1(k,n)^2 + (3\eta^2 + \eta/\mu_2)\epsilon_2(k,n)^2 \\ &\leq (1 + 3(L^2 + M^2)\eta^2 - \mu\eta/2)(\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2) \\ &\quad + (3\eta^2 + \eta/\mu)(\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2),\end{aligned} \tag{2.4.6}$$

where $L = \max\{L_1, L_2\}$, $M = \max\{M_1, M_2\}$ and $\mu = \min\{\mu_1, \mu_2\}$. Note that we have $\|\boldsymbol{s}_0 - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}_0 - \boldsymbol{\gamma}^*\|_2^2 \leq r^2$ by some initialization process. We can prove that $\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2$ for all $t \geq 0$ by induction. Specifically, assume it holds for t, then it suffices to ensure

$$(3\eta + 1/\mu)(\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2) \leq r^2(\mu/2 - 3(L^2 + M^2)\eta), \tag{2.4.7}$$

which holds when k is sufficiently large. Choosing η to be sufficiently small, we can ensure that $1 + 3(L^2 + M^2)\eta^2 - \mu\eta/2 \leq 1$. In particular, we can set $\eta = \mu/(12(L^2 + M^2))$, which implies

$$\begin{aligned}\|\boldsymbol{s}^{(t+1)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t+1)} - \boldsymbol{\gamma}^*\|_2^2 \leq \rho\big(\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2\big)\\ + (3\eta^2 + \eta/\mu)(\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2),\end{aligned}$$

with $\rho = 1 - \mu^2/(48(L^2 + M^2))$. Therefore, we have

$$\begin{aligned}\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 &\leq \rho^t\big(\|\boldsymbol{s}_0 - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}_0 - \boldsymbol{\gamma}^*\|_2^2\big) + \frac{3\eta^2 + \eta/\mu}{1-\rho}(\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2)\\ &\leq r^2\rho^t + \frac{\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2}{\mu^2},\end{aligned}$$

which completes the proof. □

2.4.3 Proofs of Specific Examples

In this section, we will provide the convergence analysis of Algorithm 2.1 for two specific examples with different noise distributions. In particular, we will show that Conditions 2.4.1 and 2.4.2 can be verified under these specific distributions. Recall the log-likelihood function

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}; \boldsymbol{Y}) = -\frac{1}{mk}\sum_{u=1}^{m}\sum_{(i,j)\in\mathcal{D}_u} \log F\left(\gamma_u(s_i - s_j); Y_{ij}^u\right). \tag{2.4.8}$$

For the ease of presentation, we will omit $\mathbf{Y}$ in the rest of the proof and assume that the observation set $\mathcal{D}_u$ is parametrized by $k = |\mathcal{D}_u|$ and vectors $\mathbf{a}_{l,u} \in \mathbb{R}^n$ for $l = 1, \ldots, k$, where each $\mathbf{a}_{l,u} = \mathbf{e}_{i_l} - \mathbf{e}_{j_l}$ for some pair of items (i_l, j_l) that is compared by user u and $\mathbf{e}_i$ is the natural basis. Then, we can rewrite the loss function in terms of vector $\boldsymbol{s}$ as follows

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) = -\frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k} \log F\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}; Y_{i_l j_l}^u\right). \tag{2.4.9}$$

Denote $g(x) = -\log F(x)$ for $x \in \mathbb{R}$. Then we can calculate the gradient of loss function $\mathcal{L}$ with respect to $\boldsymbol{s}$ and $\boldsymbol{\gamma}$.

$$\begin{aligned}\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) &= \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k} g'\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}\right)\gamma_u \mathbf{a}_{l,u},\\ \nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) &= \frac{1}{mk}\begin{bmatrix} \sum_{l=1}^{k} g'\left(\gamma_1 \mathbf{a}_{l,1}^\top \boldsymbol{s}\right)\mathbf{a}_{l,1}^\top \boldsymbol{s} \\ \vdots \\ \sum_{l=1}^{k} g'\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}\right)\mathbf{a}_{l,u}^\top \boldsymbol{s} \\ \vdots \end{bmatrix}.\end{aligned} \tag{2.4.10}$$

And the Hessian matrix can be calculated as

$$\begin{aligned}\nabla_{\boldsymbol{s}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) &= \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k} g''\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}\right)(\gamma_u)^2 \mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top,\\ \nabla_{\boldsymbol{\gamma}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) &= \frac{1}{mk}\mathrm{diag}\begin{bmatrix}\sum_{l=1}^{k} g''\left(\gamma_1 \mathbf{a}_{l,1}^\top \boldsymbol{s}\right)\mathbf{a}_{l,1}^\top \boldsymbol{s}\mathbf{a}_{l,1}^\top \boldsymbol{s}\\ \vdots \\ \sum_{l=1}^{k} g''\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}\right)\mathbf{a}_{l,u}^\top \boldsymbol{s}\mathbf{a}_{l,u}^\top \boldsymbol{s}\\ \vdots\end{bmatrix},\end{aligned} \tag{2.4.11}$$

where $\mathrm{diag}(\mathbf{x})$ is the diagonal matrix with diagonal entries given by $\mathbf{x}$.

Proof of Heterogeneous BTL model

Recall the definition in (2.3.7). The loss function can be written as

$$\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) = \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k} g\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}; Y_{i_l j_l}^u\right), \tag{2.4.12}$$

where $g(\cdot)$ is defined as

$$g(x; Y_{i_l j_l}^u) = -\log\frac{\exp(Y_{i_l j_l}^u x)}{1+\exp(x)}. \tag{2.4.13}$$

Therefore, the loss function of the HBTL model can be rewritten as follows:

$$\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) = \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k} \log\left(1+\exp(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s})\right) - Y_{i_l j_l}^u \gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}. \tag{2.4.14}$$

Recall the gradients and Hessian matrices calculated in (2.4.10) and (2.4.11). We need to calculate $g'(\cdot)$ and $g''(\cdot)$. In particular, we have

$$g'(x;Y) = \frac{-Y+(1-Y)\exp(x)}{1+\exp(x)}, \qquad g''(x;Y) = \frac{\exp(x)}{(1+\exp(x))^2}. \tag{2.4.15}$$

It is easy to verify that $g'(x)$ is monotonically increasing on $\mathbb{R}$. For any $|x|\le\theta$, we have

$$\frac{-1}{1+e^{-\theta}} \le g'(x;Y=1) \le \frac{-1}{1+e^{\theta}}, \qquad \frac{e^{-\theta}}{1+e^{-\theta}} \le g'(x;Y=0) \le \frac{e^{\theta}}{1+e^{\theta}}. \tag{2.4.16}$$

Furthermore, $g''(x) = g''(-x)$, $g''(x)$ is increasing on $(-\infty,0]$ and decreasing on $[0,\infty)$. Hence, for all $|x|\le\theta$, we have

$$e^{\theta}/(1+e^{\theta})^2 \le g''(x) \le g''(0) = 1/4. \tag{2.4.17}$$

We can further show that the following lemmas hold, which validates Conditions 2.4.1, 2.4.2, 2.4.3 and 2.4.4 used in the convergence analysis.

The first two lemmas verify the strong convexity and smoothness of $\mathcal{L}$ with respect to $\boldsymbol{s}$ and $\boldsymbol{\gamma}$ respectively.

Lemma 2.4.11. Suppose the noise ϵ follows the Gumbel distribution and the sample size $mk \ge 64(\gamma_{\max}+r)^2/(\gamma_{\min}-r)^2 n\log n$. Let $r \le \min\{s_{\max}, \sqrt{\gamma_{\max}s_{\max}}\}$, for all $\boldsymbol{s},\boldsymbol{s}' \in \mathbb{R}^n, \boldsymbol{\gamma}\in\mathbb{R}^m$ such that $\|\boldsymbol{s}-\boldsymbol{s}^*\|_2 \le r$, $\|\boldsymbol{s}'-\boldsymbol{s}^*\|_2 \le r$ and $\|\boldsymbol{\gamma}-\boldsymbol{\gamma}^*\|_2 \le r$, we have

$$\begin{aligned}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) &\ge \mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma}) + \langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma}), \boldsymbol{s}-\boldsymbol{s}'\rangle + \frac{\mu_1}{2}\|\boldsymbol{s}-\boldsymbol{s}'\|_2^2,\\ \mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) &\le \mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma}) + \langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma}), \boldsymbol{s}-\boldsymbol{s}'\rangle + \frac{L_1}{2}\|\boldsymbol{s}-\boldsymbol{s}'\|_2^2,\end{aligned}$$

where the coefficients are defined as

$$\mu_1 = \frac{(\gamma_{\min} - r)^2 e^{5\gamma_{\max} s_{\max}}}{n(1 + e^{5\gamma_{\max} s_{\max}})^2}, \qquad L_1 = \frac{(\gamma_{\max} + r)^2}{n}.$$

Lemma 2.4.12. Suppose the noise ϵ follows the Gumbel distribution and the sample size satisfies $k \geq 18(s_{\max} + r)^4 n^2/(m^2(\|\boldsymbol{s}^*\|_2 + r)^4)\log(mn)$. Let $r \leq \min\{s_{\max}, \sqrt{\gamma_{\max} s_{\max}}\}$, for all $\boldsymbol{s} \in \mathbb{R}^n, \boldsymbol{\gamma}, \boldsymbol{\gamma}' \in \mathbb{R}^m$ such that $\|\boldsymbol{s} - \boldsymbol{s}^*\|_2 \leq r$, $\boldsymbol{s}^\top \mathbf{1} = 0$, and $\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^*\|_2 \leq r, \|\boldsymbol{\gamma}' - \boldsymbol{\gamma}^*\|_2 \leq r$, we have with probability at least $1 - 1/n$ that

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) \geq \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}') + \langle \nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}'), \boldsymbol{\gamma} - \boldsymbol{\gamma}' \rangle + \frac{\mu_2}{2}\|\boldsymbol{\gamma} - \boldsymbol{\gamma}'\|_2^2,$$
$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) \leq \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}') + \langle \nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}'), \boldsymbol{\gamma} - \boldsymbol{\gamma}' \rangle + \frac{L_2}{2}\|\boldsymbol{\gamma} - \boldsymbol{\gamma}'\|_2^2,$$

where the coefficients are defined as

$$\mu_2 = \frac{(\|\boldsymbol{s}^*\|_2 + r)^2 e^{5\gamma_{\max} s_{\max}}}{n(1 + e^{5\gamma_{\max} s_{\max}})^2}, \quad L_2 = \frac{(\|\boldsymbol{s}^*\|_2 + r)^2}{n}.$$

Lemma 2.4.13. Let $r \leq \min\{s_{\max}, \sqrt{\gamma_{\max} s_{\max}}\}$, for all $\boldsymbol{s} \in \mathbb{R}^n, \boldsymbol{\gamma} \in \mathbb{R}^m$ such that $\|\boldsymbol{s} - \boldsymbol{s}^*\|_2 \leq r, \|\boldsymbol{s}' - \boldsymbol{s}^*\|_2 \leq r$ and $\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^*\|_2 \leq r, \|\boldsymbol{\gamma}' - \boldsymbol{\gamma}^*\|_2 \leq r$, we have

$$\|\nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}')\|_2 \leq \frac{\sqrt{2}(1 + 2\gamma_{\max} s_{\max})}{\sqrt{m}}\|\boldsymbol{\gamma} - \boldsymbol{\gamma}'\|_2,$$
$$\|\nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}', \boldsymbol{\gamma})\|_2 \leq \frac{\sqrt{2}(1 + 2\gamma_{\max} s_{\max})}{\sqrt{m}}\|\boldsymbol{s} - \boldsymbol{s}'\|_2.$$

Lemma 2.4.14. Let $r \leq \min\{s_{\max}, \sqrt{\gamma_{\max} s_{\max}}\}$, for all $\boldsymbol{s} \in \mathbb{R}^n, \boldsymbol{\gamma} \in \mathbb{R}^m$ such that $\|\boldsymbol{s} - \boldsymbol{s}^*\|_2 \leq r$ and $\|\boldsymbol{\gamma} - \boldsymbol{\gamma}^*\|_2 \leq r$. Denote $\bar{\mathcal{L}}$ as the expected loss which takes expectation of $\mathcal{L}$ over the random choice of comparison pair. We have

$$\|\nabla_{\boldsymbol{s}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{s}} \bar{\mathcal{L}}(\boldsymbol{s}, \boldsymbol{\gamma})\|_2 \leq \epsilon_1(k, n) := \frac{2(\gamma_{\max} + r)}{1 + e^{-5\gamma_{\max} s_{\max}}}\sqrt{\frac{2\log(2n)}{mk}},$$
$$\|\nabla_{\boldsymbol{\gamma}} \mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}) - \nabla_{\boldsymbol{\gamma}} \bar{\mathcal{L}}(\boldsymbol{s}, \boldsymbol{\gamma})\|_2 \leq \epsilon_2(k, n) := \frac{10\gamma_{\max} s_{\max}}{1 + e^{5\gamma_{\max} s_{\max}}}\sqrt{\frac{2\log(2mn)}{mk}},$$

holds with probability at least $1 - 1/n$.

Proof of Corollary 2.4.7. Now we prove the convergence of Algorithm 2.1 for Gumbel noise. Our proof will be similar to that of Theorem 2.4.5. In particular, we only need to verify that Conditions 2.4.1, 2.4.2, 2.4.3 and 2.4.4 hold when the noise follows a Gumbel distribution. According to Lemmas 2.4.11 and 2.4.12, we know that $\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma})$ is μ_1-strongly convex and L_1-smooth with respect to $\boldsymbol{s}$, and is μ_2-strongly convex and L_2-smooth with respect to $\boldsymbol{\gamma}$. More specifically, when $mk \geq 64n\log(n)$, we have

$$\mu_1 \geq (\gamma_{\min} - r)^2 e^{5C_0}/(n(1 + e^{5C_0})^2), \qquad L_1 \leq (\gamma_{\max} + r)^2/n, \tag{2.4.18}$$

where we use the fact that $\gamma_{\max} s_{\max} = C_0$. In addition, note that $s_{\max} \leq \sqrt{m}/n\|\boldsymbol{s}^*\|_s$ and $\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\| \leq r$. Hence if $mk \geq 18\log(mn)$, we have

$$\mu_2 \geq (\|\boldsymbol{s}^*\|_2 + r)^2 e^{5C_0}/(n(1 + e^{5C_0})^2), \qquad L_2 \leq (\|\boldsymbol{s}^*\|_2 + r)^2/n \tag{2.4.19}$$

By Lemma 2.4.13 and the assumption that $C_0 \geq 1/2$, we know that $\mathcal{L}$ satisfies the first-order stability (Condition 2.4.3) with $M_1 = M_2 = 4\sqrt{2}\gamma_{\max} s_{\max}/\sqrt{m}$. Note that by assumption, we have

$$s_{\max} \leq \frac{\gamma_{\min} e^{2C_0}}{16\sqrt{2}\gamma_{\max}(1 + e^{2C_0})^2}\frac{\sqrt{m}\|\boldsymbol{s}^*\|_2}{n}.$$

This immediately implies that $M = M_1 = M_2 \leq \sqrt{\mu_1\mu_2}/4$. Therefore, by similar arguments as in the proof of Theorem 2.4.5, we need to set step sizes $\eta_1 = \eta_2 = \eta < \mu/(6(L^2 + M^2))$, where $\mu = \min\{\mu_1, \mu_2\}$, $L = \max\{L_1, L_2\}$. In fact, it suffices to set

$$\eta < \frac{mne^{5C_0}\Gamma_1^2}{6(1+e^{5C_0})^2(m\Gamma_2^4 + 32n^2C_0^2)},$$

with $\Gamma_1 = \min\{\gamma_{\min}/2, \|\boldsymbol{s}^*\|_2\}$ and $\Gamma_2 = \max\{2\gamma_{\max}, 2\|\boldsymbol{s}^*\|_2\}$. We thus obtain

$$\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2\rho^t + \frac{\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2}{\mu^2} \leq r^2\rho^t + \frac{\Lambda n^2 \log(4mn^2)}{mk},$$

where $\rho = 1 - \eta(\mu - 6\eta(\Gamma_2^4/n^2 + 32C_0^2/m))/2$ and the last inequality comes from Lemma 2.4.14 with the constant Λ defined as follows:

$$\Lambda = \max\left\{\frac{200C_0^2(1+e^{5C_0})^2}{\Gamma_1^4 e^{10C_0}}, \frac{8(\gamma_{\max}+r)^2(1+e^{5C_0})^4}{\Gamma_1^4(1+e^{-5C_0})^2}\right\}.$$

This completes the proof. □

Proof of Heterogeneous Thurstone Case V model

In this subsection, we provide the analysis of our algorithm when the noise ϵ_i follows a Gaussian distribution, which results in the Thurstone model. The log-likelihood function can be written as

$$\mathcal{L}(\boldsymbol{s}, \boldsymbol{\gamma}; \boldsymbol{Y}) = \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k} g(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}; Y_{i_l j_l}^u). \tag{2.4.20}$$

with $g(\cdot)$ defined as $g(x) = -\log\Phi(x)$ with $\Phi(\cdot)$ be the CDF of the standard normal distribution. Note that $\Pr(Y_{i_l j_l}^u = 1) = \Phi(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s})$ and $\Pr(Y_{i_l j_l}^u = 0) = 1 - \Phi(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}) = \Phi(-\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s})$. Thus we can write $g(\cdot)$ as $g(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}; Y_{i_l j_l}^u) = -\log\Phi((2Y_{i_l j_l}^u - 1)\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s})$. Note that $(2Y-1)^2 = 1$, we have

$$g'(x;Y) = -\frac{(2Y-1)\Phi'(x)}{\Phi(x)}, \qquad g''(x;Y) = \frac{\Phi'(x)^2 - \Phi(x)\Phi''(x)}{\Phi(x)^2}.$$

In order to bound $g'(x)$ and $g''(x)$, we first calculate the derivatives of $\Phi(x)$ as follows:

$$\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \mathrm{d}z, \qquad \Phi'(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}, \qquad \Phi''(x) = \frac{-x}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}. \tag{2.4.21}$$

For any $\theta > 0$ such that $|x| \leq \theta$, we have

$$\frac{e^{-\theta^2/2}}{\sqrt{2\pi}\Phi(\theta)} \leq |g'(x)| \leq \frac{1}{\sqrt{2\pi}\Phi(-\theta)}.$$

We can verify that $g''(x)$ is monotonically decreasing on $\mathbb{R}^d$ and $g''(x) > 0$ also always hold. Thus for all $|x| \leq \theta$, we have $g''(\theta) \leq g''(x) \leq g''(-\theta)$.

Proof of Corollary 2.4.9. Recall the derivation of the gradient in (2.4.10) and the Hessian in (2.4.11) of the loss function $\mathcal{L}$. In order to verify Conditions 2.4.1, 2.4.2, 2.4.3 and 2.4.4, we only need the upper and lower bounds of $g'(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}; Y_{i_l j_l}^u)$ for all $u = 1, \ldots, m$ and $l = 1, \ldots, k$. Therefore, using exactly the same proof techniques as in Section 2.4.3, we can also establish strong convexity, smoothness, first-order stability and the statistical error bound for sample loss function $\mathcal{L}$ when the noise ϵ follows the standard normal distribution. We omit the proof since it is the same as that of the Gumbel case. We can verify that $\mathcal{L}$ is μ_1-strongly convex and L_1-smooth with respect to $\boldsymbol{s}$, and is μ_2-strongly convex and L_2-smooth with respect to $\boldsymbol{\gamma}$. The coefficient parameters are defined as $\mu_1 = (\gamma_{\min} - r)^2 H(5C_0)/n, L_1 = (\gamma_{\max} + r)^2 H(-5C_0)/n$,

$\mu_2 = (\|\boldsymbol{s}^*\|_2 + r)^2 H(5C_0)/n$ and $L_2 = (\|\boldsymbol{s}^*\|_2 + r)^2 H(-5C_0)/n$. Note that $H(x)$ is a function defined based on the normal CDF $\Phi(\cdot)$:

$$H(x) = [\Phi'(x)^2 - \Phi(x)\Phi''(x)]/\Phi(x)^2,$$

where Φ, Φ', Φ'' are defined in (2.4.21). The loss function $\mathcal{L}$ also satisfies Condition 2.4.3 with $M = M_1 = M_2 = (1/\Phi(-5C_0) + 5\sqrt{2\pi}H(-5C_0)\gamma_{\max}s_{\max})/\sqrt{m\pi}$. In order to make sure that $M \leq \sqrt{\mu_1\mu_2}/4$, we only need $s_{\max} \leq \sqrt{\pi}\gamma_{\min}H(5C_0)/[4\gamma_{\max}(2/\Phi(-5C_0)) + 5\sqrt{2\pi}H(-5C_0)] \cdot \sqrt{m}\|\boldsymbol{s}^*\|_2/n$. Therefore, by Theorem 2.4.5, if we choose step sizes $\eta_1 = \eta_2 = \eta$ such that

$$\eta < \frac{mn\Gamma_1^2 H(5C_0)}{6(m\Gamma_2^4 + 50n^2C_0^2)H(-5C_0)^2},$$
$$\text{with } \Gamma_2 = \min\{\gamma_{\min}/2, \|\boldsymbol{s}^*\|_2\}, \quad \Gamma_2 = \max\{2\gamma_{\max}, 2\|\boldsymbol{s}^*\|_2\},$$

then we are able to obtain the following convergence result:

$$\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2\rho^t + \frac{\epsilon_1(k,n)^2 + \epsilon_2(k,n)^2}{\mu^2}, \tag{2.4.22}$$

where $\mu = \Gamma_1^2 H(5C_0)/n$, $\rho = 1 - \eta(\mu - 6\eta(\Gamma_2^4/n^2 + 32C_0^2/m))/2$ and $\epsilon_1(k,n), \epsilon_2(k,n)$ are the statistical error bounds. Similar to the proof of Lemma 2.4.14, we know that $\epsilon_1(k,n) = (\gamma_{\max}+r)/(\sqrt{\pi}\Phi(-5C_0))\sqrt{2\log(2n)/(mk)}$ and $\epsilon_2(k,n) = 10\gamma_{\max}s_{\max}/(\sqrt{\pi}\Phi(-5C_0))\sqrt{\log(2mn)/(mk)}$. Plugging these two bounds into (2.4.22) yields

$$\|\boldsymbol{s}^{(t)} - \boldsymbol{s}^*\|_2^2 + \|\boldsymbol{\gamma}^{(t)} - \boldsymbol{\gamma}^*\|_2^2 \leq r^2\rho^t + \frac{\Lambda' n^2 \log(4mn^2)}{mk},$$

which holds with probability at least $1 - 1/n$, where Λ' is a constant defines as follows.

$$\Lambda' = \frac{2\max\{(\gamma_{\max} + r)^2, 50C_0^2\}}{\pi\Gamma_1^4 H(5C_0)^2\Phi(-5C_0)^2}.$$

This completes the proof. □

2.4.4 Proofs of Technical Lemmas

In this section, we provide the proofs of technical lemmas used in the previous section.

Proof of Lemma 2.4.11

We first lay down the following useful lemma.

Lemma 2.4.15. (Tropp, 2012) Consider a sequence of i.i.d. random matrices $\{\mathbf{X}_k\}$ in $\mathbb{R}^{d\times d}$ with $\mathbb{E}[\mathbf{X}_k] = \mathbf{0}$ and $\|\mathbf{X}_k\|_2 \leq R$. Then for all $t \geq 0$

$$\Pr\left(\left\|\sum_k \mathbf{X}_k\right\| \geq t\right) \leq d\exp\left(-\frac{t^2}{2\sigma^2 + 2Rt/3}\right),$$

where $\sigma^2 = \|\sum_k \mathbb{E}[\mathbf{X}_k^2]\|_2$.

Proof of Lemma 2.4.11. Using Taylor expansion, we have

$$\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) = \mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma}) + \langle\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma}), \boldsymbol{s} - \boldsymbol{s}'\rangle + \frac{1}{2}(\boldsymbol{s} - \boldsymbol{s}')^\top\nabla_{\boldsymbol{s}}^2\mathcal{L}(\widetilde{\boldsymbol{s}},\boldsymbol{\gamma})(\boldsymbol{s} - \boldsymbol{s}'),$$

where $\widetilde{\boldsymbol{s}} = \boldsymbol{s} + \theta(\boldsymbol{s}' - \boldsymbol{s})$ for some $\theta \in (0,1)$. In order to show the strong convexity and smoothness of $\mathcal{L}$, we need to bound the minimal and maximum eigenvalues of $\nabla_{\boldsymbol{s}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})$. Note that $\boldsymbol{s},\boldsymbol{\gamma}$ lie in a neighborhood with radius r of the true parameters $\boldsymbol{s}^*,\boldsymbol{\gamma}^*$ respectively. When $r \leq \min\{s_{\max}, \sqrt{\gamma_{\max}s_{\max}}\}$, we have

$$|\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}| \leq |(\gamma_u - \gamma_u^*)\mathbf{a}_{l,u}^\top(\boldsymbol{s} - \boldsymbol{s}^*)| + |\gamma_u^*\mathbf{a}_{l,u}^\top(\boldsymbol{s} - \boldsymbol{s}^*)| + |\gamma_u^*\mathbf{a}_{l,u}^\top\boldsymbol{s}^*| \leq 5\gamma_{\max}s_{\max}. \tag{2.4.23}$$

For any $\boldsymbol{\Delta} \in \mathbb{R}^n$, we have

$$\begin{aligned}\frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\frac{(\gamma_u)^2\exp(5\gamma_{\max}s_{\max})}{(1+\exp(5\gamma_{\max}s_{\max}))^2}\boldsymbol{\Delta}^\top\mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top\boldsymbol{\Delta} &\le \boldsymbol{\Delta}^\top\nabla_{\boldsymbol{s}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})\boldsymbol{\Delta}\\ &= \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}g''\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)(\gamma_u)^2\boldsymbol{\Delta}^\top\mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top\boldsymbol{\Delta}\\ &\le \frac{1}{4mk}\sum_{u=1}^{m}\sum_{l=1}^{k}(\gamma_u)^2\boldsymbol{\Delta}^\top\mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top\boldsymbol{\Delta},\end{aligned}$$

where we used the monotonicity of g''. Since $\mathbf{a}_{l,u} = \mathbf{e}_{i_l} - \mathbf{e}_{j_l}$ and i_l, j_l are uniformly distributed, we have $\mathbb{E}[\mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top] = \mathbb{E}[\mathbf{e}_{i_l}\mathbf{e}_{i_l}^\top + \mathbf{e}_{j_l}\mathbf{e}_{j_l}^\top - \mathbf{e}_{i_l}\mathbf{e}_{j_l}^\top - \mathbf{e}_{j_l}\mathbf{e}_{i_l}^\top] = 2/n\mathbf{I} - 2/n(\mathbf{1}\mathbf{1}^\top/n)$. We define

$$\mathbf{X}_{l,u} = (\gamma^u)^2\left[\mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top - \frac{2(\mathbf{I}-\mathbf{1}\mathbf{1}^\top/n)}{n}\right], \quad \mathbf{L} = \frac{2(\mathbf{I}-\mathbf{1}\mathbf{1}^\top/n)}{n}. \tag{2.4.24}$$

Thus we have $\mathbb{E}[\mathbf{X}_{l,u}] = \mathbf{0}$. Furthermore, we have $\|\mathbf{X}_{l,u}\|_2 \le 2(\gamma_{\max}+r)^2$ and $\mathbb{E}[\mathbf{X}_{l,u}^2] \le 4(\gamma_{\max}+r)^4(n-1)/n^2(\mathbf{I}-\mathbf{1}\mathbf{1}^\top/n)$. Applying Lemma 2.4.15 yields

$$\begin{aligned}\Pr\left(\left\|\frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\mathbf{X}_{l,u}\right\|_2 \ge t\right) &\le 2n\exp\left(\frac{-t^2}{8(\gamma_{\max}+r)^4(n-1)/(n^2mk) + 4t(\gamma_{\max}+r)^2/(3mk)}\right)\\ &\le 2n\exp\left(\frac{-t^2}{8(\gamma_{\max}+r)^4/(nmk) + 4t(\gamma_{\max}+r)^2/(3mk)}\right),\end{aligned}$$

which implies that

$$\begin{aligned}\left\|\frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\mathbf{X}_{l,u}\right\|_2 &\le \frac{8(\gamma_{\max}+r)^2\log n}{3mk} + 4(\gamma_{\max}+r)^2\sqrt{\frac{\log n}{nmk}}\\ &\le 8(\gamma_{\max}+r)^2\sqrt{\frac{\log n}{nmk}}\end{aligned}$$

holds with probability at least $1-1/n$, where the last inequality holds when $mk \ge 4/9n\log n$. Therefore, we have

$$\|\nabla_{\boldsymbol{s}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})\|_2 \le (\gamma_{\max}+r)^2\left(\frac{1}{2n} + 2\sqrt{\frac{\log n}{nmk}}\right) \le \frac{(\gamma_{\max}+r)^2}{n}.$$

On the other hand, for any $\boldsymbol{\Delta} \in \mathbb{R}^n$ such that $\boldsymbol{\Delta}^\top\mathbf{1} = 0$, we have

$$\frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\boldsymbol{\Delta}^\top\mathbf{X}_{l,u}\boldsymbol{\Delta} \ge -8\gamma_{\max}^2\sqrt{\frac{\log n}{nmk}}\|\boldsymbol{\Delta}\|_2^2,$$

which implies

$$\boldsymbol{\Delta}^\top\nabla_{\boldsymbol{s}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})\boldsymbol{\Delta} \ge \left(\frac{2(\gamma_{\min}-r)^2}{n} - 8(\gamma_{\max}+r)^2\sqrt{\frac{\log}{nmk}}\right)\|\boldsymbol{\Delta}\|_2^2.$$

Therefore, when k is sufficiently large such that $mk \ge 64(\gamma_{\max}+r)^2/(\gamma_{\min}-r)^2 n\log n$, we have

$$\lambda_{\min}\left(\nabla_{\boldsymbol{s}}^2\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})\right) \ge \frac{(\gamma_{\max}-r)^2e^{5\gamma_{\max}s_{\max}}}{n(1+e^{5\gamma_{\max}s_{\max}})^2}.$$

This completes the proof. □

Proof of Lemma 2.4.12

Proof. Using Taylor expansion, we get

$$\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) = \mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}') + \langle \nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}'), \boldsymbol{\gamma}-\boldsymbol{\gamma}'\rangle + \frac{1}{2}(\boldsymbol{\gamma}-\boldsymbol{\gamma}')^\top \nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\widetilde{\boldsymbol{\gamma}})(\boldsymbol{\gamma}-\boldsymbol{\gamma}'), \tag{2.4.25}$$

where $\widetilde{\boldsymbol{\gamma}} = \boldsymbol{\gamma} + \theta(\boldsymbol{\gamma}'-\boldsymbol{\gamma})$ for some $\theta \in (0,1)$. Recall the Hessian matrix with respect to $\boldsymbol{\gamma}$:

$$\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) = \frac{1}{mk}\mathrm{diag}\begin{bmatrix} \sum_{l=1}^k g''\left(\gamma_1 \mathbf{a}_{l,1}^\top \boldsymbol{s}\right)\mathbf{a}_{l,1}^\top \boldsymbol{s}\mathbf{a}_{l,1}^\top \boldsymbol{s} \\ \vdots \\ \sum_{l=1}^k g''\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}\right)\mathbf{a}_{l,u}^\top \boldsymbol{s}\mathbf{a}_{l,u}^\top \boldsymbol{s} \\ \vdots \end{bmatrix}.$$

For any fixed u, we denote $X_{l,u} = \mathbf{a}_{l,u}^\top \boldsymbol{s}\mathbf{a}_{l,u}^\top \boldsymbol{s} - \boldsymbol{s}^\top \mathbf{L}\boldsymbol{s}$, where $\mathbf{L}$ is defined as in (2.4.24). Recall the calculation of g'' in (2.4.15),(2.4.17) and that $|\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}| \le 5\gamma_{\max} s_{\max}$ by (2.4.23), we have

$$\frac{e^{5\gamma_{\max}s_{\max}}}{(1+e^{5\gamma_{\max}s_{\max}})^2} \le g''\left(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s}\right) = \frac{\exp(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s})}{\left(1+\exp(\gamma_u \mathbf{a}_{l,u}^\top \boldsymbol{s})\right)^2} \le \frac{1}{4}.$$

Since $\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})$ is a diagonal matrix, the eigenvalues of $\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})$ can be bounded by

$$\begin{aligned} \frac{e^{5\gamma_{\max}s_{\max}}}{(1+e^{5\gamma_{\max}s_{\max}})^2} \min_u \frac{1}{mk}\sum_{l=1}^k \left(\mathbf{a}_{l,u}^\top \boldsymbol{s}\right)^2 &\le \lambda_{\min}(\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(,\boldsymbol{\gamma})) \\ &\le \lambda_{\max}(\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(,\boldsymbol{\gamma})) \\ &\le \frac{1}{4}\max_u \frac{1}{mk}\sum_{l=1}^k \left(\mathbf{a}_{l,u}^\top \boldsymbol{s}\right)^2. \end{aligned} \tag{2.4.26}$$

Since $\boldsymbol{s}^\top \mathbf{1} = 0$, it is easy to verify $\mathbb{E}[X_{l,u}] = \mathbb{E}[\boldsymbol{s}^\top(\mathbf{a}_{l,u}\mathbf{a}_{l,u}^\top - \mathbf{L})\boldsymbol{s}] = 0$ and $|X_{l,u}| \le 6(s_{\max}+r)^2$. For any fixed u, applying Hoeffding's inequality yields

$$\Pr\left(-\frac{1}{mk}\sum_{l=1}^k X_{l,u} \ge t\right) = \Pr\left(\frac{1}{mk}\sum_{l=1}^k X_{l,u} \ge t\right) \le \exp\left(-\frac{m^2t^2k}{18(s_{\max}+r)^4}\right).$$

Further applying union bound, we have

$$\Pr\left(\max_u \frac{1}{mk}\sum_{l=1}^k X_{l,u} \ge t\right) \le \sum_u \Pr\left(\frac{1}{k}\sum_{l=1}^k X_{l,u} \ge mt\right) \le m\exp\left(-\frac{m^2t^2k}{18(s_{\max}+r)^4}\right),$$

which immediately implies that

$$\begin{aligned} \lambda_{\max}(\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})) &\le \frac{1}{4}\max_u \frac{1}{mk}\sum_{l=1}^k \mathbf{a}_{l,u}^\top \boldsymbol{s}\mathbf{a}_{l,u}^\top \boldsymbol{s} \\ &\le \frac{(\|\boldsymbol{s}^*\|_2+r)^2}{2n} + \frac{3(s_{\max}+r)^2}{4m}\sqrt{\frac{2\log(mn)}{k}} \\ &\le \frac{(\|\boldsymbol{s}^*\|_2+r)^2}{n} \end{aligned} \tag{2.4.27}$$

holds with probability at least $1-1/n$, where the last inequality is true when the sample size satisfies $k \ge 5(s_{\max}+r)^4 n^2/(m^2(\|\boldsymbol{s}^*\|_2+r)^4)\log(mn)$. On the other hand, we also have

$$\Pr\left(\max_u -\frac{1}{mk}\sum_{l=1}^k X_{l,u} \ge t\right) \le \sum_u \Pr\left(-\frac{1}{k}\sum_{l=1}^k X_{l,u} \ge mt\right) \le m\exp\left(-\frac{m^2t^2k}{18(s_{\max}+r)^4}\right),$$

which leads to the conclusion that

$$\begin{aligned}\lambda_{\min}(\nabla^2_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})) &\geq \frac{e^{5\gamma_{\max}s_{\max}}}{(1+e^{5\gamma_{\max}s_{\max}})^2}\max_u \frac{1}{mk}\sum_{l=1}^{k}\mathbf{a}_{l,u}^\top \boldsymbol{s}\mathbf{a}_{l,u}^\top \boldsymbol{s}\\ &\geq \frac{e^{5\gamma_{\max}s_{\max}}}{(1+e^{5\gamma_{\max}s_{\max}})^2}\left(\frac{2(\|\boldsymbol{s}^*\|_2+r)^2}{n}-\frac{3(s_{\max}+r)^2}{m}\sqrt{\frac{2\log(mn)}{k}}\right)\\ &\geq \frac{(\|\boldsymbol{s}^*\|_2+r)^2 e^{5\gamma_{\max}s_{\max}}}{n(1+e^{5\gamma_{\max}s_{\max}})^2} \end{aligned} \tag{2.4.28}$$

holds with probability at least $1-1/n$, where the last inequality is due to $k \geq 18(s_{\max}+r)^4 n^2/(m^2(\|\boldsymbol{s}^*\|_2+r)^4)\log(mn)$. □

Proof of Lemma 2.4.13

Proof. Recall the gradient of $\mathcal{L}$ with respect to $\boldsymbol{s}$ in (2.4.10). It holds that

$$\begin{aligned}\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}')\|_2 &= \left\|\frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\left(g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)\gamma_u - g'\left(\gamma_u'\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)\gamma_u'\right)\mathbf{a}_{l,u}\right\|_2\\ &\leq \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\left[\left|g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)(\gamma_u-\gamma_u')\right|\right.\\ &\quad\left.+\left|\left(g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)-g'\left(\gamma_u'\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)\right)\gamma_u'\right|\right]\|\mathbf{a}_{l,u}\|_2.\end{aligned}$$

Note that we have $|g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})|\leq 1$ and $\|\mathbf{a}_{l,u}\|_2=\sqrt{2}$. In addition, by the mean value theorem we have

$$g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)-g'\left(\gamma_u'\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)=g''(x)(\gamma_u-\gamma_u')\mathbf{a}_{l,u}^\top\boldsymbol{s},$$

where $x=t\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}+(1-t)\gamma_u'\mathbf{a}_{l,u}^\top\boldsymbol{s}$ for some $t\in(0,1)$. By plugging the range of γ_u and $\boldsymbol{s}$, we have $|x|\leq 5\gamma_{\max}s_{\max}$ by (2.4.23) and hence $|g''(x)|=|e^x/(1+e^x)^2|\leq 1/4$. Now we can bound $\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}')\|_2$ as follows:

$$\begin{aligned}\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}')\|_2 &\leq \frac{1}{mk}\sum_{u=1}^{m}\sum_{l=1}^{k}\sqrt{2}(1+2\gamma_{\max}s_{\max})|\gamma_u-\gamma_u'|\\ &\leq \frac{\sqrt{2}(1+2\gamma_{\max}s_{\max})}{\sqrt{m}}\|\boldsymbol{\gamma}-\boldsymbol{\gamma}'\|_2.\end{aligned}$$

Now we prove the upper bound of $\|\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma})\|_2$. First, we have by (2.4.10) that

$$\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma})=\frac{1}{mk}\begin{bmatrix}\sum_{l=1}^{k}\mathbf{a}_{l,1}^\top\left(g'\left(\gamma_1\mathbf{a}_{l,1}^\top\boldsymbol{s}\right)\boldsymbol{s}-g'\left(\gamma_1\mathbf{a}_{l,1}^\top\boldsymbol{s}'\right)\boldsymbol{s}'\right)\\ \vdots\\ \sum_{l=1}^{k}\mathbf{a}_{l,u}^\top\left(g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)\boldsymbol{s}-g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}'\right)\boldsymbol{s}'\right)\\ \vdots\end{bmatrix}.$$

Note that for each u, we have

$$\begin{aligned}&\mathbf{a}_{l,u}^\top\left(g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)\boldsymbol{s}-g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}'\right)\boldsymbol{s}'\right)\\ &=\mathbf{a}_{l,u}^\top\left[g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)(\boldsymbol{s}-\boldsymbol{s}')+\left(g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)-g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}'\right)\right)\boldsymbol{s}'\right].\end{aligned} \tag{2.4.29}$$

For the first term in (2.4.29), we have

$$\left|\mathbf{a}_{l,u}^\top g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)(\boldsymbol{s}-\boldsymbol{s}')\right|\leq\sqrt{2}\|\boldsymbol{s}-\boldsymbol{s}'\|_2.$$

For the second term in (2.4.29), applying the mean value theorem yields

$$\left|\mathbf{a}_{l,u}^\top\left(g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)-g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}'\right)\right)\boldsymbol{s}'\right|=\left|g''(x)\gamma_u\mathbf{a}_{l,u}^\top\left(\boldsymbol{s}-\boldsymbol{s}'\right)\mathbf{a}_{l,u}^\top\boldsymbol{s}'\right|\leq\frac{5\sqrt{2}\gamma_{\max}s_{\max}}{4}\|\boldsymbol{s}-\boldsymbol{s}'\|_2,$$

where $x=t\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}+(1-t)\gamma^u\mathbf{a}_{l,u}^\top\boldsymbol{s}'$ for some $t\in(0,1)$. Therefore, we have

$$\|\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s}',\boldsymbol{\gamma})\|_2\leq\frac{\sqrt{2}(1+2\gamma_{\max}s_{\max})}{\sqrt{m}}\|\boldsymbol{s}-\boldsymbol{s}'\|_2,$$

which completes our proof. □

Proof of Lemma 2.4.14

Proof. According to (2.4.10), the gradient of $\mathcal{L}$ with respect to $\boldsymbol{s}$ is

$$\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})=\frac{1}{mk}\sum_u\sum_l\frac{\left(-Y+(1-Y)\exp(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\right)\gamma_u\mathbf{a}_{l,u}}{1+\exp(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})}.$$

By assumption we have $|\gamma_u|\leq(\gamma_{\max}+r)$ and $|\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}|\leq5\gamma_{\max}s_{\max}$ by (2.4.23). In addition, we have $\|\gamma_u\mathbf{a}_{l,u}/(1+\exp(\gamma_u\mathbf{a}_{l,u}\boldsymbol{s}))\|_2\leq\sqrt{2}(\gamma_{\max}+r)/(1+e^{-5\gamma_{\max}s_{\max}})$. Applying Hoeffding's inequality, we have

$$\Pr\left(\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s},\boldsymbol{\gamma})\|_2\geq t\right)\leq2\exp\left(\frac{-(1+e^{-5\gamma_{\max}s_{\max}})^2mkt^2}{8(\gamma_{\max}+r)^2}\right),$$

which implies that

$$\|\nabla_{\boldsymbol{s}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{s}}\bar{\mathcal{L}}(\boldsymbol{s},\boldsymbol{\gamma})\|_2\leq\frac{2(\gamma_{\max}+r)}{1+e^{-5\gamma_{\max}s_{\max}}}\sqrt{\frac{2\log(2n)}{mk}}$$

holds with probability at least $1-1/n$. Recall the calculation in (2.4.10), the gradient of $\mathcal{L}$ with respect to $\boldsymbol{\gamma}$ is

$$\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})=\frac{1}{mk}\begin{bmatrix}\sum_{l=1}^k g'\left(\gamma_1\mathbf{a}_{l,1}^\top\boldsymbol{s}\right)\mathbf{a}_{l,1}^\top\boldsymbol{s}\\ \vdots\\ \sum_{l=1}^k g'\left(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s}\right)\mathbf{a}_{l,u}^\top\boldsymbol{s}\\ \vdots\end{bmatrix}.$$

The squared statistical error is

$$\|\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{\gamma}}\bar{\mathcal{L}}(\boldsymbol{s},\boldsymbol{\gamma})\|_2=\frac{1}{mk}\sqrt{\sum_u\left[\sum_l\left(g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}-\mathbb{E}[g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}]\right)^\top\boldsymbol{s}\right]^2},$$

which implies for all $t\geq0$

$$\begin{aligned}&\Pr\left(\|\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma})-\nabla_{\boldsymbol{\gamma}}\bar{\mathcal{L}}(\boldsymbol{s},\boldsymbol{\gamma})\|_2\geq t\right)\\&\leq\Pr\left(\max_u\frac{1}{k}\sum_l\left(g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}-\mathbb{E}[g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}]\right)^\top\boldsymbol{s}\geq\sqrt{m}t\right)\\&\leq\sum_u\Pr\left(\frac{1}{k}\sum_l\left(g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}-\mathbb{E}[g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}]\right)^\top\boldsymbol{s}\geq\sqrt{m}t\right),\end{aligned}$$

where the last inequality is due to union bound. For each user u, we have

$$|(g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}-\mathbb{E}[g'(\gamma_u\mathbf{a}_{l,u}^\top\boldsymbol{s})\mathbf{a}_{l,u}])^\top\boldsymbol{s}|\leq\frac{10\gamma_{\max}s_{\max}}{1+e^{-5\gamma_{\max}s_{\max}}}.$$

Applying Hoeffding's inequality yields

$$\Pr\left(\|\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) - \nabla_{\boldsymbol{\gamma}}\bar{\mathcal{L}}(\boldsymbol{s},\boldsymbol{\gamma})\|_2 \geq t\right) \leq 2m\exp\left(\frac{-(1+e^{-5\gamma_{\max}s_{\max}})^2 t^2 mk}{100\gamma_{\max}^2 s_{\max}^2}\right),$$

which immediately leads to the conclusion that

$$\|\nabla_{\boldsymbol{\gamma}}\mathcal{L}(\boldsymbol{s},\boldsymbol{\gamma}) - \nabla_{\boldsymbol{\gamma}}\bar{\mathcal{L}}(\boldsymbol{s},\boldsymbol{\gamma})\|_2 \leq \frac{10\gamma_{\max}s_{\max}}{1+e^{-5\gamma_{\max}s_{\max}}}\sqrt{\frac{2\log(2mn)}{mk}}$$

holds with probability at least $1 - 1/n$. This completes the proof. □

Proof of Proposition 2.3.1

Proof. Since the PDF g of the noise terms ϵ_i is log-concave, and because the convolution of log-concave functions is log-concave Merkle (1998), the CDF F of $\epsilon_j - \epsilon_i$ for any pair i, j is also log-concave. Hence $h(x) = -\log F(x)$ is convex. The loss function is the sum of terms of the form $h_{iju} = h(\gamma_u(s_i - s_j))$. Fix i, j, and u. We have

$$\nabla_{\boldsymbol{s}}^2 h_{iju} = h''(\gamma_u(s_i - s_j))(\gamma_u)^2(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^\top,$$

where $\mathbf{e}_i$ is the standard unit vector for coordinate i in $\mathbb{R}^n$. By the convexity of h and the fact that $(\mathbf{e}_i - \mathbf{e}_j)(\mathbf{e}_i - \mathbf{e}_j)^\top$ is positive-definite, the loss function is convex in $\boldsymbol{s}$. Similarly, it is easy to show that it is convex in $\boldsymbol{\gamma}$. □

2.5 Experiments

In this section, we present experimental results to show the performance of the proposed algorithm on heterogeneous populations of users. The experiments are conducted on both synthetic and real data with both benign users and adversarial users. We use the Kendall's tau correlation Kendall (1948) between the estimated and true rankings to measure the similarity between rankings, which is defined as $\tau = \frac{2(c-d)}{n(n-1)}$, where c and d are the number of pairs on which the two rankings agree and disagree, respectively. Pairs that are tied in at least one of the rankings are not counted in c or d.

Baseline methods: In Gumbel noise setting, we compare Algorithm 2.1 based on our proposed HBTL model with (i) the BTL model that can be optimized through iterative maximum-likelihood methods (Negahban et al., 2012) or spectral methods such as Rank Centrality (Negahban et al., 2016); and (ii) the CrowdBT algorithm (Chen et al., 2013), which is a variation of BTL that allows users with different levels of accuracy. In the normal noise setting, we compare Algorithm 2.1 based on our proposed HTCV model with TCV model. We also implemented a TCV equivalent of CrowdBT and report its performance as CrowdTCV.

2.5.1 Experimental Results on Synthetic Data

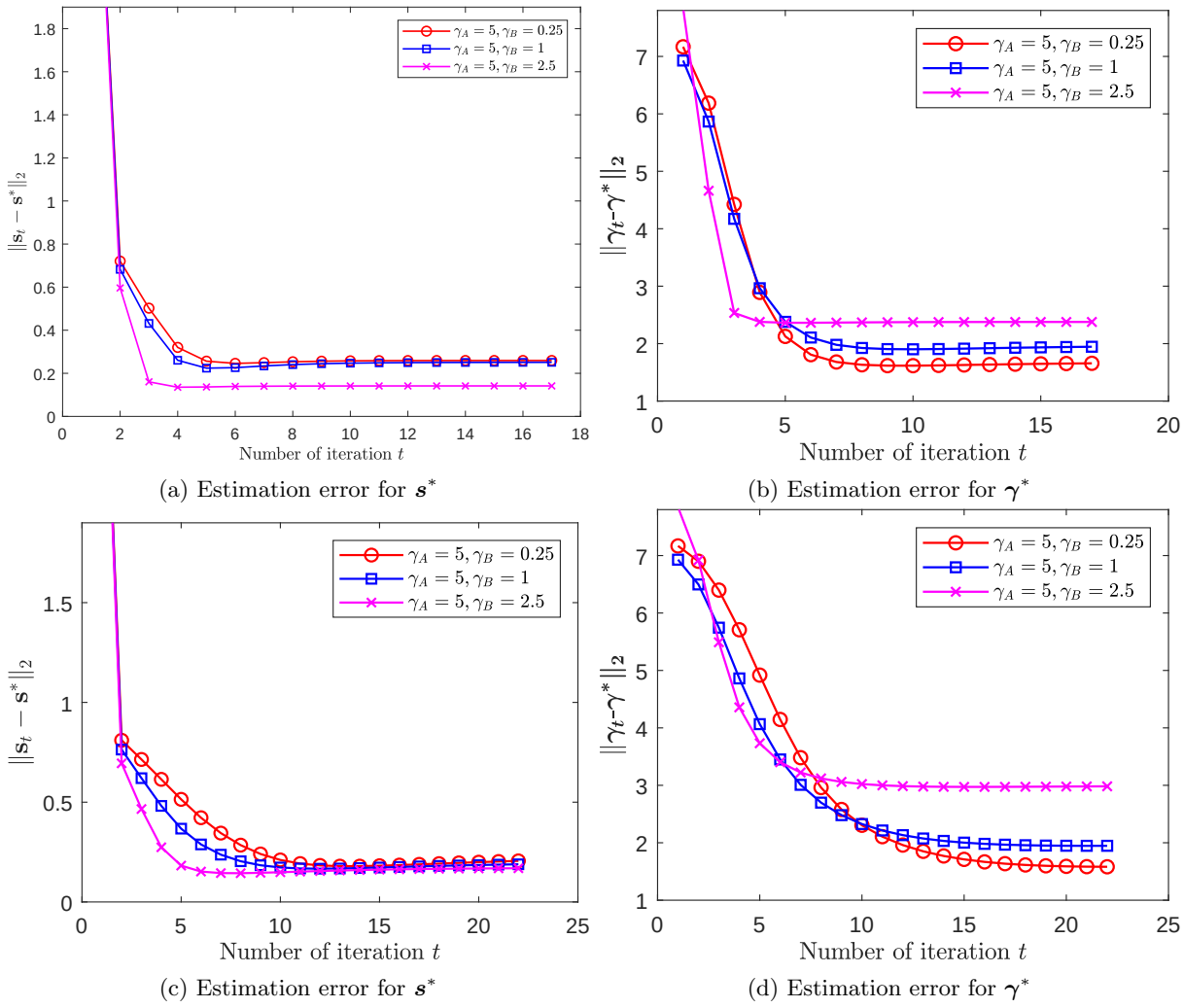
We set number of items $n = 20$, number of users $m = 9$ and set the ground truth score vector $\boldsymbol{s}$ to be uniformly distributed in $[0, 1]$. The m users are divided into groups A and B, consisting of 3 and 6 users respectively. These two groups of users generate heterogeneous data in the sense that users in group A are more accurate than those in group B. We vary γ_A in the range of $\{2.5, 5, 10\}$ and γ_B in the range of $\{0.25, 1, 2.5\}$, which leads to in total 9 configurations of data generation. For each configuration, we conduct the experiment under the following two settings:

(1) **Benign:** $\gamma_1, \ldots, \gamma_3 = \gamma_A$ (Group A); $\gamma_4, \ldots, \gamma_9 = \gamma_B$ (Group B).

(2) **Adversarial:** $\gamma_1 = -\gamma_A$, $\gamma_2, \gamma_3 = \gamma_A$ (Group A); $\gamma_4, \gamma_5 = -\gamma_B$, $\gamma_6, \ldots, \gamma_9 = \gamma_B$ (Group B).

We also test on various densities of compared pairs, which effectively controls the sample size. In particular, we choose 4 sets of α, which denote the portion of all possible pairs that are compared. The larger the value, the more pairs are compared by each user. The simulation process is as follows: we first generate $n(n-1)$ ordered pairs of items, where n is the number of items. This is equivalent to comparing each unique pair of items twice. Then for each pair of items, response from every annotator had a probability of α to be recorded and used for training the model. And α is chosen from $\{0.2, 0.4, 0.6, 0.8\}$ to make up for four runs. Each experiment is repeated 100 times with different random seeds.

Under setting (1), we plot the estimation error of Algorithm 2.1 v.s. number of iterations for HBTL and HTCV model in Figures 2.2a-2.2b and 2.2c-2.2d respectively. In all settings, our algorithm enjoys a linear convergence rate to the true parameters up to statistical errors, which is well aligned with the theoretical results in Theorem 2.4.5.



(a) Estimation error for $\boldsymbol{s}^*$

(b) Estimation error for $\boldsymbol{\gamma}^*$

(c) Estimation error for $\boldsymbol{s}^*$

(d) Estimation error for $\boldsymbol{\gamma}^*$

Figure 2.2: Evolution of estimation errors vs. number of iterations t for HBTL model (a-b) and HTCV model (c-d). The plots show the convergence behavior of both score vector $\boldsymbol{s}^*$ and accuracy parameters $\boldsymbol{\gamma}^*$ estimation.

When there is no adversarial users in the system, the ranking results for Gumbel noises under different configurations of γ_A and γ_B are shown in Table 2.1 and the ranking results for normal noises under different configurations of γ_A and γ_B are shown in Table 2.2. In both tables, each cell presents the Kendall's tau correlation between the aggregated ranking and the ground truth, averaged over 100 trials. For each experimental setting, we use the bold text to denote the method which achieved highest performance. We also underline the highest score whenever there is a tie. It can be observed that in almost all cases, HBTL provides much more accurate rankings than BTL and HTCV significantly outperforms TCV as well. In particular, the larger the difference between γ_A and γ_B is, the more significant the improvement is. The only exception is when $\gamma_A = \gamma_B = 2.5$, in which case the data is not heterogeneous and our HRUM model has no advantage. Nevertheless, our method still achieve comparable performance as BTL for non-heterogeneous data. It can also be observed that HBTL generally outperforms CrowdBT. But the advantage is not large, as CrowdBT also includes the different accuracy levels of different users. Importantly, however, as discussed

in Section 2.3.1, CrowdBT is not compatible with the additive noise in Thurstonian models and cannot be extended in a natural way to ranked data other than pairwise comparison. In addition, unlike CrowdBT, our method enjoys strong theoretical guarantees while maintaining a good performance. Tables 2.1 and 2.2 also illustrate an important fact: If there are users with high accuracy, the presence of low quality data does not significantly impact the performance of Algorithm 2.1.

When there are a portion of adversarial users as stated in setting (2), we consider adversarial users whose accuracy level γ_u may take negative values as discussed above. The results for Gumbel and normal noises under setting (2) are shown in Table 2.3 and Table 2.4 respectively. It can be seen that in this case, the difference between the methods is even more pronounced.

2.5.2 Experimental Results on Real-World Data

We evaluate our method on two real-world datasets. The first one named "Reading Level" (Chen et al., 2013) contains English text excerpts whose reading difficulty level is compared by workers. 624 workers annotated 490 excerpts which resulting in a total of $12,728$ pairwise comparisons. We also used Mechanical Turk to collect another dataset named "Country Population". In this crowdsourcing task, we asked workers to compare the population between two countries and pick the one which has more population. Since the population ranking of countries has a universal consensus, which can be obtained by looking up demographic data, it is a better choice than those movie rankings which subjects to personal preferences. There were 15 countries as shown in Table 2.5 which made up to 105 pairwise comparisons. The values were collected according to the latest demography statistics on Wikipedia for each country as of March 2019. Each user was asked 16 pairs randomly selected from all those 105 pairs. A total of 199 workers provided response to this task through Mechanical Turk. These two datasets were both collected in online crowdsourcing environments so that we can expect varying worker accuracy where effectiveness of our approach can be demonstrated.

In real-world datasets, it may happen that two items from two subsets are never compared with each other, directly or indirectly. In such cases, the ranking will not be unique. Furthermore, if data is sparse, the estimates may suffer from overfitting. To address these issues, regularization is often used. While this can be done in a variety of ways, for the sake of comparison with CrowdBT, we use virtual node regularization (Chen et al., 2013). Specifically, it is assumed that there is a virtual item of utility $s_0 = 0$ which is compared to all other items by a virtual user. This leads to the loss function $\mathcal{L} + \lambda_0 \mathcal{L}_0$, where $\mathcal{L}_0 = -\sum_{i\in[n]} \log F(s_0 - s_i) - \sum_{i\in[n]} \log F(s_i - s_0)$ and $\lambda_0 \geq 0$ is a tuning parameter.

We evaluate the performance of the methods for $\lambda_0 = 0, 1, 5, 10$. For different values of λ_0, HBTL performs best more often than any other method and, in particular, it performs best for $\lambda_0 = 0$. Table 2.6 reports the best performance of each method across different regularization values for the two real-world data experiment. It can be observed that HBTL and HTCV outperform their counterparts, CrowdBT and CrowdTCV, as well as the uniform models, BTL and TCV.

2.5.3 Analysis on regularization effects

Detailed result with various regularization settings can be found in Table 2.7 and Table 2.8. The reported values are Kendall's tau correlation. It shows that without regularization our method outperforms other methods. And with virtual node trick, it shows relative amount of improvement in the final ranking result, yet not essential. However, this method needs to tune another parameter λ_0. If no gold/ground-truth comparison is given, there will be no validation standard to tune this parameter. Furthermore, the performance of the proposed methods is less dependent on the regularization parameter, which facilitates their application to real data. It is also interesting to see that our method is less prone to be affected by the regularization parameter.

Table 2.1: Kendall's tau correlation for different method under Gumbel noise. Group A users all have the accuracy level γ_A and Group B users all have the accuracy level γ_B. α represents the portion of all possible pairwise comparisons each annotator labeled in the simulation. The **bold** number highlights the highest performance and the underlined number indicates a tie.

Observ. Ratio	γ_B	Methods	γ_A		
			2.5	5	10
$\alpha = 0.8$	0.25	BTL	0.767±0.055	0.836±0.043	0.879±0.032
		CrowdBT	0.847±0.042	0.928±0.023	0.962±0.016
		HBTL	**0.850**±0.041	**0.930**±0.024	**0.964**±0.015
	1.0	BTL	0.863±0.036	0.896±0.028	0.923±0.026
		CrowdBT	0.875±0.033	0.930±0.024	0.967±0.018
		HBTL	0.875±0.033	0.930±0.024	**0.969**±0.017
	2.5	BTL	**0.933**±0.022	0.946±0.019	0.959±0.018
		CrowdBT	0.931±0.024	0.947±0.019	0.967±0.017
		HBTL	0.931±0.025	**0.948**±0.021	**0.972**±0.015
$\alpha = 0.6$	0.25	BTL	0.743±0.064	0.814±0.048	0.853±0.037
		CrowdBT	0.823±0.050	**0.909**±0.034	0.954±0.018
		HBTL	**0.824**±0.051	0.908±0.033	**0.955**±0.018
	1.0	BTL	0.837±0.036	0.872±0.033	0.903±0.033
		CrowdBT	**0.853**±0.035	0.911±0.031	0.955±0.018
		HBTL	0.851±0.033	**0.913**±0.028	**0.958**±0.017
	2.5	BTL	**0.913**±0.032	0.931±0.024	0.948±0.021
		CrowdBT	0.910±0.028	0.935±0.020	0.961±0.016
		HBTL	0.912±0.029	**0.936**±0.022	**0.967**±0.017
$\alpha = 0.4$	0.25	BTL	0.671±0.062	0.761±0.053	0.812±0.048
		CrowdBT	0.764±0.065	0.872±0.037	0.933±0.024
		HBTL	**0.769**±0.061	**0.873**±0.034	**0.934**±0.022
	1.0	BTL	0.791±0.051	0.844±0.043	0.866±0.035
		CrowdBT	0.798±0.050	0.889±0.029	0.934±0.027
		HBTL	**0.806**±0.051	**0.891**±0.031	**0.936**±0.026
	2.5	BTL	**0.882**±0.034	0.910±0.030	0.919±0.027
		CrowdBT	0.879±0.034	0.912±0.026	0.943±0.022
		HBTL	0.880±0.032	**0.916**±0.028	**0.945**±0.020
$\alpha = 0.2$	0.25	BTL	0.575±0.095	0.663±0.078	0.712±0.069
		CrowdBT	0.644±0.094	0.798±0.055	**0.884**±0.035
		HBTL	**0.665**±0.090	**0.805**±0.051	0.882±0.034
	1.0	BTL	**0.708**±0.073	0.768±0.057	0.804±0.039
		CrowdBT	0.696±0.081	0.813±0.052	0.876±0.034
		HBTL	0.702±0.079	**0.819**±0.052	**0.882**±0.034
	2.5	BTL	**0.820**±0.044	0.861±0.043	0.883±0.033
		CrowdBT	0.803±0.048	0.857±0.037	0.898±0.030
		HBTL	0.807±0.049	0.861±0.038	**0.904**±0.029

Table 2.2: Kendall's tau correlation for different methods under noise from the normal distribution. Group A users all have the accuracy level γ_A and Group B users all have the accuracy level γ_B. α represents the portion of all possible pairwise comparisons each annotator labeled in the simulation. The **bold** number highlights the highest performance and the underlined number indicates a tie.

Observ. Ratio	γ_B	Methods	γ_A		
			2.5	5	10
$\alpha = 0.8$	0.25	TCV	0.811±0.048	0.860±0.040	0.885±0.036
		CrowdTCV	0.881±0.032	0.943±0.021	0.971±0.014
		HTCV	**0.882**±0.030	0.943±0.021	0.971±0.015
	1.0	TCV	0.885±0.036	0.910±0.027	0.925±0.029
		CrowdTCV	0.897±0.030	0.944±0.020	0.973±0.015
		HTCV	0.897±0.033	0.944±0.020	**0.975**±0.013
	2.5	TCV	0.945±0.021	0.956±0.018	0.965±0.018
		CrowdTCV	0.945±0.021	0.954±0.019	0.976±0.014
		HTCV	0.944±0.021	**0.959**±0.017	**0.981**±0.014
$\alpha = 0.6$	0.25	TCV	0.763±0.059	0.830±0.043	0.850±0.041
		CrowdTCV	0.845±0.038	**0.926**±0.023	0.961±0.020
		HTCV	**0.846**±0.040	0.925±0.025	0.961±0.020
	1.0	TCV	0.862±0.038	0.892±0.034	0.912±0.025
		CrowdTCV	0.870±0.035	0.930±0.028	0.962±0.019
		HTCV	**0.875**±0.033	**0.932**±0.027	**0.963**±0.018
	2.5	TCV	**0.927**±0.027	0.943±0.021	0.955±0.019
		CrowdTCV	0.925±0.027	0.946±0.026	0.968±0.015
		HTCV	0.925±0.027	**0.952**±0.022	**0.974**±0.013
$\alpha = 0.4$	0.25	TCV	0.691±0.073	0.790±0.047	0.809±0.048
		CrowdTCV	0.804±0.050	0.901±0.028	**0.946**±0.022
		HTCV	**0.808**±0.049	**0.904**±0.028	0.945±0.022
	1.0	TCV	0.821±0.047	0.859±0.036	0.875±0.036
		CrowdTCV	0.832±0.044	0.900±0.035	0.946±0.020
		HTCV	**0.836**±0.043	**0.904**±0.032	**0.947**±0.020
	2.5	TCV	**0.901**±0.027	0.921±0.029	0.935±0.026
		CrowdTCV	0.895±0.031	0.923±0.028	0.950±0.019
		HTCV	0.895±0.030	**0.926**±0.025	**0.957**±0.018
$\alpha = 0.2$	0.25	TCV	0.599±0.088	0.688±0.077	0.738±0.060
		CrowdTCV	0.689±0.080	0.826±0.046	**0.899**±0.031
		HTCV	**0.693**±0.082	**0.828**±0.049	0.898±0.034
	1.0	TCV	0.733±0.070	0.791±0.055	0.815±0.041
		CrowdTCV	0.729±0.074	0.836±0.043	**0.904**±0.033
		HTCV	**0.740**±0.072	**0.841**±0.038	0.901±0.031
	2.5	TCV	**0.856**±0.041	0.878±0.036	0.888±0.032
		CrowdTCV	0.844±0.048	0.873±0.035	0.905±0.027
		HTCV	0.848±0.041	**0.881**±0.036	**0.913**±0.026

Table 2.3: Kendall's tau correlation for different methods under noise from the Gumbel distribution when a third of the users are *adversarial*. The **bold** number highlights the highest performance and the underlined number indicates a tie.

Observ. Ratio	γ_B	Methods	γ_A		
			2.5	5	10
$\alpha = 0.8$	0.25	BTL	0.443±0.107	0.569±0.096	0.614±0.085
		CrowdBT	0.852±0.044	0.925±0.023	**0.967**±0.017
		HBTL	0.852±0.045	**0.926**±0.023	0.966±0.017
	1.0	BTL	0.575±0.089	0.663±0.071	0.710±0.074
		CrowdBT	0.873±0.037	0.931±0.023	**0.967**±0.014
		HBTL	**0.875**±0.037	**0.932**±0.024	0.966±0.017
	2.5	BTL	0.725±0.057	0.780±0.046	0.798±0.047
		CrowdBT	0.931±0.025	0.948±0.019	0.966±0.016
		HBTL	0.931±0.025	**0.951**±0.019	**0.973**±0.015
$\alpha = 0.6$	0.25	BTL	0.384±0.122	0.491±0.107	0.557±0.095
		CrowdBT	0.822±0.046	0.908±0.030	0.953±0.019
		HBTL	**0.824**±0.044	**0.910**±0.028	**0.954**±0.018
	1.0	BTL	0.546±0.097	0.627±0.078	0.670±0.080
		CrowdBT	0.852±0.037	0.911±0.029	0.954±0.018
		HBTL	**0.854**±0.037	**0.914**±0.028	**0.956**±0.019
	2.5	BTL	0.684±0.078	0.736±0.064	0.755±0.062
		CrowdBT	0.910±0.028	0.934±0.025	0.960±0.016
		HBTL	**0.912**±0.029	**0.936**±0.024	**0.965**±0.017
$\alpha = 0.4$	0.25	BTL	0.323±0.130	0.405±0.132	0.485±0.109
		CrowdBT	0.742±0.169	0.877±0.033	**0.934**±0.025
		HBTL	**0.766**±0.059	0.877±0.035	0.933±0.024
	1.0	BTL	0.448±0.118	0.544±0.096	0.583±0.094
		CrowdBT	0.810±0.044	0.886±0.031	0.934±0.026
		HBTL	**0.819**±0.045	**0.891**±0.031	0.934±0.029
	2.5	BTL	0.627±0.087	0.660±0.075	0.698±0.063
		CrowdBT	0.879±0.034	0.913±0.027	0.939±0.023
		HBTL	**0.880**±0.032	**0.914**±0.029	**0.948**±0.022
$\alpha = 0.2$	0.25	BTL	0.246±0.145	0.305±0.151	0.361±0.143
		CrowdBT	0.613±0.235	**0.712**±0.356	0.848±0.256
		HBTL	**0.614**±0.263	0.709±0.380	0.848±0.249
	1.0	BTL	0.336±0.154	0.407±0.127	0.452±0.132
		CrowdBT	0.644±0.282	0.795±0.176	0.878±0.038
		HBTL	**0.650**±0.281	**0.807**±0.172	**0.888**±0.040
	2.5	BTL	0.498±0.106	0.548±0.103	0.571±0.098
		CrowdBT	0.803±0.049	0.858±0.039	0.897±0.032
		HBTL	**0.807**±0.049	**0.865**±0.039	**0.900**±0.029

Table 2.4: Kendall tau correlation for different methods under noise from the normal distribution when a third of the users are *adversarial*. The **bold** number highlights the highest performance and the underlined number indicates a tie.

Observ. Ratio	γ_B	Methods	γ_A		
			2.5	5	10
$\alpha = 0.8$	0.25	TCV	0.471±0.105	0.590±0.095	0.640±0.075
		CrowdTCV	0.882±0.034	**0.938**±0.023	0.972±0.017
		HTCV	0.882±0.033	0.937±0.023	**0.973**±0.016
	1.0	TCV	0.642±0.083	0.694±0.068	0.722±0.064
		CrowdTCV	0.893±0.030	0.945±0.020	0.973±0.016
		HTCV	**0.895**±0.031	**0.947**±0.019	**0.975**±0.017
	2.5	TCV	0.772±0.055	0.804±0.045	0.821±0.050
		CrowdTCV	**0.945**±0.021	0.956±0.019	0.978±0.014
		HTCV	0.944±0.021	**0.960**±0.019	**0.982**±0.013
$\alpha = 0.6$	0.25	TCV	0.416±0.129	0.527±0.107	0.552±0.099
		CrowdTCV	0.847±0.039	0.924±0.025	0.960±0.020
		HTCV	0.847±0.039	**0.925**±0.023	0.960±0.020
	1.0	TCV	0.569±0.086	0.648±0.066	0.686±0.080
		CrowdTCV	0.866±0.036	0.930±0.024	0.966±0.018
		HTCV	**0.870**±0.036	**0.932**±0.025	0.966±0.018
	2.5	TCV	0.718±0.060	0.762±0.045	0.786±0.055
		CrowdTCV	**0.926**±0.027	0.949±0.023	0.969±0.014
		HTCV	0.925±0.027	**0.952**±0.020	**0.972**±0.014
$\alpha = 0.4$	0.25	TCV	0.359±0.119	0.472±0.116	0.514±0.103
		CrowdTCV	0.797±0.053	0.893±0.034	**0.942**±0.022
		HTCV	**0.799**±0.048	**0.896**±0.031	0.938±0.022
	1.0	TCV	0.487±0.116	0.577±0.088	0.587±0.088
		CrowdTCV	0.842±0.049	0.898±0.029	**0.945**±0.021
		HTCV	**0.843**±0.046	**0.902**±0.027	0.944±0.022
	2.5	TCV	0.648±0.073	0.704±0.071	0.718±0.066
		CrowdTCV	0.895±0.031	0.925±0.031	0.951±0.021
		HTCV	0.895±0.030	**0.929**±0.028	**0.957**±0.018
$\alpha = 0.2$	0.25	TCV	0.259±0.147	0.349±0.135	0.382±0.133
		CrowdTCV	0.600±0.340	0.826±0.044	**0.895**±0.038
		HTCV	**0.636**±0.282	**0.828**±0.044	0.893±0.036
	1.0	TCV	0.397±0.119	0.436±0.115	0.469±0.100
		CrowdTCV	0.721±0.065	**0.834**±0.043	0.901±0.033
		HTCV	**0.736**±0.066	0.832±0.046	**0.905**±0.032
	2.5	TCV	0.518±0.102	0.577±0.098	0.600±0.077
		CrowdTCV	0.843±0.049	0.873±0.037	0.908±0.030
		HTCV	**0.848**±0.041	**0.880**±0.036	**0.917**±0.028

Table 2.5: Ground truth for "Country Population" dataset.

Country	Population (million)
China	1410
India	1340
United States	324
Indonesia	264
Brazil	209
Pakistan	197
Nigeria	191
Bangladesh	165
Russia	144
Mexico	129
Japan	127
Ethiopia	105
Philippines	104.9
Egypt	97.6
Vietnam	95.5

Table 2.6: Performance of ranking algorithms on real-world dataset. The **bold** number highlights the highest performance.

Dataset	BTL	TCV	CrowdBT	CrowdTCV	HBTL	HTCV
Reading Level	0.3472	0.3452	0.3737	0.3672	**0.3763**	0.3729
Country Population	0.7524	0.7524	0.7714	0.7714	**0.7905**	0.7714

Table 2.7: Performance of ranking algorithms for the "Reading Level" dataset with different regularization parameters. The **bold** number highlights the highest performance.

	$\lambda_0 = 0$	$\lambda_0 = 1$	$\lambda_0 = 5$	$\lambda_0 = 10$
BTL	0.3299	0.3433	0.3472	0.3402
TCV	0.3294	0.3423	0.3452	0.3375
CrowdBT	0.3490	**0.3737**	0.3648	0.3535
CrowdTCV	0.3512	0.3672	0.3511	0.3388
HBTL	**0.3608**	0.3660	0.3719	**0.3763**
HTCV	0.3578	0.3696	**0.3729**	0.3680

Table 2.8: Performance of ranking algorithms for the "Country Population" dataset with different regularization parameters. The **bold** number highlights the highest performance.

	$\lambda_0 = 0$	$\lambda_0 = 1$	$\lambda_0 = 5$	$\lambda_0 = 10$
BTL	0.7524	0.7524	0.7524	0.7524
TCV	0.7524	0.7524	0.7524	0.7524
CrowdBT	0.7714	0.7714	**0.7714**	0.7524
CrowdTCV	0.7714	0.7714	**0.7714**	0.7524
HBTL	**0.7905**	**0.7905**	0.7524	0.7524
HTCV	0.7714	0.7714	0.7524	0.7524

Chapter 3

Heterogeneous Active Ranking under Strong Stochastic Assumption

3.1 Introduction

Nowadays, it is common to collect large-scale datasets in order to facilitate the process of knowledge discovery. Due to its scale, such data collection is usually carried out by crowdsourcing (Kumar and Lease, 2011; Chen et al., 2013), where different entities with diverse backgrounds generate subsets of the data. While crowdsourcing makes it possible to scale up the size, it also brings new challenges when it comes to the cost of operation and cleanness of the data. For example, the optimal ranking algorithm in the single-user setting (Ren et al., 2019) may not be straightforwardly extended to the heterogeneous setting while maintaining optimality. In particular, if we know the most accurate user among the set of users providing comparisons, the best we can do is to apply optimal single-user[1] ranking algorithms such as Iterative-Insertion-Ranking (`IIR`) (Ren et al., 2019) by querying only the most accurate user. Unfortunately, in practice, the accuracies of the users are often unknown. A naive solution may be to randomly select a user to query and use the comparisons provided by this user to insert an item into the ranked list per `IIR`. However, as we show later, this naive method usually bears a high sample complexity. Therefore, it is of great interest to design methods that can adaptively select a subset of users at each time to query pairwise comparisons in order to insert an item correctly into the ranked list.

In this chapter, we study the rank aggregation problem, where a heterogeneous set of users provide noisy pairwise comparisons for the items. We propose a novel algorithm that queries comparisons for pairs of items from a changing active user set. Specifically, we maintain a short history of user responses for a set of comparisons. When the inferred rank of these comparisons is estimated to be true with a high confidence, it is then used to calculate a reward based on the recorded responses. Then an upper confidence bound (UCB)-style elimination process is performed to remove inaccurate users from active user set.

3.2 Related Work

For passive ranking problems, a static dataset is given beforehand. Inference of the ranking often relies on models of ranked data, such as the Bradley-Terry-Luce (BTL) model (Bradley and Terry, 1952) and the Thurstone model (Thurstone, 1927). In contrast to passive algorithms, active algorithms leverage assumptions embedded in the models to identify the most informative pairs to query, thus reducing the sample complexity of queries. For instance, in Maystre and Grossglauser (2017), under the assumption that the true scores for N items are generated by a Poisson process, with $O(N\text{poly}(\log(N))$ comparisons, an *approximate* ranking of N items can be found. Let the probability of making a correct comparison between item i and the most similar item to item i be $\frac{1}{2}+\Delta_i$ and let $\Delta_{\min} = \min_{i\in[N]} \Delta_i$. An instance-dependent sample complexity

[1] In this chapter, we use the term single-user to refer to the case where only one information source is queried at each time. And the same for multi-user as multiple information sources can be queried at each time.

bound of $O(N\log(N)\Delta_{\min}^{-2}\log(N/(\delta\Delta_{\min}))$ is provided along with a QuickSort based algorithm by Szörényi et al. (2015). In Ren et al. (2019), an analysis for a distribution agnostic active ranking scheme is provided. To achieve a δ-correct *exact* ranking, $O(\sum_{i\in[N]}\Delta_i^{-2}(\log\log(\Delta_i^{-1})+\log(N/\delta)))$ comparisons are required. The exact inference requirement results in repeated queries of the same pair, which costs a constant overhead compared to approximate inference.

3.3 Preliminaries and Problem Setup

3.3.1 Ranking from Noisy Pairwise Comparisons

Suppose there are N items that we want to rank and M users to be queried. An item is indexed by an integer $i\in[N]$. We assume there is a unique *true ranking* of the N items. A user is also indexed by an integer $u\in[M]$. For a subset of users, we use $\mathcal{U}\subseteq[M]$ to denote the index subset. In each time step, we can pick a pair of items i and j and ask a user u whether item i is better than item j. The comparison returned by the user may be noisy. We assume that for any pair of items (i,j) with true ranking $i\succ j$, the probability that user u answers the query correctly is $p_u(i,j)=\Delta_u+1/2$, where $\Delta_u\in(0,\frac{1}{2}]$ is referred to as the accuracy level of user u. When some of the Δ_u's are different from the others, we call the set of users *heterogeneous*. We assume comparison results for item pairs, regardless the queried user, are mutually independent. While this independence assumption may not always hold for real datasets, it is commonly adopted in the literature as it facilitates the analysis (Falahatgar et al., 2017b, 2018; Jin et al., 2020).

In this chapter, we aim to achieve the exact ranking for a ranking problem defined as follows.

Definition 3.3.1 (Exact Ranking with Multiple Users)**.** Given N items, M users, and $\delta\in(0,1)$, our goal is to identify the true ranking among the N items with probability at least $1-\delta$. An algorithm $\mathcal{A}$ is δ-correct if, for any instance of the input, it will return the correct result in finite time with probability at least $1-\delta$.

To actively eliminate the users in the user pool, we define an α-optimal user as follows.

Definition 3.3.2. Let $\mathcal{U}\subseteq[M]$ be an arbitrary subset of users. If a user $x\in\mathcal{U}$ satisfies $\Delta_x+\alpha\geq\max_{u\in\mathcal{U}}\Delta_u$, then x is called an *α-optimal user in $\mathcal{U}$*. If a user is α-optimal among all M users, then it is called an (global) α-optimal user.

3.3.2 Iterative Insertion Ranking with a Single User

When there is only one user u to be queried ($M=1$), the problem defined in Section 3.3.1 reduces to the exact ranking problem with a single user, for which Ren et al. (2019) proposed the Iterative-Insertion-Ranking (`IIR`) algorithm. The sample complexity (i.e., the total number of queries) to achieve exact ranking with probability $1-\delta$ is characterized by the following proposition:

Proposition 3.3.3 (Adapted from Theorems 2 and 12 in Ren et al. (2019))**.** Given $\delta\in(0,1/12)$ and an instance of N items, the number of comparisons used by any δ-correct algorithm $\mathcal{A}$ on this instance is at least

$$\Theta\big(N\Delta_u^{-2}\big(\log\log\Delta_u^{-1}+\log(N/\delta)\big)\big). \tag{3.3.1}$$

Moreover, the `IIR` algorithm proposed by Ren et al. (2019) can output the exact ranking using this number of comparisons, with probability $1-\delta$.

The complexity above can be decomposed into the complexity of inserting each item into a constructed sorting tree.

In this chapter, we consider a more challenging ranking problem, where multiple users with heterogeneous levels of accuracies can be queried each time. In the multi-user setting, the optimal sample complexity in (3.3.1) can be achieved only if we know which user is the best user, i.e., $u^*=\arg\max_{u\in[M]}\Delta_u$. The optimal sample complexity can then be written as

$$\mathcal{C}_{u^*}(N)=\Theta\big(N\Delta_{u^*}^{-2}\big(\log\log\Delta_{u^*}^{-1}+\log(N/\delta)\big)\big). \tag{3.3.2}$$

However, with no prior information on the users' comparison accuracies, it is unclear whether we can achieve a sample complexity close to (3.3.2). In this scenario, the most primitive route is to perform no inference on the users' accuracy and randomly choose users to query. This leads to an equivalent accuracy of $\bar{\Delta}_0 = \frac{1}{M}\sum_{u\in[M]}\Delta_u$ and a sample complexity given as

$$\mathcal{C}_{\text{ave}}(N) = \Theta\big(N\bar{\Delta}_0^{-2}\big(\log\log\bar{\Delta}_0^{-1} + \log(N/\delta)\big)\big). \tag{3.3.3}$$

Compared with the best possible complexity (3.3.2), the sample complexity (3.3.3) is larger by a factor (ignoring logarithmic factors) up to M^2, because the ratio between Δ_{u^*} and $\bar{\Delta}_0$ could vary a lot for different set of users and can be as large as M. This is certainly undesirable, especially when there are a large number of items to be ranked. Therefore, an immediate question is: Can we design an algorithm that has a smaller multiplicative factor in its sample complexity compared with the optimal sample complexity? What we will propose in the following section is an algorithm that can achieve a sublinear regret, where the regret is defined as the difference between the sample complexity of the proposed algorithm and the optimal sample complexity.

3.4 Adaptive Sampling and User Elimination

The main framework of our procedure is derived based on the `Iterative-Insertion-Ranking` algorithm proposed in Ren et al. (2019), which, to the best of our knowledge, is the first algorithm that has matching instance-dependent upper and lower sample complexity bounds for active ranking problems in the single-user setting. We assume that the strong stochastic transitivity (SST) assumption defined in Falahatgar et al. (2017b, 2018) holds in our setting. The ranking algorithm comprises the following four hierarchical parts and operates on a Preference Interval Tree (PIT) (Feige et al., 1994; Ren et al., 2019), which stores the currently inserted and sorted items.

1. `Adaptive Iterative-Insertion-Ranking` (`Ada-IIR`): the main procedure which calls `IAI` to insert an item into a PIT with a high probability of correctness. It is displayed in Algorithm 3.2.

2. `Iterative-Attempting-Insertion` (`IAI`): the subroutine which calls `ATI` to insert the current item $z \in [N]$ into the ranked list with an error ϵ, and iteratively calls `ATI` by decreasing the error until the probability that item z is inserted to the correct position is high enough. It is displayed in Algorithm 3.5.

3. `Attempting-Insertion` (`ATI`): the subroutine that traverses the Preference Interval Tree using binary search (Feige et al., 1994) to find the node where the item should be inserted with error ϵ. To compare the current item and any node in the tree, it calls `ATC` to obtain the comparison result. It is displayed in Algorithm 3.6.

4. `Attempting-Comparison` (`ATC`): the subroutine that adaptively samples queries from a subset of users for a pair of items (z, j), where z is the item currently being inserted and j is any other item. `ATC` records the number of queries each user provides and the results of the comparisons. It is displayed in Algorithm 3.4.

In the heterogeneous rank aggregation problem, each user may have a different accuracy level from the others. Therefore, we adaptively sample the comparison data from a subset of users. In particular, we maintain an active set $\mathcal{U} \subseteq [M]$ of users, which contains the potentially most accurate users from the entire group. We add a user elimination phase to the main procedure (Algorithm 3.2) based on the elimination idea in multi-armed bandits (Slivkins et al., 2019; Lattimore and Szepesvári, 2020) to update this active set. In particular, we view each user as an arm in a multi-armed bandit, where the reward is 1 if the answer from a certain user is correct and 0 if wrong. After an item is successfully inserted by `IAI`, we call Algorithm 3.3 (`EliminateUser`) to eliminate users with low accuracy levels before we proceed to the next item.

To estimate the accuracy levels of users, a vector $\mathbf{s}_z \in \mathbb{R}^M$, recording the counts of responses from each user for item z, is maintained during the whole period of inserting item z. We further keep track of two matrices $A_z, B_z \in \mathbb{R}^{N\times M}$. When a pair (z, j) (where z refers to the item currently being inserted and j to an arbitrary item) is compared by user $u \in \mathbb{R}^M$ in Algorithm 3.4, we increase $A[j, u]$ by 1 if user u thinks z is better than j and increase $B[j, u]$ by 1 otherwise. We use w to record the total number of times that

item z is deemed better by any users and use the average $\widehat{p} = w/t$ to provide an estimation of the average accuracy $|\mathcal{U}|^{-1}\sum_{u\in\mathcal{U}} p_{ij}^u$. The variables A_z, B_z, and $\mathbf{s}_z$ are global variables, shared by different subroutines throughout the process. After an item z is successfully inserted, A_z, B_z will be discarded and the space allocated can be used for A_{z+1}, B_{z+1} (See Line 32 of Algorithm 3.2).

We use the 0/1 reward for each user to indicate whether the provided pairwise comparison is correct. Nevertheless, this reward is not known immediately after each arm-pull since the correctness depends on the ranking of items which is also unknown. But when `IAI` returns *inserted*, the item recently inserted has a high probability to be in the right place. Our method takes advantage of this fact by constructing a fairly accurate prediction of pairwise comparison for the item with all other already inserted items in the PIT. Then an estimate of the reward $\boldsymbol{n}_z$ can be obtained with the help of recorded responses A_z and B_z, which are updated in `ATC` as described in the preceding paragraph. At last, in Algorithm 3.3 a UCB-style condition is imposed on estimated accuracy levels $\boldsymbol{\mu} = \boldsymbol{n}_z/\boldsymbol{s}_z$.

We borrow the definition of Preference Interval Tree (PIT) (Feige et al., 1994; Ren et al., 2019) based on which we can insert items to a ranked list. Specifically, given a list of ranked items S the PIT can be constructed using the following Algorithm 3.1.

Algorithm 3.1 Build PIT

Input parameters: S

Data structure: `Node` $= \{left, mind, right, lchild, rchild, parent\}$, $left, mid, right$ holds index values, $lchild, rchild, parent$ points to any other `Node`.

Initialize: $N = |S|$

$X =$ `CreateEmptyNode` returns an empty Node with above mentioned data structure
X.left $= -1$
X.right $= |S|$
X.mid $= \lfloor (X.left + X.right)/2 \rfloor$
queue $=$ [X]
while queue.`NotEmpty` **do**
  X $=$ queue.`PopFront`
  X.mid $= \lfloor (X.left + X.right)/2 \rfloor$
  if X.right - X.left > 1 **then**
    lnode $=$ `CreateEmptyNode`
    lnode.left $=$ X.left
    lnode.right $=$ mid
    X.lchild $=$ lnode
    rnode $=$ `CreateEmptyNode`
    queue.append(lnode)
    rnode.left $=$ X.mid
    rnode.right $=$ X.right
    X.rchild $=$ rnode
    queue.append(rnode)
  end if
end while
replace -1 with $-\infty$, $|S|$ with ∞ in each `Node`.left and `Node`.right.

For the completeness of our paper, we also present the subroutines `Iterative-Attempting-Insertion` (`IAI`) and `Attempting-Insertion` (`ATI`) in this section, which are omitted in Section 3.4 due to space limit. In particular, `IAI` is displayed in Algorithm 3.5 and `ATI` is displayed in Algorithm 3.6. Both algorithms are proposed by Ren et al. (2019) for adaptive sampling in the single user setting.

3.4.1 A Two-stage Algorithm as Baseline

In this section, we present an alternative simple scheme, called two-stage ranking with a heterogeneous set of users. This provides another baseline with which we can compare `Ada-IIR`. Additionally, it can be useful in situations with a large number of users, i.e., $M = \Omega(\sqrt{N})$, where `Ada-IIR` is less effective.

Algorithm 3.2 Main Procedure: `Adaptive Iterative-Insertion-Ranking` (`Ada-IIR`)

Global Variables:
$z \in \mathbb{N}$: the index of the item being inserted into the ranked list.
$A_z \in \mathbb{R}^{N \times M}$: $A_z[j,u]$ is the number of times that user u thinks item z is better than item j.
$B_z \in \mathbb{R}^{N \times M}$: $B_z[j,u]$ is the number of times that user u thinks item z is worse than item j.
$\mathbf{s}_z \in \mathbb{R}^M$: total number of responses by each user so far.
Input parameters: Items to rank $S = [N]$ and confidence δ
Initialize: $\boldsymbol{n}_1 = \boldsymbol{s}_1 = \mathbf{0}$

$Ans \leftarrow$ the list containing only $S[1]$
for $z \leftarrow 2$ to $|S|$ **do**
  $\boldsymbol{n}_z = \boldsymbol{n}_{z-1}, \boldsymbol{s}_z = \boldsymbol{s}_{z-1}, A_z = \mathbf{0}, B_z = \mathbf{0}$
  `IAI`$(S[z], Ans, \delta/(n-1))$ ▷Algorithm 3.5 (global variables $A_z, B_z, \boldsymbol{s}_z$ are updated here)
  for $j \in [z-1]$ **do**
    if $S[z] > S[j]$ in PIT **then**
      $\boldsymbol{n}_z = \boldsymbol{n}_z + A_z[j, *]$
    else
      $\boldsymbol{n}_z = \boldsymbol{n}_z + B_z[j, *]$
    end if
  end for
  $\mathcal{U}_z \leftarrow$ `EliminateUser`$(\mathcal{U}_{z-1}, \mathbf{n}_z, \mathbf{s}_z, \delta/(n-1))$ ▷Algorithm 3.3
end for
return Ans;

Algorithm 3.3 Subroutine: `EliminateUser`

Input parameters: $(\mathcal{U}, \mathbf{n}, \mathbf{s}, \delta)$.

Set $S = \sum_{u \in [M]} \mathbf{s}_u$, $\mathbf{s}_{\min} = \min_{u \in \mathcal{U}} \mathbf{s}_u$, $\boldsymbol{\mu}_u = \mathbf{n}_u / \mathbf{s}_u$, $r = \sqrt{\log(2|\mathcal{U}|/\delta)/(2\mathbf{s}_{\min})}$
Set $\mathbf{LCB} = \boldsymbol{\mu} - r\mathbf{1}$ and $\mathbf{UCB} = \boldsymbol{\mu} + r\mathbf{1}$.
if $S \geq 2M^2 \log(NM/\delta)$ **then**
  for $u \in \mathcal{U}$ **do**
    Remove user u from $\mathcal{U}$ if $\exists u' \in \mathcal{U}, \mathbf{UCB}_u < \mathbf{LCB}_{u'}$.
  end for
end if
return $\mathcal{U}$

Algorithm 3.4 Subroutine: `Attempt-To-Compare (ATC)` $(z, j, \mathcal{U}, \epsilon, \delta)$

Input: items (z, j) to be compared, set of users $\mathcal{U}$, confidence parameter ϵ, δ. M is the number of users originally.

$m = |\mathcal{U}|, \widehat{p} = 0, w = 0, \widehat{y} = 1$. Number of rounds $r = 1$. $r_{\max} = \lceil \frac{1}{2}\epsilon^{-2} \log \frac{2}{\delta} \rceil$.
while $r \leq r_{\max}$ **do**
  Choose u uniformly at random from $\mathcal{U}$
  Obtain comparison result from user u as y_{ij}^u
  Increment the counter of responses collected from this user $\boldsymbol{s}_z[u] \leftarrow \boldsymbol{s}_z[u] + 1$
  if $y_{ij}^u > 0$ **then**
    $A_z[j, u] \leftarrow A_z[j, u] + 1$, $w \leftarrow w + 1$
  else
    $B_z[j, u] \leftarrow B_z[j, u] + 1$
  end if
  $\widehat{p} \leftarrow w/r$, $r \leftarrow r + 1$, $c_r \leftarrow \sqrt{\frac{1}{2t} \log(\frac{\pi^2 r^2}{3\delta})}$
  if $|\widehat{p} - \frac{1}{2}| \geq c_r$ **then**
    break
  end if
end while
if $\widehat{p} \leq \frac{1}{2}$ **then**
  $\widehat{y} = 0$
end if
return: $\widehat{y}$

Algorithm 3.5 Subroutine: `Iterative Attempt To Insert(IAI)`

Input parameters: (i, S, δ)
Initialize: For all $\tau \in \mathbb{Z}^+$, set $\epsilon_\tau = 2^{-(\tau+1)}$ and $\delta_\tau = \frac{6\delta}{\pi^2 \tau^2}$; $t \leftarrow 0$; $Flag \leftarrow$ *unsure*;

repeat
  $t \leftarrow t + 1$;
  $Flag \leftarrow$`ATI`$(i, S, \epsilon_\tau, \delta_\tau)$;
until $Flag =$ *inserted*

Algorithm 3.6 Subroutine: `Attempt To Insert(ATI)`.

Input parameters: (i, S, ϵ, δ)

Initialize: Let z be a PIT constructed from S, $h \leftarrow \lceil 1 + \log_2(1 + |S|) \rceil$, the depth of z

For all leaf nodes u of z, initialize $c_u \leftarrow 0$; Set $t^{\max} \leftarrow \lceil \max\{4h, \frac{512}{25} \log \frac{2}{\delta}\} \rceil$ and $q \leftarrow \frac{15}{16}$

$X \leftarrow$ the root node of z;
for $t \leftarrow 1$ to $t^{\max}$ **do**
  if X is the root node **then**
    if $\texttt{ATC}(i, X.\text{mid}, \epsilon, 1 - q) = i$ **then**
      $X \leftarrow X.\text{rchild}$
    else
      $X \leftarrow X.\text{lchild}$
    end if
  else if X is a leaf node **then**
    if $\texttt{ATC}(i, X.\text{left}, \epsilon, 1 - \sqrt{q}) = i \wedge \texttt{ATC}(i, X.\text{right}, \epsilon, 1 - \sqrt{q}) = X.\text{right}$ **then**
      $c_X \leftarrow c_X + 1$
      if $c_X > b^t := \frac{1}{2}t + \sqrt{\frac{t}{2} \log \frac{\pi^2 t^2}{3\delta}} + 1$ **then**
        Insert i into the corresponding interval of X and
        return *inserted*
      end if
    else if $c_X > 0$ **then**
      $c_X \leftarrow c_X - 1$
    else
      $X \leftarrow X.\text{parent}$
    end if
  else
    if $\texttt{ATC}(i, X.\text{left}, \epsilon, 1 - \sqrt[3]{q}) = X.\text{left} \vee \texttt{ATC}(i, X.\text{right}, \epsilon, 1 - \sqrt[3]{q}) = i$ **then**
      $X \leftarrow X.\text{parent}$
    else if $\texttt{ATC}(i, X.\text{mid}, \epsilon, 1 - \sqrt[3]{q}) = i$ **then**
      $X \leftarrow X.\text{rchild}$
    else
      $X \leftarrow X.\text{lchild}$
    end if
  end if
end for
if there is a leaf node u with $c_u \geq 1 + \frac{5}{16}t^{\max}$ **then**
  Insert i into the corresponding interval of u and
  return *inserted*
else
  return *unsure*
end if

Two-stage ranking first performs user-selection and then item-ranking. In the user-selection stage, we search for an α-optimal user for some small α. Specifically, we first take an arbitrary pair of items (i, j) and then run the Iterative-Insertion-Ranking (`IIR`) algorithm (see Theorem 3.3.3) on them to determine the order, e.g., $i \succ j$, with high probability. Note that at this point, users have not been distinguished yet. So we take each query from a randomly chosen user. As discussed in Section 3.3.2, this is equivalent to querying the user $\bar{u}$ whose accuracy is $\bar{\Delta}_0$. Given $i \succ j$, the problem of finding an α-optimal user is reduced to pure exploration of an α-optimal arm in the context of multi-armed bandit: making queries about the pre-determined item pair from user u is the same as generating outcomes from an arm with Bernoulli$(\frac{1}{2}+\Delta_u)$ reward, e.g., if user u returns the answer $i \succ j$ then we get reward 1, otherwise we get reward 0. Hence, an α-optimal user is equivalent of an α-optimal arm. For determining an α-optimal arm, we can adopt the Median-Elimination (ME) algorithm from Even-Dar et al. (2002). After ME returns an α-optimal user u_α, we discard all other users and rank items by only querying u_α. Ranking with a single user can again be done by the `IIR` algorithm. In summary, two-stage ranking is composed of three procedures: `IIR` for determining the order of i and j, ME for obtaining an α-optimal user, and `IIR` again for producing the final ranking. A more formal statement of two-stage ranking is presented in Wu et al. (2022).

3.5 Theoretical Analysis

3.5.1 Sample Complexity of the Proposed Algorithm

First, we define the function $F(x)$ as follows:

$$F(x) = x^{-2}(\log\log x^{-1} + \log(N/\delta)). \tag{3.5.1}$$

Define $\bar{\Delta}_z$ to be the average accuracy of all users in the current active set.

$$\bar{\Delta}_z = \frac{1}{\mathcal{U}_z}\sum_{u\in\mathcal{U}_z}\Delta_u \tag{3.5.2}$$

We then present an upper bound on the sample complexity of `Ada-IIR` (Algorithm 3.2). Although $F(x)$ depends on N and δ^{-1}, the dependence is only logarithmic, and it does not affect the validity of reasoning via big-O notations.

Theorem 3.5.1. For any $\delta > 0$, with probability at least $1-\delta$, Algorithm 3.2 returns the exact ranking of the N items, and it makes at most $\mathcal{C}_{\text{Alg}}(N)$ queries, where $\mathcal{C}_{\text{Alg}}(N) = O(\sum_{z=2}^{N}\bar{\Delta}_z^{-2}(\log\log\bar{\Delta}_z^{-1} + \log(N/\delta))) = O(\sum_{z=2}^{N}F(\bar{\Delta}_z))$.

Proof of Theorem 3.5.1. The analysis on the sample complexity follows a similar route as Ren et al. (2019) due to the similarity in algorithm design. In fact, since we randomly choose a user from $\mathcal{U}_t$ and query it for a feedback, it is equivalent to querying a single user with the averaged accuracy $\frac{1}{2}+\bar{\Delta}_z$, where $\bar{\Delta}_z := \frac{1}{|\mathcal{U}_z|}\sum_{u\in\mathcal{U}_z}\Delta_u$. This means most of the theoretical results from Ren et al. (2019) can also apply to our algorithm. The following lemmas characterize the performance of each subroutine:

Lemma 3.5.2 (Lemma 9 in Ren et al. (2019))**.** For any input pair (i, j) and a set of users $\mathcal{U}$, Algorithm 3.4 terminates in $\lceil r_{\max}\rceil = \lceil\epsilon^{-2}\log(2/\delta)\rceil$ queries. If $\epsilon \le \bar{\Delta}$, then the returned $\widehat{y}$ indicates the preferable item with probability at least $1-\delta$.

Lemma 3.5.3 (Lemma 10 in Ren et al. (2019))**.** Algorithm 3.6 returns after $O(\epsilon^2\log(|S|/\delta)$ queries and, with probability $1-\delta$, correctly insert or return unsure. Additionally, if $\epsilon \le \bar{\Delta}$, Algorithm 3.6 will insert correctly with probability $1-\delta$.

Lemma 3.5.4 (Lemma 11 in Ren et al. (2019))**.** With probability $1-\delta$, Algorithm 3.5 correctly insert the item and makes $O(\bar{\Delta}^{-2}(\log\log\bar{\Delta}^{-1} + \log(N/\delta)))$ queries at most.

When inserting the z-th item, we makes at most $\bar{\Delta}_z^{-2}(\log\log\bar{\Delta}_z^{-1}+\log(N/\delta))$ queries, for $z = 2, 3, \ldots, N$.

The number of total queries can be obtained by summing up the term above, which is

$$\mathcal{C}_{\mathrm{Alg}}(N) = O\bigg(\sum_{z=2}^{N} \bar{\Delta}_z^{-2}(\log\log \bar{\Delta}_z^{-1} + \log(N/\delta))\bigg).$$

□

3.5.2 Sample Complexity Gap Analysis

While Theorem 3.5.1 characterizes the sample complexity of Algorithm 3.2 explicitly, the result is not directly comparable with the sample complexity of the oracle algorithm that only queries the best user $\mathcal{C}_{u^*}(N)$ or the complexity of the naive random-query algorithm $\mathcal{C}_{\mathrm{ave}}(N)$. Based on Theorem 3.5.1, we can derive the following more elaborate sample complexity for Algorithm 3.2.

Theorem 3.5.5. Suppose there are N items and M users initially. Denote $S_z = \sum_{u\in[M]}(\mathbf{s}_z)_u$ to be the number of all queries made before inserting item z (Line 4 in Algorithm 3.2). The proposed algorithm has the following sample complexity upper bound:

$$\begin{aligned}\mathcal{C}_{\mathrm{Alg}}(N,M) = \Theta(NF(\Delta_{u^*})) &+ O\bigg(\sum_{z=2}^{N} \mathbb{1}\left\{S_z < 2M^2\log(NM/\delta)\right\}\left(F(\bar{\Delta}_0) - F(\Delta_{u^*})\right)\bigg) \\ &+ O\bigg(L(\mathcal{U}_0)\sqrt{\log(2MN/\delta)}\sum_{z=2}^{N}\mathbb{1}\{S_z \geq 2M^2\log(NM/\delta)\}\sqrt{\frac{M}{S_z}}\bigg),\end{aligned} \tag{3.5.3}$$

where $L(\mathcal{U}_0) = \frac{F(c\Delta_{u^*}^3) - F(\Delta_{u^*})}{\Delta_{u^*} - c\Delta_{u^*}^3}$ is an instance-dependent factor, with only logarithmic dependence on N and δ^{-1}(through F), and where $c = 1/25$ is a global constant.

The first lemma we will introduce is about the confidence interval:

Lemma 3.5.6. With probability $1-\delta$, it holds for any $z \in [N]\backslash\{1\}$ and $u \in \mathcal{U}_z$,

$$\frac{1}{2} + \Delta_u \in \Big[(\mathbf{LCB}_z)_u, (\mathbf{UCB}_z)_u\Big].$$

This also indicates that when inserting the z-th item, for any $u \in \mathcal{U}_z$,

$$\Delta_{u^*} - \Delta_u \leq 4r_z.$$

Proof of Theorem 3.5.6. Recall that $(\boldsymbol{\mu}_z)_u$ is the empirical mean of the Bernoulli variable with parameter $\frac{1}{2} + \Delta_u$. For a given z and u, by Hoeffding's inequality we have

$$\mathbb{P}\bigg(\Big|(\boldsymbol{\mu}_z)_u - \Big(\frac{1}{2} + \Delta_u\Big)\Big| > r_z\bigg) \leq 2e^{-2(\mathbf{s}_z)_u r_u^2} \leq 2e^{-2(\mathbf{s}_z)_{\min} r_u^2} \leq \frac{\delta}{|\mathcal{U}_z|N},$$

and applying union bound over $z = 2, 3, \ldots, N$ and $u \in \mathcal{U}_z$ gives the claim.

Under this event, we have

$$\begin{aligned}\Delta_{u^*} - \Delta_u &= \Big(\frac{1}{2} + \Delta_{u^*}\Big) - \Big(\frac{1}{2} + \Delta_u\Big) \\ &\leq (\mathbf{UCB}_z)_{u^*} - (\mathbf{LCB}_z)_u \\ &\leq (\mathbf{UCB}_z)_{u^*} - (\mathbf{LCB}_z)_{u^*} + (\mathbf{UCB}_z)_u - (\mathbf{LCB}_z)_u \\ &= 4r_z,\end{aligned}$$

where the first inequality is clearly from the confidence interval, and the second inequality holds because the two confidence intervals should intersect. □

Next, we will introduce another lemma concerning the growth of $(\mathbf{s}_z)_u$ for each $u \in \mathcal{U}_z$.

Lemma 3.5.7. Denote S_z as all queries made till inserting the z-th item and $M = |\mathcal{U}_0|$. Suppose $S_z \geq 2M^2 \log(NM/\delta)$. With probability $1-\delta$, we have for any $z \in \{2, 3, \ldots, N\}$,

$$(\mathbf{s}_z)_{\min} \geq \frac{S_z}{2M}.$$

Proof of Theorem 3.5.7. For fixed z and $u \in \mathcal{U}_z$, by Hoeffding's inequality we have

$$\begin{aligned}\mathbb{P}\bigg(\frac{(\mathbf{s}_z)_u}{S_z} - \frac{1}{M} < -\frac{1}{2M}\bigg) &\leq \mathbb{P}\bigg(\frac{(\mathbf{s}_z)_u}{S_z} - \mathbb{E}\bigg[\frac{(\mathbf{s}_z)_u}{S_z}\bigg] < -\frac{1}{2M}\bigg) \\ &\leq \exp\bigg(-\frac{S_z}{2M^2}\bigg) \leq \frac{\delta}{NM}.\end{aligned}$$

Applying union bound we know that with probability $1-\delta$,

$$(\mathbf{s}_z)_u \geq \frac{S_z}{2M}, \forall z \in \{2, 3, \ldots, N\}, \forall u \in \mathcal{U}_z.$$

Since $(\mathbf{s}_z)_{\min} := \min_{u \in \mathcal{U}_z} (\mathbf{s}_z)_u$, we have

$$(\mathbf{s}_z)_{\min} \geq \frac{S_z}{2M}, \forall z \in \{2, 3, \ldots, N\}.$$

□

With the two lemmas above, we can control the accuracy gap as follows:

Lemma 3.5.8. Denote $\bar{\Delta}_z = \frac{1}{|\mathcal{U}_z|} \sum_{u \in \mathcal{U}_z} \Delta_u$. Suppose $S_z \geq 2|M|^2 \log(NM/\delta)$. With probability $1-2\delta$, we have for any $t \in [N]$,

$$\Delta_{u^*} - \bar{\Delta}_z \leq \mathrm{polylog}(N, M, \delta^{-1}) \cdot \sqrt{\frac{M}{S_z}}.$$

Proof of Theorem 3.5.8. The proof has two steps:

From Theorem 3.5.7 we know that with probability $1-\delta$,

$$(\mathbf{s}_z)_{\min} \geq \frac{S_z}{2M}, \forall t \in [N], \forall u \in \mathcal{U}_z.$$

From Theorem 3.5.6, we know with probability $1-\delta$(recall that $(\mathbf{r}_z)_u = \sqrt{\frac{\log(2|\mathcal{U}_z|N/\delta)}{2(\mathbf{s}_z)_{\min}}}$),

$$\begin{aligned}\Delta_{u^*} - \Delta_u &\leq 4r_z \\ &\leq 4\sqrt{\frac{M \log(2MN/\delta)}{S_z}} \\ &= 4\sqrt{\log(2MN/\delta)} \cdot \sqrt{\frac{M}{S_z}}.\end{aligned}$$

□

Define function $F(x) = x^{-2}(\log\log(x^{-1}) + \log(N/\delta))$ with $x \in (0, 1/2]$. We care about the following term GAP which characterize the query complexity gap between our algorithm and the optimal user.

$$\mathrm{GAP}(N, M, \delta) = \sum_{z=2}^{N} F(\bar{\Delta}_z) - F(\Delta_{u^*}).$$

The following lemma provide a way to linear bound the gap between function values:

Lemma 3.5.9. $F(x) = x^{-2}(\log\log(x^{-1}) + \log(N/\delta))$ with $x \in (0, 1/2]$ is a convex function over $(0, 1/2]$, and for any $\Delta \in [a, b]$, we have

$$F(\Delta) - F(b) \le \frac{F(a) - F(b)}{b - a} \cdot (b - \Delta) = L(a, b) \cdot (b - \Delta).$$

Furthermore, under the event of Theorem 3.5.8, for any $z \in [N]$ such that $S_z > 2M^2 \log(NM/\delta)$, we have $\bar{\Delta}_z \in [c\Delta_{u^*}^3, \Delta_{u^*}]$ and therefore

$$F(\bar{\Delta}_z) - F(\Delta_{u^*}) \le \frac{F(c\Delta_{u^*}^3) - F(\Delta_{u^*})}{\Delta_{u^*} - c\Delta_{u^*}^3} \cdot (\Delta_{u^*} - \bar{\Delta}_z) = L(\mathcal{U}_0) \cdot (\Delta_{u^*} - \bar{\Delta}_z).$$

Here we use $L(\mathcal{U}_0) = \frac{F(c\Delta_{u^*}^3) - F(\Delta_{u^*})}{\Delta_{u^*} - c\Delta_{u^*}^3}$ is indeed a instance-dependent factor, with only logarithmic dependent in N and δ^{-1}(in F). c is a global constant and in fact $c = 1/25$.

Proof. Differentiate $F(x)$ twice and it can be verified that $F''(x) > 0$. For any $\Delta \in [a, b]$, the inequality above is easy to prove via convexity.

The rest is to prove that $\forall t \in [N]$, we have $\bar{\Delta}_z \in [\Delta_{u^*}/M, \Delta_{u^*}]$. It is clear that the upper bound holds because $\Delta_{u^*} := \max_{u \in \mathcal{U}_0} \Delta_u$.

The lower bound is proved as follows: We still have $\bar{\Delta}_z > \Delta_{u^*}/M$ because at any time u^* always remains in the user set and by the assumption $\Delta_u > 0$.

Also, since $S_z > 2M^2 \log(NM/\delta)$, by Theorem 3.5.8, we have

$$\begin{aligned}
\Delta_{u^*} - \bar{\Delta}_z &\le 4\sqrt{\frac{M \log(2MN/\delta)}{S_z}} \\
&\le 4\sqrt{\frac{M \log(2MN/\delta)}{2M^2 \log(NM/\delta)}} \\
&\le \frac{4}{\sqrt{M}}.
\end{aligned}$$

Now we will prove that

$$\max\left\{\frac{\Delta_{u^*}}{M}, \Delta_{u^*} - \frac{4}{\sqrt{M}}\right\} \ge c\Delta_{u^*}^3.$$

Suppose $\frac{\Delta_{u^*}}{M} < c\Delta_{u^*}^3$, then we have $M > c^{-1}\Delta_{u^*}^{-2}$, this means

$$\Delta_{u^*} - \frac{4}{\sqrt{M}} \ge \Delta_{u^*} - 4\sqrt{c}\Delta_{u^*} \ge c\Delta_{u^*}^3.$$

The last inequality is due to $\Delta_{u^*} \le 1/2$ and $c = 1/25$. □

Now we are ready to prove the main result:

Proof of Theorem 3.5.5. Based on our algorithmic design, we will not eliminate any user until the cumulative number of queries S_z reach the threshold $S_z \ge 2M^2 \log(NM/\delta)$. We have

$$\begin{aligned}
\mathrm{GAP}(N, M, \delta) &= \sum_{z=2}^{N} F(\bar{\Delta}_z) - F(\Delta_{u^*}) \\
&= \underbrace{\sum_{z=2}^{N} \mathbb{1}\{S_z < 2M^2 \log(NM/\delta)\}\big(F(\bar{\Delta}_z) - F(\Delta_{u^*})\big)}_{I_1} \\
&\quad + \underbrace{\sum_{z=2}^{N} \mathbb{1}\{S_z \ge 2M^2 \log(NM/\delta)\}\big(F(\bar{\Delta}_z) - F(\Delta_{u^*})\big)}_{I_2}.
\end{aligned}$$

For I_1, no elimination is performed, so $\mathcal{U}_z = \mathcal{U}_0$, and we have

$$I_1 = \sum_{z=2}^{N} \mathbb{1}\{S_z < 2M^2 \log(NM/\delta)\}\big(F(\bar{\Delta}_0) - F(\Delta_{u^*})\big).$$

For each term in I_2, we have $F(\bar{\Delta}_z) - F(\Delta_{u^*}) \le L(\mathcal{U}_0) \cdot 4\sqrt{\log(2MN/\delta)} \cdot \sqrt{\frac{M}{S_z}}$ due to Theorem 3.5.9 and Theorem 3.5.8. Therefore,

$$I_2 \le L(\mathcal{U}_0) 4\sqrt{\log(2MN/\delta)} \sum_{z=2}^{N} \mathbb{1}\{S_z \ge 2M^2 \log(NM/\delta)\} \sqrt{\frac{M}{S_z}}.$$

□

3.5.3 Discussion on the Sample Complexity Gap and the Optimality of the Proposed Algorithm

A few discussions are necessary to show the meaning of the result in previous section. First, if the number of users $M \gg N$, then no user is eliminated because each user will be queried so few times that no meaningful inference can be made. Since the goal is to achieve the accuracy of the best user, more inaccurate users only make the task more difficult. Therefore, it is necessary to impose assumptions on M with respect to N.

This intuition can be made more precise. Suppose we loosely bound S_t as $S_t \ge t \log(t/\delta)$, which is reasonable since for a very accurate user the algorithm will spend roughly no more than $O(\log(t/\delta))$ comparisons to insert one item. This means the complexity can be bounded as (ignoring log factors)

$$\mathcal{C}_{\text{Alg}}(N, M) = O\big(NF(\Delta_{u^*})\big) + \widetilde{O}\big(M^2\big(F(\bar{\Delta}_0) - F(\Delta_{u^*})\big)\big) + \widetilde{O}\big(L(\mathcal{U}_0)\big(\sqrt{M}(\sqrt{N} - M)\big)\big). \tag{3.5.4}$$

If $M = \Omega(\sqrt{N})$, then this is not ideal because our algorithm won't eliminate any user until $\Omega(N)$ items are inserted with accuracy $\bar{\Delta}_0$, which already leads to a gap linear in N compared with the best complexity $\mathcal{C}_{u^*}$. In this case, our algorithm roughly makes the same amount of queries as $\mathcal{C}_{\text{ave}}$.

In order to avoid the bad case, it is necessary to assume $M = o(\sqrt{N})$ so that the last two terms become negligible (notice that $L(\mathcal{U}_0)$ is an instance-dependent constant). Now we restate Theorem 3.5.5 with the additional assumption, and compare it with the baselines.

Proposition 3.5.10. Suppose we have M users and N items to rank exactly, with $M = o(\sqrt{N})$. We have the following complexity along with (3.3.2) and (3.3.3):

$$\begin{aligned}
\mathcal{C}_{u^*}(N, M) &= \Theta(NF(\Delta_{u^*})), \\
\mathcal{C}_{\text{ave}}(N, M) &= \Theta(NF(\bar{\Delta}_0)), \\
\mathcal{C}_{\text{Alg}}(N, M) &= \Theta(NF(\Delta_{u^*})) + o(N\big(F(\bar{\Delta}_0) - F(\Delta_{u^*})\big)) + o\big(N\big).
\end{aligned}$$

The last two terms of $\mathcal{C}_{\text{Alg}}(N, M)$ are negligible when compared with the first term. Therefore, our algorithm can perform comparably efficiently as if the best user were known while enjoying an advantage over the naive algorithm with sample complexity $\mathcal{C}_{\text{ave}}(N, M)$.

Proof of Theorem 3.5.10. Suppose $M = o(N^{1/2})$, since $S_z \ge z \log(z/\delta) \ge z$(at least one comparison for an item), from (3.5.3) we have

$$\sum_{z=2}^{N} \mathbb{1}\left\{S_z < 2M^2 \log(NM/\delta)\right\} \le \sum_{z=2}^{N} \mathbb{1}\left\{z < 2M^2 \log(NM/\delta)\right\} = o(N).$$

The third term can be bounded with the fact $\mathbb{1}\left\{z < 2M^2 \log(NM/\delta)\right\} \le 1$,

$$\begin{aligned}
&L(\mathcal{U}_0)\sqrt{\log(2MN/\delta)}\sum_{z=2}^{N}\mathbb{1}\{S_z \ge 2M^2\log(NM/\delta)\}\sqrt{\frac{M}{S_z}} \\
&\le L(\mathcal{U}_0)\sqrt{\log(2MN/\delta)}\sum_{z=2}^{N}\sqrt{\frac{M}{S_z}} \\
&\le L(\mathcal{U}_0)\sqrt{\log(2MN/\delta)}\sum_{z=2}^{N}\sqrt{\frac{M}{z}} \\
&\le 2L(\mathcal{U}_0)\sqrt{\log(2MN/\delta)}\sqrt{MN} \\
&= O(L(\mathcal{U}_0)\sqrt{\log(MN/\delta)}\sqrt{MN}).
\end{aligned}$$

□

$L(\mathcal{U}_0)$ is actually dominated by the minimal mean accuracy $\min_z \bar{\Delta}_z$ throughout the algorithm. In practice, $L(\mathcal{U}_0)$ is usually a constant, related to all users' accuracy. In the worst theoretical case, $L(\mathcal{U}_0)$ will be dominated by $F(\Delta_{u^*}/M) = \widetilde{O}(M^2)$, which further turns the last term into $\widetilde{O}(M^{5/2}N^{1/2})$, and requires $M = o(N^{1/5})$ so that this term becomes negligible.

Remark 3.5.11. Note that if we set $\mathcal{U}_0 = \{u^*\}$ for our algorithm, it will achieve exactly the same complexity as (3.3.2) indicates. Similarly, if we construct a new user $\bar{u}$ where $\Delta_{\bar{u}} = \bar{\Delta}_0$ and set $\mathcal{U}_0 = \{\bar{u}\}$, our algorithm will recover exactly (3.3.3). By this argument and the fact that Big-O notations hide no M, the first term in each equation actually has the same absolute constant factor. Therefore, our algorithm is indeed comparable with the best user.

Remark 3.5.12. Notice that $F(x) \to +\infty$ when $x \to 0$. This means $\mathcal{C}_{\text{ave}}$ is very sensitive to the initial average accuracy margin $\bar{\Delta}_0$. In the case where there is only one best user u^* and all other users have a near-zero margin $\Delta_u \to 0$, $\mathcal{C}_{\text{ave}}$ can be very large compared with $\mathcal{C}_{u^*}$.

Remark 3.5.13. In the experiments, we notice that even with $N = 10$ and $M = 9$, after inserting the first item, each user has already been queried for enough times so that $S_2 \ge 2M^2 \log(NM/\delta)$, which makes the second term in (3.5.3) vanish.

3.6 Experiments

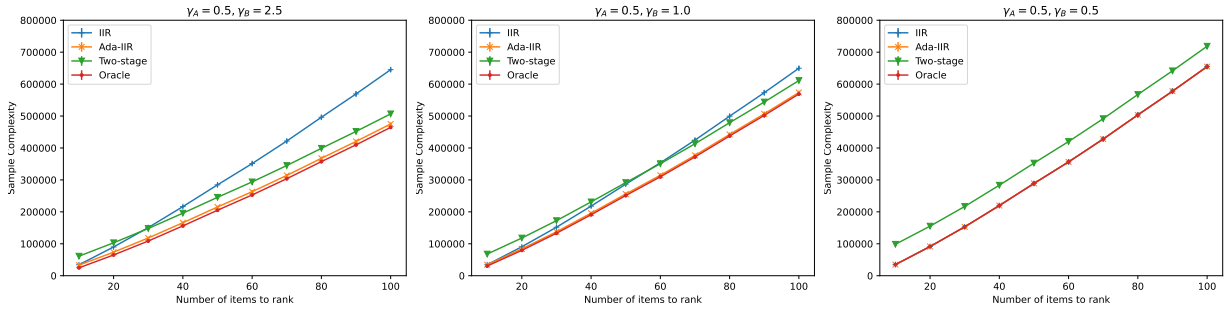
In this section, we study the empirical performance of the following algorithms on both synthetic and real-world datasets:

- `IIR` (Ren et al., 2019): The original single-user algorithm adapted to the multi-user case by querying a user selected uniformly at random.
- `Ada-IIR`: The proposed method.
- `Two-stage ranking`: A simple method described in Section 3.4.1.
- `Oracle`: Query only the best user as if it is known.

Confidence parameter $\delta = 0.25$, $\alpha = 0.05$ is set if required by algorithm.

3.6.1 Synthetic Experiment

In our experiment, we use a similar setup as that of Section 2.5, except that every pair has same disatnace. In particular, we consider a set of users $[M]$, whose accuracies are set by $p_u(i,j) = (1+\exp(\gamma_u(s_j - s_i)))^{-1}$, for $u \in [M]$ and items $i, j \in [N]$, where parameter γ_u determines the user accuracy and s_i, s_j are the utility scores of the corresponding items in the BTL model. Larger values of γ_u lead to more accurate users. We



(a) $\gamma_A = 0.5$, $\gamma_B = 2.5$ (b) $\gamma_A = 0.5$, $\gamma_B = 1.0$ (c) $\gamma_A = 0.5$, $\gamma_B = 0.5$

Figure 3.1: Sample complexities v.s. number of items for all algorithms. (a) (b) and (c) are different heterogeneous user settings where the accuracy of two group of users differs.

set $s_i - s_j = 3$ if $i \prec j$ and $s_i - s_j = -3$ otherwise. Note that here we assume that the accuracy of user u is the same for all pair of items (i, j) as long as $i \prec j$. We assume that there are two distinct groups of users: the high-accuracy group in which the users have the same accuracy $\gamma_u = \gamma_B \in \{0.5, 1.0, 2.5\}$ in three different settings; and the low-accuracy group in which the users have the same accuracy $\gamma_u = \gamma_A = 0.5$ in all settings. This set of γ_u, s_i, s_j is chosen so that $p_u(i, j)$ for accurate users ranges from 0.55 to 0.99 and inaccurate users have a value close to 0.55.

The number of items to be ranked ranges from 10 to 100. Each setting is repeated 100 times with randomly generated data. To showcase the effectiveness of active user selection, we tested a relatively adverse situation where only 12 out of $M = 36$ users are highly accurate.

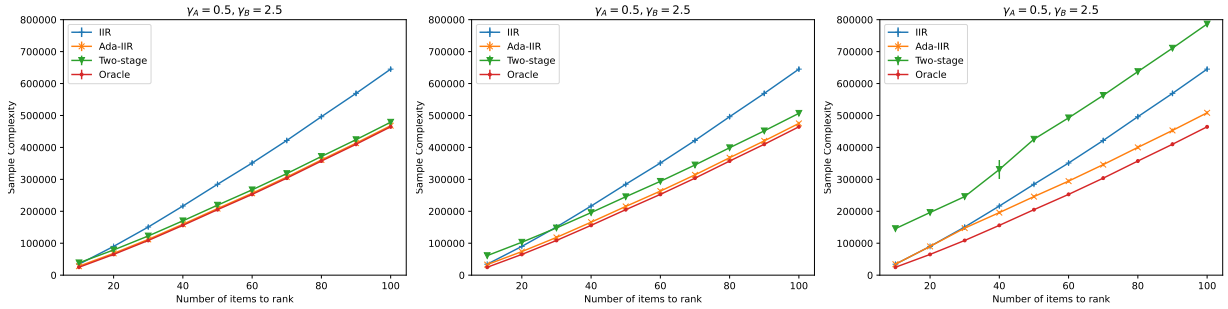
The average sample complexity and standard deviation over 100 runs are plotted in Fig. 3.1. Note that the standard deviation is hard to see, given that it is small compared to the average. In most cases, the proposed method achieves nearly identical performance to the oracle algorithm, with only a small overhead. For two-stage ranking, we observe a constant overhead regardless the accuracy of the users. It may outperform the non-adaptive one (`IIR`) if there exist enough highly accurate users such as in Fig. 3.1a. However, the situation is less favorable for the two-stage algorithm when the cost of finding the best user overwhelms the savings of queries due to increased accuracy as shown in Fig. 3.1b. It may even have an adverse effect when accuracies are similar, as shown in Fig. 3.1c.

When we increase the total number of users and keep their accuracy the same, as shown in Fig. 3.2, the `Ada-IIR` algorithm is able to tackle the increasing difficulty in finding more accurate users within a larger pool. Although, the overhead increases, our proposed method can adapt to each case and deliver near optimal performance.

In our experiments every algorithm is able to recover the exact rank with respect to the ground truth, which is reasonable since the `IIR` algorithm is designed to output an exact ranking. And due to the union bounds used to guarantee a high probability correct output, the algorithms tend to request more than enough queries so we did not see a case in which a non-exact ranking was produced.

3.6.2 Real-world Experiment

The above synthetic experiments serve as a proof of concept. We add one more experiment based on the real data, the setting is from the "Country Population" dataset from Jin et al. (2020). In this dataset the population of 15 countries were ranked by workers. Since the ground-truth Δ_u is not available, we first used the method described in the same work to infer the user accuracy and item parameters. During the simulation, the responses are generated according to their model with these parameters. As we have discussed in sample complexity analysis, the number of users should fall in a reasonable range. Thus, we randomly sub-sample a set of 25 users since the set of users provided by the dataset is excessive. The results, shown in Table 3.1, suggest that the `Ada-IIR` provides a moderate improvement over the non-adaptive algorithm.



(a) $M = 18$ (b) $M = 36$ (c) $M = 72$

Figure 3.2: Sample complexities v.s. number of items for all algorithms. (a) (b) and (c) are different settings where the number of users differs. The accuracy of two groups of users are $\gamma_A = 0.5$, $\gamma_B = 2.5$.

METHOD	SAMPLE COMPLEXITY
`IIR`	59223 ± 3183
`Two-stage`	85027 ± 2619
`Ada-IIR`	52693 ± 2739
`Oracle`	43855 ± 2365

Table 3.1: Experiments on Country Population with 15 items and 25 users.

Part II

Efficient Ranking under Weak Stochastic Transitivity Assumption

Chapter 4

Active Ranking under Weak Stochastic Transitivity

4.1 Introduction

To guarantee that the ranking is consistent with the preference probabilities, stochastic is often assumed, a formal discussion around SST and WST has been done in Section 1.4. However, SST can be too strong for scenarios where preference probabilities are not based on comparing a single quantifiable attribute. For instance, in sports, match outcomes are usually affected by team tactics. Team k may play a tactic that counters team i, resulting in a higher winning rate against team i compared with team j. Furthermore, items usually have multidimensional features and people may compare different pairs based on different features. A close pair in the overall ranking is thus not necessarily harder to compare than a pair that has a large gap. For example, when comparing cars, people might compare a given pair based on their interior design and another pair based on performance. As another example, in an experiment with games of chance with different probabilities of winning and payoffs (Tversky, 1969), it was observed that "people chose between adjacent gambles according to the payoff and between the more extreme gambles according to probability, or expected value." Motivated by such applications, in this chapter, we are interested in the problem of recovering the full ranking of n items under a more general setting, where only WST holds, while SST is not assumed to hold.

Existing algorithms (Mohajer et al., 2017; Ren et al., 2019) cannot avoid comparing every item i with the item i^* that is the most similar to i, i.e., $\left|p_{i,i^*} - \frac{1}{2}\right| = \min_{j \neq i}\{\left|p_{i,j} - \frac{1}{2}\right|\}$. Further, (Ren et al., 2019) pointed out that comparing item pairs that are adjacent in the true ranking are necessary. When SST holds, adjacent pairs are also the most difficult pairs to distinguish, existing methods thus achieve sample-efficiency. For example, the Iterative-Insertion-Ranking (`IIR`) algorithm proposed in (Ren et al., 2019) maintains a preference tree and performs ranking by inserting items one after another. During the insertion process, every item is possible to be compared with every other items (and thus the most similar one), depending on the relative order of insertion and the true ranking. `IIR` was shown to be sample complexity optimal under SST and some other conditions.

However, when SST does not hold, comparing nonadjacent items harms the performance. Consider an extreme scenario where the true ranking is $1 \succ 2 \succ 3$ and $p_{1,2} = p_{2,3} = 0.8, p_{1,3} = \frac{1}{2} + 2^{-10}$. If item 1 is directly compared to item 3, then it takes $\Theta\left(2^{20}\right)$ comparisons[1]. For instance, in `IIR`, this can happen during the insertion process of item 3 when item 1 happens to be the root of the preference tree. On the other hand, a simple fix exists as we can let the three pairs to be compared simultaneously. The comparisons between items 1 and 2, items 2 and 3 will terminate much earlier and provide us with the information $1 \succ 2$, $2 \succ 3$, which is enough to recover the total ranking. Therefore, it is important to devise an algorithm whose sample complexity will not be harmed when SST fails to hold.

We propose an active ranking algorithm, named `Probe-Rank` , that ranks n items based on pairwise

[1]In fact, according to (Farrell, 1964), we need $\Theta\left((p_{i,j} - 1/2)^{-2}\right)$ comparisons to be confident enough about the order between any two items i and j , $i, j \in [n]$.

comparisons. A comparison of related algorithms are presented in Table 4.1.

Table 4.1: δ-correct algorithms for exact ranking with sample complexity guarantee under WST assumption.

Algorithm	Sample complexity
Single Elimination Tournament (Mohajer et al., 2017)	$O\left(\frac{n(\log n)^2 \log(1/\delta)}{\min_{1\le i<j\le n} \Delta_{i,j}^2}\right)$
Iterative-Insertion-Ranking (IIR) (Ren et al., 2019)	$O\left(\sum_{i=1}^{n} \frac{1}{\Delta_i^2}\left(\log\log \frac{1}{\Delta_i} + \log \frac{n}{\delta}\right)\right)$
`Probe-Rank` (this work)	$O\left(n \sum_{i=1}^{n} \frac{1}{(\widetilde{\Delta}_i)^2}\left(\log\log \frac{1}{\widetilde{\Delta}_i} + \log \frac{n}{\delta}\right)\right)$

4.2 Problem Setup

Without loss of generality, let $[n] = \{1, 2, \ldots, n\}$ denote the set of n items. We write $p \sim Uni(a, b)$ to denote that p is sampled uniformly at random from the interval (a, b), and use $\text{Ber}(p)$ to denote a Bernoulli random variable which equals 1 with probability p. We assume that there exists a total ordering '$\succ$' over $[n]$ such that $\sigma_1 \succ \sigma_2 \succ \cdots \succ \sigma_n$ for some permutation σ of $[n]$. The permutation σ is referred to as the true ranking. Two items are called adjacent if they are adjacent in σ, i.e., one ranks right next to the other. To ensure that the true ranking σ is consistent with comparisons, we also assume that i has a higher rank than j if and only if $p_{i,j} > \frac{1}{2}$. In other words, if an item i is more preferred than j in σ, then i has a better chance to win the comparison with j. This assumption is known as *Weak Stochastic Transitivity (WST)*.

Intuitively, the closer $p_{i,j}$ is to $\frac{1}{2}$, the more difficult it becomes to obtain the ordering between i and j. Therefore, the probability gap $\Delta_{i,j}$, defined as $\Delta_{i,j} = \left|p_{i,j} - \frac{1}{2}\right|$, provides a characterization of the ranking task difficulty and will be used as a parameter for measuring sample complexities of algorithms. For instance, (Ren et al., 2019, lemma 12) shows that for any δ-correct algorithm $\mathcal{A}$, $\limsup_{\Delta\to 0} \frac{T_{\mathcal{A}}[\Delta]}{\Delta^{-2}(\log\log \Delta^{-1} + \log \delta^{-1})} > 0$, where $T_{\mathcal{A}}[\Delta]$ is the expected number of samples taken by $\mathcal{A}$ on two items with probability gap Δ. Further, for each item i, we define $\Delta_i = \min_{j:j\neq i} \Delta_{i,j}$, the minimum probability gap between item i and any other item j, and

$$\widetilde{\Delta}_i = \min_{j:j \text{ and } i \text{ are adjacent in } \sigma} \Delta_{i,j}, \tag{4.2.1}$$

the minimum probability gap between i and its adjacent items in the true ranking. Note that $\Delta_i \le \widetilde{\Delta}_i$ by definition and the equality holds when SST is satisfied.

4.3 Proposed Algorithm

In this section, we propose a δ-correct algorithm for exact ranking of all problem instances that satisfy the WST condition. Our algorithm is designed to outperform existing methods in situations where nonadjacent items can be more difficult to compare than adjacent items.

To avoid spending unnecessary samples on item pairs with small probability gaps, we propose a subroutine named *Successive-Comparison* (SC) (see Algorithm 4.1). SC uses a parameter τ for controlling to what extent the comparison should last. Specifically, SC compares a given item pair for a fixed number $b_\tau = \lceil (2/\epsilon_\tau^2) \log(1/\delta_\tau) \rceil$ times with an accuracy level $\epsilon_\tau = 2^{-\tau}$ and confidence level $\delta_\tau = 6\delta/(\tau^2\pi^2)$. If the empirical probability that i (respectively, j) wins is over $1/2$ by more than $\epsilon_\tau/2$, then SC returns i (respectively, j) as the more preferred item. Otherwise, SC will return 'unsure' to inform us that more samples are needed.

For two items i and j, $\text{SC}(i, j, \delta, \tau)$ will be called successively with τ increasing by 1 at a time. Later, we will show that after τ gets large enough such that $\epsilon_\tau \le \Delta_{i,j}$, the correct ordering between i and j will be returned with high probability.

Subroutine 4.1 Successive-Comparison(i, j, δ, τ) (SC)

Require: items i, j, confidence level δ, probing parameter τ

$w_i = 0, \epsilon_\tau = 2^{-\tau}, \delta_\tau = \frac{\delta}{c\tau^2}, c = \frac{\pi^2}{6}$, $b_\tau = \left\lceil \frac{2}{\epsilon_\tau^2} \log \frac{1}{\delta_\tau} \right\rceil$
for $t = 1$ to b_τ **do**
  compare i and j once; if i wins, $w_i = w_i + 1$
end for
$\widehat{p}_i = w_i / b_\tau$
if $\widehat{p}_i - \frac{1}{2} > \frac{1}{2}\epsilon_\tau$ **then**
  return $[i, j]$
else if $\widehat{p}_i - \frac{1}{2} < -\frac{1}{2}\epsilon_\tau$ **then**
  return $[j, i]$
else
  return 'unsure'
end if

Partial order preserving graph During the ranking process, we maintain a directed graph T to store the partial orders we have obtained from SC instances so far. The graph T is initialized with n nodes $V_1, \ldots, V_n$ and no edge exists between any two nodes. Nodes $V_1, V_2, \ldots, V_n$ represent items $1, 2, \ldots, n$, respectively. In our algorithm, T is involved with three types of operations, *edge update*, *node removal* and *maximal set selection*. Every time an instance of SC returns a pairwise order, e.g., $i \succ j$, we add a directed edge from V_i to V_j, written as $T = T \cup (i \succ j)$. Moreover, we also complete all edges in the transitive closure of the existing edges. In other words, if the edge between V_i and V_j induces a directed path from V_{k_1} to V_{k_2}, then a directed edge from V_{k_1} to V_{k_2} is also added to T. By completing the transitive closure, we can avoid comparing pairs whose ordering can be inferred from current knowledge and keep T acyclic. In the ranking process, we only run comparisons on item pairs that are not connected by edges and hence no contradictions in orderings will be returned by SC. By removing node V_i, we remove V_i and all edges of V_i from T. The maximal elements of T are the nodes which do not have any incoming edges. Since edges represent comparison results returned by SC, maximal elements correspond to items that have not lost to any other items. Note that since T is acyclic, maximal elements always exist.

Next, we establish our ranking algorithm ***Probe-Rank*** (see Algorithm 4.2). `Probe-Rank` finds the true ranking by performing maxing for $n - 1$ rounds. In every round t, subroutine *Probe-Max* returns an item in S_t as the most preferred item (the maximum), where S_t denotes the set of remaining unranked items right before round t. The strategy of Probe-Max is to repeatedly apply SC on all item pairs. For every item pair (i, j), we initialize a global variable $\tau_{i,j}$ as the probing parameter for SC instances that run over i, j. The graph T storing obtained partial orders is also viewed as a global variable. Parameters $\tau_{i,j}$ and graph T will be accessed and altered in Probe-Max.

Algorithm 4.2 `Probe-Rank`

Require: items $[n]$, confidence level δ

$S_1 = [n], Ans = [0]^n$, initialize T, $\tau_{i,j} = 1$ for all pairs of items $i \neq j$
for $t = 1$ to $n - 1$ **do**
  $i_{max} =$ Probe-Max$(S_t, 2\delta/n^2)$
  remove i_{max} from T; $Ans[t-1] = i_{max}$; $S_{t+1} = S_t \setminus \{i_{max}\}$
end for
$Ans[n-1] = S_n[0]$
return Ans

In Probe-Max(S, δ) (see Algorithm 4.3), SC instances are performed only on items that are possible to be the actual maximum. Let U be the set of maximal elements in T. By definition, every item in U has not lost to any other item in S yet. Assuming all previous comparison results (obtained form SC) are correct, to find the actual maximum, it suffices to focus on items in U. We use S^2 to denote the set of all unordered item pairs in S, i.e., $S^2 = \{(a, b) : a, b \in S, a \neq b\}$. All 'legitimate' pairs that can potentially provide us with

information about the maximum item in S are thus

$$P = \{(i,j) : (i \in U \text{ or } j \in U), (i,j) \in S^2, (i,j) \notin T\}, \tag{4.3.1}$$

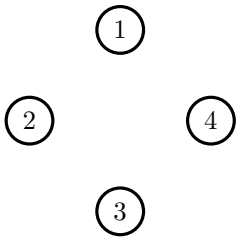
where $(i,j) \notin T$ means that nodes V_i and V_j are not connected in T. While U contains more than one items, Probe-Max keeps applying SC on item pairs in P. If an item in U loses a comparison, then we remove it from U. In every iteration of the while loop, the pairs (i^*, j^*) in P with the smallest τ value are chosen and $\text{SC}(i^*, j^*, \delta, \tau_{i^*,j^*})$ are performed. Note that the τ value increases by one after each call of SC. Starting with item pairs with small τ values guarantees that we do not miss any useful information that can be obtained by paying only a small amount of comparisons.

Subroutine 4.3 Probe-Max(S, δ)

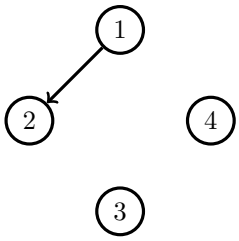
Require: set of unranked items S, SC confidence level δ
Let U be the set of maximal elements according to T
while $|U| > 1$ **do**
  Let $P = \{(i,j) : (i \in U \text{ or } j \in U), (i,j) \in S^2, (i,j) \notin T\}$
  for (a,b) in $\text{argmin}_{(x,y)\in P} \tau_{x,y}$ **do**
    $Ans = \text{SC}(a, b, \delta, \tau_{a,b})$; $\tau_{a,b} = \tau_{a,b} + 1$
    if Ans is not 'unsure' **then**
      $(w, l) = Ans$ {w is winner, l is loser}
      $T = T \cup (w \succ l)$
      if $|U| > 1$ and $l \in U$ **then**
        $U = U \setminus \{l\}$
      end if
    end if
  end for
end while
return $U[0]$

We provide a simple example demonstrating the ranking process.

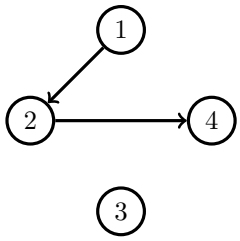
Example 4.3.1. Consider items $\{1, 2, 3, 4\}$ with true ranking $1 \succ 2 \succ 3 \succ 4$. Fig. 4.1 shows the status of T, U, S_t throughout the ranking process. In particular, we assume the pairwise comparison results are all correct and returned in order $1 \succ 2$, $2 \succ 4$, $1 \succ 3$, $2 \succ 3$, $3 \succ 4$.



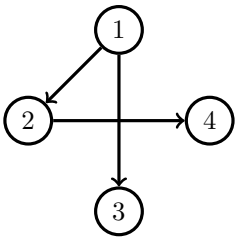
(a) ranking starts. $S_1 = \{1,2,3,4\}$, $U = S_1$.



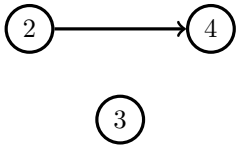
(b) $1 \succ 2$ returned. $U = \{1,3,4\}$.



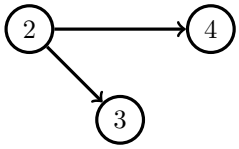
(c) $2 \succ 4$ returned. $U = \{1,3\}$.



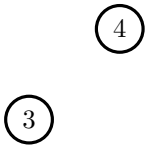
(d) $1 \succ 3$ returned. $U = \{1\}$.



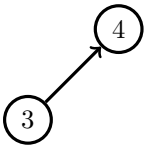
(e) 1 is the maximum, remove it. $S_2 = \{2,3,4\}$, $U = \{2,3\}$.



(f) $2 \succ 3$ returned. $U = \{2\}$.



(g) 2 is the maximum, remove it. $S_3 = \{3,4\}$, $U = \{3,4\}$.



(h) $3 \succ 4$ returned. $U = \{3\}$.

Figure 4.1: An illustration of the steps by Probe-Ranking, assuming true ranking as $1 \succ 2 \succ 3 \succ 4$.

4.3.1 A Sample-efficient Variant of `Probe-Rank`

In this section, we present a variant of `Probe-Rank`, named `Probe-Rank-SE` . When demonstrating more detailed experiments in Section 4.5.1, `Probe-Rank-SE` is also included and is shown to have better practical performance. However, we will not prove its correctness due to the high similarity it shares with `Probe-Rank`.

Compared with `Probe-Rank`, the variant `Probe-Rank-SE` finds the ranking also by performing $n-1$ steps of maxing and differs only in the subroutine for collecting comparison samples. Specifically, `Probe-Rank-SE` takes queries from all unknown item pairs simultaneously. Comparison results for pairs that terminate earlier are still collected and stored in the graph T, which represents our current knowledge about the ranking. We use T to decide whether to pause, drop or resume comparisons of remaining item pairs.

We adopt the Successive Elimination (SE) algorithm from (Even-Dar et al., 2002), shown in Algorithm 4.4, as a procedure to perform comparisons.

Subroutine 4.4 Successive Elimination (modified for comparing two items)

Require: items i, j, confidence level δ

$t = 1$
while true **do**
  Compare i and j for 2^t times; Let $\widehat{p}_i^t$ be the winning rate of i
  Let $\alpha_t = \sqrt{\frac{\log(ct^2/\delta)}{2^t}}$, $c = \frac{\pi^2}{3}$
  if $\widehat{p}_i^t - \frac{1}{2} > \alpha_t$ **then**
    return $i \succ j$
  else if $\widehat{p}_i^t - \frac{1}{2} < -\alpha_t$ **then**
    return $j \succ i$
  else
    $t = t + 1$
  end if
end while

It was shown that with probability at least $1-\delta$, Algorithm 4.4 correctly returns the more preferred item

between i and j using at most $O\left(\frac{1}{\Delta_{i,j}^2}\left(\log\frac{1}{\delta}+\log\log\frac{1}{\Delta_{i,j}}\right)\right)$ comparisons (Even-Dar et al., 2002, Remark 1).

In `Probe-Rank-SE` , we do not call SE directly, rather, SE is used as a black-boxed unit that repeatedly collects query samples from the input pair i, j. Moreover, after every sample, it generates feedback which is either Null, $i \succ j$ or $j \succ i$, where Null corresponds to that the number of samples has not accumulated to 2^t or $\left|\widehat{p}_i^t - \frac{1}{2}\right| < \alpha_t$; feedback $i \succ j$ and $j \succ i$ correspond to that inside the black box, SE actually terminates and returns the order between i and j. Note that the SE procedure can be replaced by any algorithm that can rank two items, including all best-arm-identification algorithms.

Denote the instance of Successive Elimination that runs over items i, j with confidence level δ as $\mathrm{SE}_{i,j}(\delta)$. When the value of δ is given without ambiguity, we will drop the dependence and write $\mathrm{SE}_{i,j}$ as a shorthand. We define two operations on $\mathrm{SE}_{i,j}$, named advance and feed. The advance operation returns one of the three possible internal outcomes, Null, $i \succ j$ or $j \succ i$. The feed operation is used for simulating the sampling process. We write feed $(\mathrm{SE}_{i,j}, Y_{i,j})$ to represent that $\mathrm{SE}_{i,j}$ is fed with a comparison sample $Y_{i,j}$. As a black-boxed unit, before advance returns one of $i \succ j$ and $j \succ i$, advance and feed operations are invoked in an alternating fashion. The idea of viewing a sampling subroutine as a black-box controlled by artificial operations was also used in (Ailon et al., 2014), but for a different problem setting.

`Probe-Rank-SE` is presented in Algorithm 4.5. We initialize $\binom{n}{2}$ independent instances of $\mathrm{SE}_{i,j}\left(2\delta/n^2\right)$, each for obtaining the order between an item pair $(i, j), 1 \le i < j \le n$. The probability of being unable to recover the true ranking is thus upper bounded by probability that at least one of the SE instances fails, which is at most δ. Same as `Probe-Rank`, we use T to denote the transitive closure composed of results returned by the SE instances.

Algorithm 4.5 `Probe-Rank-SE`

Require: items $[n]$, confidence level δ
1: $S_1 = [n], Ans = [0]^n$; initialize T
2: initialize $\mathrm{SE}_{i,j}\left(\frac{2\delta}{n^2}\right)$ for all $1 \le i < j \le n$
3: **for** t from 1 to $n-1$ **do**
4: $\quad i_{max}$ =Probe-Max-SE(S_t)
5: $\quad$ remove i_{max} from T; $Ans[t-1] = i_{max}$; $S_{t+1} = S_t \setminus \{i_{max}\}$
6: **end for**
7: $Ans[n-1] = S_n[0]$
8: **return** Ans

The procedure Probe-Max-SE serves as a switch for the SE instances. Let S_t^2 denote the set of unordered item pairs $\{(i, j) : i, j \in S_t, i \ne j\}$. In each round t, all SE instances for 'legitimate' pairs in S_t^2 are turned on and take queries in a round-robin fashion. 'Legitimate' pairs are similarly defined as in `Probe-Rank`. A pair (i, j) is 'legitimate' if the order between i, j is unknown, i.e., not in T, and at least one of i and j is a maximal element in S_t.

Algorithm 4.6 Probe-Max-SE(S_t)

Let U be sets of maximal elements according to T
while $|U| \geq 1$ **do**
  $C = [\,]$
  for (i, j) in S_t^2 **do**
    if $(i \in U$ or $j \in U)$ and $(i, j) \notin T$ **then**
      compare i with j once and get result $Y_{i,j}$; feed $\left(\mathrm{SE}_{i,j}\left(\frac{\delta}{n^2}\right), Y_{i,j}\right)$
      if advance $\left(\mathrm{SE}_{i,j}\left(\frac{2\delta}{n^2}\right)\right) == i \succ j$ **then**
        C.append($[i, j]$)
      else if advance $\left(\mathrm{SE}_{i,j}\left(\frac{2\delta}{n^2}\right)\right) == j \succ i$ **then**
        C.append($[j, i]$)
      end if
    end if
  end for
  for w, l in C **do**
    if $(w, l) \notin T$ **then**
      $T = T \cup (w \succ l)$
      if $|U| > 1$ and $l \in U$ **then**
        $U = U \setminus \{l\}$
      end if
    end if
  end for
end while
return $U[0]$

4.4 Theoretical Analysis

4.4.1 Upper Bound on the Sample Complexity

Theorem 4.4.1. Let $\delta > 0$ be an arbitrary constant. For all problem instances satisfying the Weak Stochastic Transitivity (WST) property, with probability at least $1 - \delta$, `Probe-Rank` returns the true ranking of n items and conducts at most

$$O\left(n \sum_{i=1}^{n} \left(\widetilde{\Delta}_i^{-2}\right)\left(\log\log\left(\widetilde{\Delta}_i^{-1}\right) + \log\left(\frac{n}{\delta}\right)\right)\right) \tag{4.4.1}$$

comparisons, where $\widetilde{\Delta}_i$ is defined as in (4.2.1).

We first show in the following lemma that the subroutine Successive-Comparison returns desired outcomes with high probability. Given an item pair (i, j) with probability gap $\Delta_{i,j} > 0$ and a positive integer τ, we say $\mathrm{SC}(i, j, \delta, \tau)$ is successful if one of the following two events holds,

$$\mathcal{E}_1 = \{\Delta_{i,j} \geq \epsilon_\tau \text{ and SC correctly returns } [i, j]\}, \tag{4.4.2}$$
$$\mathcal{E}_2 = \{\Delta_{i,j} < \epsilon_\tau \text{ and SC returns 'unsure' or } [i, j]\}. \tag{4.4.3}$$

Lemma 4.4.2. For an item pair (i, j) with probability gap $\Delta_{i,j} > 0$ and a positive integer τ, $\mathrm{SC}(i, j, \delta, \tau)$ is successful with probability at least $1 - \frac{\delta}{c\tau^2}$, where $c = \frac{\pi^2}{6}$.

Proof of Theorem 4.4.2. Hoeffding's inequality gives that

$$\Pr\left(\widehat{p}_i - p_{i,j} \leq -\frac{1}{2}\epsilon_\tau\right) \leq \exp\left(-2b_\tau\left(\frac{1}{2}\epsilon_\tau\right)^2\right) \leq \frac{\delta}{c\tau^2}. \tag{4.4.4}$$

Therefore, the probability that SC outputs $[j,i]$ is at most

$$\Pr\left(\widehat{p}_i - \frac{1}{2} < -\frac{1}{2}\epsilon_\tau\right) \le \Pr\left(\widehat{p}_i - p_{i,j} \le -\frac{1}{2}\epsilon_\tau\right) \le \frac{\delta}{c\tau^2}, \tag{4.4.5}$$

and the probability that SC returns $[i,j]$ or 'unsure' is at least $1 - \frac{\delta}{c\tau^2}$.

Further, if $\Delta_{i,j} \ge \epsilon_\tau$, the probability that SC returns $[i,j]$ is at least

$$\Pr\left(\widehat{p}_i - \frac{1}{2} > \frac{1}{2}\epsilon_\tau\right) = \Pr\left(\widehat{p}_i > \frac{1}{2} + \frac{1}{2}\epsilon_\tau\right) \ge \Pr\left(\widehat{p}_i > p_{i,j} - \frac{1}{2}\epsilon_\tau\right) \ge 1 - \frac{\delta}{c\tau^2}. \tag{4.4.6}$$

This completes the proof. □

By Theorem 4.4.2, with high probability, SC does not return the incorrect ordering. Further, if τ is large enough, then SC is guaranteed to return the correct ordering. We use Theorem 4.4.2 to show the theoretical performance of `Probe-Rank`.

Proof of Theorem 4.4.1. Define events

$$\mathcal{E}_{i,j}(\tau) = \{\text{SC}\left(i, j, 2\delta/n^2, \tau\right) \text{ is successful}\}. \tag{4.4.7}$$

Define the bad event

$$\mathcal{E}^{bad} = \cup_{(i,j)\in[n]^2} \cup_{\tau=1}^{\infty} \left(\mathcal{E}_{i,j}(\tau)\right)^c. \tag{4.4.8}$$

By the union bound and Theorem 4.4.2

$$\Pr\left(\mathcal{E}^{bad}\right) \le \sum_{(i,j)\in[n]^2} \sum_{\tau=1}^{\infty} \frac{2\delta}{cn^2\tau^2} \le \sum_{\tau=1}^{\infty} \frac{\delta}{c\tau^2} \le \delta. \tag{4.4.9}$$

In the following, we assume that $\mathcal{E}^{bad}$ does not happen.

Correctness. We show that when $\mathcal{E}^{bad}$ does not happen, in every round t, Probe-Max$(S_t, 2\delta/n^2)$ (Line 125 of Algorithm 4.2) correctly returns the most preferred item in the set of remaining items S_t. Since the probability of $\mathcal{E}^{bad}$ is upper bounded by δ, the correctness of `Probe-Rank` thus follows.

Let x be the most preferred item in S_t. When $\mathcal{E}^{bad}$ does not happen, all comparison results returned by SC are correct and T is always consistent with the true ranking. Thus, no item in S_t is known to rank higher than x, i.e., at the beginning of Algorithm 4.3, $x \in U$. Moreover, x will not be eliminated from U since x will not lose to any other item in S_t during calls of SC.

We show that any other item in U will be eliminated from U after a finite number of iterations of the while loop in Probe-Max. Let $y \ne x$ be an item in U. Since x is the maximum, $y \prec x$ in the true ranking. Whenever $\epsilon_{\tau_{y,x}} \le \Delta_{x,y}$, a successful call of SC $\left(x, y, 2\delta/n^2, \tau_{x,y}\right)$ will return the result $x \succ y$ and remove y from U if $\mathcal{E}^{bad}$ does not happen. Since $\epsilon_{\tau_{y,x}}$ converges to 0, there must exist $\tau^*_{x,y}$ such that $\epsilon_{\tau^*_{x,y}} \le \Delta_{x,y}$. After each execution of SC, the corresponding τ value increases by one, therefore after at most $\binom{n}{2}\tau^*_{x,y}$ iterations of the while loop, SC $\left(x, y, 2\delta/n^2, \tau^*_{x,y}\right)$ must have been called. The same argument holds for any $y \in U, y \ne x$.

Sample complexity. We first note the asymptotic behavior that for any $N > 0$,

$$\sum_{\tau=1}^{N} b_\tau \le \sum_{\tau=1}^{N} \frac{2}{4^{-\tau}} \log\frac{c\tau^2 n^2}{\delta} \le \sum_{\tau=1}^{N} \frac{2}{4^{-\tau}} \log\frac{cN^2 n^2}{\delta} = O\left(4^N \log\frac{cN^2\delta^2}{\delta}\right) = O\left(b_N\right). \tag{4.4.10}$$

Without loss of generality, we assume the true ranking is $1 \succ 2 \succ \cdots \succ n$. When $\mathcal{E}^{bad}$ does not happen, all comparison results returned by SC coincide with the true ranking. Therefore, for every $i \in [n-1]$, item i belongs to $S_1, S_2, \ldots, S_i$ and gets eliminated during the execution of Probe-Max$\left(S_i, 2\delta/n^2\right)$.

Recall that SC is only called over item pairs in which at least one of them is a maximal element. For every SC called on items a, b, if a is maximal, we say item a initializes the comparison and we charge the number of comparisons taken by SC to item a (if both a and b are maximal, we charge the number of samples to both).

Let $c(a)$ denote the number of comparisons charged to a. The total sample complexity of `Probe-Rank` is thus at most $\sum_{a\in[n]} c(a)$.

Fix $i \in [n]$. We use τ_i° to denote the value of $\tau_{i,i-1}$ when the order between i and $i-1$ is revealed. Define $\tau_1^\circ = 0$ for completeness. We note that the order between i and $i-1$ can not be inferred from any other comparison results therefore can only be returned by SC $\left(i, i-1, 2\delta/n^2, \tau_i^\circ\right)$. When $\mathcal{E}^{bad}$ does not happen, $\tau_i^\circ \le \left\lceil \log \frac{1}{\Delta_{i,i-1}} \right\rceil$ since a successful call of SC $\left(i, i-1, 2\delta/n^2, \left\lceil \log \frac{1}{\Delta_{i,i-1}} \right\rceil\right)$ will return the order.

For each $j \neq i$, we use $\tau_{i,j}^*$ to denote the value of $\tau_{i,j}$ when the last time SC is initialized by i and called over i, j before the beginning of Probe-Max$\left(S_i, 2\delta/n^2\right)$. In other words, for any $\tau > \tau_{i,j}^*$, if SC $\left(i, j, 2\delta/n^2, \tau\right)$ is called in Probe-Max$\left(S_t, 2\delta/n^2\right)$ for some $t < i$, then it must not be initialized by i. Moreover, we use $\tau_{i,j}^t$ to denote the value of $\tau_{i,j}$ right after Probe-Max$(S_t, 2\delta/n^2)$ terminates. Since i is ranked and removed from T after Probe-Max$(S_i, 2\delta/n^2)$ is called, $\tau_{i,j}^i$ is also the value of $\tau_{i,j}$ when `Probe-Rank` terminates. It is clear that

$$c(i) \le \sum_{j\neq i} \sum_{\tau=1}^{\tau_{i,j}^*} b_\tau + \sum_{j\neq i} \sum_{\tau=\tau_{i,j}^{i-1}+1}^{\tau_{i,j}^i} b_\tau. \tag{4.4.11}$$

We consider the first term on the right-hand side of (4.4.11). Before Probe-Max$\left(S_{i-1}, 2\delta/n^2\right)$ terminates, item $i-1$ is in T. Therefore, whenever i is a maximal element, the order between i and $i-1$ must have not been revealed. So when i initializes the comparison SC $\left(i, j, 2\delta/n^2, \tau_{i,j}^*\right)$, the item pair $(i, i-1)$ is also in the set of 'legitimate' pairs P. Therefore, $\tau_{i,j}^*$ is no larger than the value of $\tau_{i,i-1}$ at that point, and further no larger than τ_i°. The same argument holds for any j. It follows that

$$\sum_{j\neq i} \sum_{\tau=1}^{\tau_{i,j}^*} b_\tau \le \sum_{j\neq i} \sum_{\tau=1}^{\tau_{i,j}^*} b_\tau \le \sum_{\tau=1}^{\tau_i^\circ} n b_\tau. \tag{4.4.12}$$

Next, we bound the second term on the right-hand side of (4.4.11). Note that if there is no SC called during Probe-Max$(S_i, 2\delta/n^2)$, then $\sum_{j\neq i} \sum_{\tau=\tau_{i,j}^{i-1}+1}^{\tau_{i,j}^i} b_\tau = 0$. So it suffices to consider the case when at least one instance of SC is called during Probe-Max$(S_i, 2\delta/n^2)$. Consider the last group of SC called in Probe-Max$(S_i, 2\delta/n^2)$, here group means that there might be multiple item pairs whose τ values are the minimum in P. Denote their τ values by τ^i. There must be some SC $\left(a_i, b_i, 2\delta/n^2, \tau^i\right)$ returning $b_i \succ a_i$ such that a_i is a maximal item, otherwise no maximal item is removed from U and Probe-Max will not terminate. When $\mathcal{E}^{bad}$ does not happen, a_i is not the maximum in S_i so $a_i > i$. Thus, item $a_i - 1$ is also in S_i and before the call of SC $\left(a_i, b_i, 2\delta/n^2, \tau^i\right)$, the ordering between $a_i - 1$ and a_i is not revealed, i.e., $\tau^i \le \tau_{a_i}^\circ$. Moreover, $\tau_{i,j}^i \le \tau^i$ by the fact that we always compare item pairs with the smallest τ values. It follows that

$$\sum_{j\neq i} \sum_{\tau=\tau_{i,j}^{i-1}+1}^{\tau_{i,j}^i} b_\tau \le n \sum_{\tau=1}^{\tau^i} b_\tau = O\left(n b_{\tau^i}\right). \tag{4.4.13}$$

The same argument holds for all $i \in [n-1]$.

Consider the sets

$$\mathcal{D}_1 = \{b_{\tau^i} : i = 1, 2, \ldots, n-1\}, \quad \mathcal{D}_2 = \cup_{i=2}^n \mathcal{D}_2^i = \cup_{i=2}^n \{b_\tau : \tau = 1, 2, \ldots, \tau_i^\circ\}. \tag{4.4.14}$$

We claim that if $i_1 \neq i_2$, then the pairs (a_{i_1}, τ^{i_1}) and (a_{i_2}, τ^{i_2}) do not equal. With the facts that $a_i > i$ and $\tau^i \le \tau_{a_i}^\circ$, there is an injective mapping from $\mathcal{D}_1$ to $\mathcal{D}_2$ given by b_{τ^i} is mapped to the element b_{τ^i} in $\mathcal{D}_2^{a_i}$. It follows that

$$\sum_{i=1}^{n-1} O\left(n b_{\tau^i}\right) = O\left(\sum_{x\in\mathcal{D}_1} nx\right) \le O\left(\sum_{x\in\mathcal{D}_2} nx\right) = O\left(\sum_{i=2}^{n} \sum_{\tau=1}^{\tau_i^\circ} n b_\tau\right). \tag{4.4.15}$$

The reason for pairs (a_{i_1}, τ^{i_1}) and (a_{i_2}, τ^{i_2}) equal if and only if $i_1 = i_2$ is as follows. Let $i_2 > i_1$ and suppose $a_{i_1} = a_{i_2} = a$. When SC $\left(a, b_{i_1}, 2\delta/n^2, \tau^{i_1}\right)$ is called, SC $\left(a, b, 2\delta/n^2, \tau^{i_1}\right)$ for all b such that $(a, b) \notin T$ and

$\tau_{a,b} = \tau^{i_1}$ are also called. It follows that $\tau_{a,b} > \tau^i$ for all such b after this point. When SC $\left(a, b_{i_2}, 2\delta/n^2, \tau^{i_2}\right)$ is called, the order between a and b_{i_2} is not know and thus also not known when SC $\left(a, b_{i_1}, 2\delta/n^2, \tau^{i_1}\right)$ was called. So τ^{i_2} must be larger than τ^{i_1}.

Combining (4.4.11), (4.4.12) and (4.4.15) gives,

$$\sum_{i=1}^{n} c(i) \leq \sum_{i=2}^{n} \sum_{j \neq i} \sum_{\tau=1}^{\tau^*_{i,j}} b_\tau + \sum_{i=1}^{n-1} \sum_{j \neq i} \sum_{\tau=\tau^{i-1}_{i,j}+1}^{\tau^i_{i,j}} b_\tau \tag{4.4.16}$$

$$\leq O\left(\sum_{i=2}^{n} \sum_{\tau}^{\tau^\circ_i} n b_\tau\right) = O\left(n \sum_{i=2}^{n} b_{\tau^\circ_i}\right). \tag{4.4.17}$$

The desired sample complexity follows from $\tau^\circ_i \leq \left\lceil \log \frac{1}{\Delta_{i,i-1}} \right\rceil$ and

$$b_{\lceil \log \frac{1}{\Delta} \rceil} = O\left(\frac{1}{\Delta^2}\left(\log\log\frac{1}{\Delta} + \log\frac{n}{\delta}\right)\right), \tag{4.4.18}$$

which completes the proof. □

By the preceding theorem, the sample complexity of `Probe-Rank` is upper bounded by the sum of terms $(\widetilde{\Delta}_i)^{-2}(\log\log(\widetilde{\Delta}_i)^{-1} + \log(n/\delta))$ with an additional multiplicative factor of n. Recall from Section 1.2 that the term $(\widetilde{\Delta}_i)^{-2}(\log\log(\widetilde{\Delta}_i)^{-1} + \log(n/\delta))$ can be viewed as a lower bound on the number of comparisons that is needed for obtaining the order between i and its adjacent items with confidence level δ/n. Theorem 4.4.1 thus suggests that in `Probe-Rank`, every item is compared until it can be distinguished from its neighbors and no further. This matches with our intuition that only comparisons between adjacent items are necessary, and a single nonadjacent pair being extremely hard to distinguish should not harm the overall sample complexity. In contrast, sample complexities of existing algorithms are determined by the smallest probability gap between items, which can lead to a substantially large amount of unnecessary comparisons.

However, `Probe-Rank` achieves the dependence on $\widetilde{\Delta}_i$ instead of Δ_i at the cost of an additional multiplicative factor of n. Intuitively, because we have zero prior information about which items are adjacent and which are not, `Probe-Rank` pays $\Theta(n)$ attempts for each item i in order to 'identify' its neighbors and get the ordering feedback.

We compare `Probe-Rank` with the state-of-the-art `IIR` algorithm. Let $\mathcal{C}(\text{Probe})$ and $\mathcal{C}(\texttt{IIR})$ denote the sample complexities of two algorithms. From Table 4.1 and Theorem 4.4.1,

$$\mathcal{C}(\text{Probe}) = \sum_{i=1}^{n} \widetilde{\Theta}\left(n(\widetilde{\Delta}_i)^{-2}\right), \quad \mathcal{C}(\texttt{IIR}) = \sum_{i=1}^{n} \widetilde{\Theta}\left((\Delta_i)^{-2}\right), \tag{4.4.19}$$

noting that from the proofs, the sample complexity upper bounds are both tight in the worst case.

Under WST with no other conditions assumed, $\Delta_i \leq \widetilde{\Delta}_i$. In particular, when $\widetilde{\Delta}_i/\Delta_i = \Theta(\sqrt{n})$ for all i, then $\mathcal{C}(\text{Probe})$ and $\mathcal{C}(\texttt{IIR})$ are of the same asymptotic order with respect to n; if $\widetilde{\Delta}_i/\Delta_i = \omega(\sqrt{n})$, then `Probe-Rank` is asymptotically more sample-efficient than `IIR`. These phenomena are also reflected in our numerical experiments in Section 4.5 (see Fig. 4.3).

4.4.2 Lower Bound on the Sample Complexity

We first recall the common notion of (ϵ, δ)-correctness: with probability at least $1 - \delta$, the algorithm will output a ranking $\widehat{\sigma}$ such that for all $i \succ_{\widehat{\sigma}} j$, $p(i,j) \geq \frac{1}{2} - \epsilon$. Intuitively, this means the algorithm will only mis-rank those pairs satisfying $|p(i,j) - 1/2| < \epsilon$.

We construct a class of hard instances for the single-user setting. Each instance is indexed by a ranking σ.

Definition 4.4.3 ($\mathcal{I}_{\text{WST}}$)**.** Consider N items with an underlying ordering σ. For all $i \succ_\sigma j$,

$$p^\sigma(i,j) = \begin{cases} \frac{1}{2} + \epsilon, & \text{if } \sigma(i) = 1 \text{ and } \sigma(j) = 2, \\ \frac{1}{2}, & \text{otherwise,} \end{cases}$$

and for $i \prec_\sigma j$, $p(i,j) = 1 - p(j,i)$.

Solving the instance class above can be reduced to solving the one-sided instance class described below in Problem 4.4.4. The reduction is done by query (i,j) and (j,i) on $\mathcal{I}_{\mathrm{WST'}}$ equally likely to simulate the same environment as in $\mathcal{I}_{\mathrm{WST}}$. Therefore, $\mathcal{I}_{\mathrm{WST}}$ is at least as hard as $\mathcal{I}_{\mathrm{WST'}}$, up to constants.

Definition 4.4.4 ($\mathcal{I}_{\mathrm{WST'}}$)**.** Consider N items with an underlying ordering σ. For all i, j,

$$p^\sigma(i,j) = \begin{cases} \frac{1}{2} + 2\epsilon, & \text{if } \sigma(i) = 1 \text{ and } \sigma(j) = 2, \\ \frac{1}{2}, & \text{otherwise.} \end{cases}$$

For any $(2\epsilon, \delta)$-correct ranking algorithm that outputs a 2ϵ-correct ranking under $\mathcal{I}_{\mathrm{WST'}}$ with probability at least $1 - \delta$, we have that the algorithm must correctly rank between the largest item $\sigma^{-1}(1)$ and the second-largest one $\sigma^{-1}(2)$. Intuitively, this implies that the algorithm has to go over almost all pairs to correctly identify $\sigma^{-1}(1)$ and $\sigma^{-1}(2)$, which is signified by a biased coin among N^2 fair coins. We have the following result:

Theorem 4.4.5. For any (ϵ, δ)-correct algorithm $\mathcal{A}$, there exist a ranking σ and corresponding $p(i,j)$ such that with probability at least δ, $\sum_{i,j} C_{i,j} = \Omega\left(\frac{N^2 \log(1/\delta)}{\epsilon^2}\right)$, where $C_{i,j}$ denotes the queries made at (i,j).

The lower bound is tight up to logarithmic factors because a simple algorithm that allocates comparisons evenly to each pair will guarantee an ϵ-approximate estimation of $p(i,j)$, thus ensuring the ranking is (ϵ, δ)-correct.

Proof. Let $\mathcal{A}$ be an δ-correct algorithm. For any ranking σ, it correspond to a problem instance in $\mathcal{I}_{\mathrm{WST'}}$. We denote $\mathbb{P}^{\mathcal{A}}_{a,b}$ as the canonical bandit distribution of algorithm $\mathcal{A}$ under environment with $p(i,j) = \frac{1}{2} + \epsilon$ when $(i,j) = (a,b)$ and $p(i,j) = \frac{1}{2}$ otherwise. We also denote $\mathbb{P}^{\mathcal{A}}_0$ as the canonical bandit distribution of algorithm $\mathcal{A}$ under environment with $p(i,j) = \frac{1}{2}$ everywhere.

Since $\mathcal{A}$ is an δ-correct algorithm, its prediction on the ranking between a and b, denoted as $\widehat{\sigma}(a)$ and $\widehat{\sigma}(b)$, must align with the true ranking $\sigma(a) > \sigma(b)$ with probability at least $1 - \delta$:

$$\mathbb{P}^{\mathcal{A}}_{a,b}\big(\widehat{\sigma}(a) < \widehat{\sigma}(b)\big) \le \delta, \forall a, b \in [N], a \neq b.$$

Denote $X = \sum_{i,j} C_{i,j}$ the total number of queries made by $\mathcal{A}$ before it stops. Define the constant

$$\bar{x} := \inf\big\{x : \max_{a,b} \mathbb{P}^{\mathcal{A}}_{a,b}(X > x) \le \delta\big\}.$$

Here, $\bar{x}$ serves as a probabilistic lower bound of the total number of queries for all instances. This is the quantity we aim to bound from below in the coming reasoning.

Lemma 4.4.6. For the fixed $\bar{x}$, we have that

$$\mathbb{P}^{\mathcal{A}}_0(X > \bar{x}) \ge 1 - 2\delta.$$

Proof of Theorem 4.4.6. We define two new distributions $\widetilde{\mathbb{P}}^{\mathcal{A}}_{1,2}$ and $\widetilde{\mathbb{P}}^{\mathcal{A}}_{2,1}$, where $\widetilde{\mathbb{P}}^{\mathcal{A}}_{1,2}$ denotes the canonical bandit distribution of algorithm $\mathcal{A}$ under environment with $p(i,j) = \frac{1}{2} + \alpha$ when $(i,j) = (1,2)$ and $p(i,j) = \frac{1}{2}$ otherwise. $\widetilde{\mathbb{P}}^{\mathcal{A}}_{2,1}$ is defined similarly.

We have that

$$\mathbb{P}^{\mathcal{A}}_0(X \le \bar{x}) = \mathbb{P}^{\mathcal{A}}_0(\widehat{\sigma}(1) > \widehat{\sigma}(2), X \le \bar{x}) + \mathbb{P}^{\mathcal{A}}_0(\widehat{\sigma}(1) < \widehat{\sigma}(2), X \le \bar{x}),$$

and for $\mathbb{P}^{\mathcal{A}}_0(\widehat{\sigma}(1) > \widehat{\sigma}(2), X \le \bar{x})$, we have for any α,

$$\begin{aligned} \mathbb{P}^{\mathcal{A}}_0(\widehat{\sigma}(1) > \widehat{\sigma}(2), X \le \bar{x}) &\le \widetilde{\mathbb{P}}^{\mathcal{A}}_{2,1}(\widehat{\sigma}(1) > \widehat{\sigma}(2), X \le \bar{x}) \\ &\quad + \sup_{\mathcal{F} \cap \{w : X \le \bar{x}\}} |\mathbb{P}^{\mathcal{A}}_0(\mathcal{F} \cap \{w : X \le \bar{x}\}) - \widetilde{\mathbb{P}}^{\mathcal{A}}_{2,1}(\mathcal{F} \cap \{w : X \le \bar{x}\})| \\ &\le \delta + \underbrace{\sup_{\mathcal{F} \cap \{w : X \le \bar{x}\}} |\mathbb{P}^{\mathcal{A}}_0(\mathcal{F} \cap \{w : X \le \bar{x}\}) - \widetilde{\mathbb{P}}^{\mathcal{A}}_{2,1}(\mathcal{F} \cap \{w : X \le \bar{x}\})|}_{d_{\mathrm{TV}}(\mathbb{P}^{\mathcal{A}}_0, \widetilde{\mathbb{P}}^{\mathcal{A}}_{2,1} | X \le \bar{x})}, \end{aligned}$$

where the first inequality comes from the definition of total variance distance; the second inequality comes from $\mathcal{A}$ being δ-correct so that $\mathbb{P}_{2,1}^{\mathcal{A}}(\widehat{\sigma}(1) > \widehat{\sigma}(2)) \le \delta$. Let α converge to 0, we have that the total variance distance will also converge to 0 when $X \le \bar{x}$. Therefore, the above inequality implies that $\mathbb{P}_0^{\mathcal{A}}(\widehat{\sigma}(1) > \widehat{\sigma}(2), X \le \bar{x}) \le \delta$.

Applying the same argument to $\mathbb{P}_0^{\mathcal{A}}(\widehat{\sigma}(1) < \widehat{\sigma}(2), X \le \bar{x})$, we then conclude with $\mathbb{P}_0^{\mathcal{A}}(X > \bar{x}) \ge 1 - 2\delta$. □

Consider a new algorithm $\mathcal{A}'$ that performs exactly the same as $\mathcal{A}$, until $\mathcal{A}$ stops or its total number of queries reaches $\bar{x}$. In the latter case, $\mathcal{A}'$ will stop and return 'null'. We have that $\mathcal{A}'$ is an algorithm such that:

$$\mathbb{P}_{a,b}^{\mathcal{A}'}\big(\widehat{\sigma} = \text{'null'}\big) \le \delta, \forall a, b \in [N], a \ne b; \quad \mathbb{P}_0^{\mathcal{A}'}\big(\widehat{\sigma} \ne \text{'null'}\big) \le 2\delta,$$

where 2δ comes from two cases of failure: 1. outputting a wrong ranking as $\mathcal{A}$ with probability at most δ; 2. outputting 'null' when the queries exceed limit $\bar{x}$ with probability at most δ.

By the Bretagnolle–Huber inequality, we have $\exp\big(-d_{\mathrm{KL}}(\mathbb{P}_0^{\mathcal{A}'} \| \mathbb{P}_{a,b}^{\mathcal{A}'})\big) \le 6\delta$. Further, denoting we have

$$\begin{aligned}
&\exp\bigg(-\frac{1}{N(N-1)}\sum_{a,b}\sum_{i,j} C'_{i,j}\mathrm{KL}\big(p_0(i,j)\big\|p_{a,b}(i,j)\big)\bigg)\\
&\le \frac{1}{N(N-1)}\sum_{a,b}\exp\bigg(-\sum_{i,j} C'_{i,j}\mathrm{KL}\big(p_0(i,j)\big\|p_{a,b}(i,j)\big)\bigg)\\
&= \frac{1}{N(N-1)}\sum_{a,b}\exp\Big(-d_{\mathrm{KL}}(\mathbb{P}_0^{\mathcal{A}'}\|\mathbb{P}_{a,b}^{\mathcal{A}'})\Big)\\
&\le 6\delta,
\end{aligned}$$

where the first inequality comes from Jensen's inequality and $C'_{i,j}$ denotes the queries made by $\mathcal{A}'$ at (i,j); the first equation comes from the decomposition of KL-divergence for the canonical bandit model. $\mathrm{KL}(p\|q)$ denotes the KL-divergence between two Bernoulli random variables with expectation p and q. Note that for $(i,j) \ne (a,b)$, $p_0(i,j) = p_{a,b}(i,j) = 1/2$.

Rearranging the terms and remove those terms with $p_0(i,j) = p_{a,b}(i,j)$ gives

$$\sum_{a,b} C'_{a,b} \ge \frac{N(N-1)\log(1/(6\delta))}{\mathrm{KL}(1/2\|1/2+\epsilon)} = \Omega\bigg(\frac{N^2\log(1/\delta)}{\epsilon^2}\bigg).$$

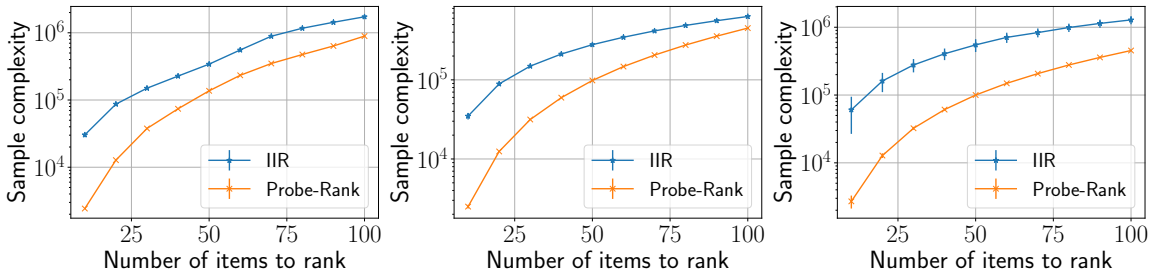
Notice that, $C'_{i,j}$ denotes the queries made by $\mathcal{A}'$ at (i,j), which satisfies that $\sum_{a,b} C'_{a,b} \le \bar{x}$, which as defined, serves as a high-probability lower bound on the sample complexity of $\mathcal{A}$. □

4.5 Experiments

In this section, we present numerical experiments demonstrating the practical performance of `Probe-Rank`. We compare `Probe-Rank` with the `IIR` algorithm, which was shown to outperform all the other baseline algorithms both theoretically and numerically (Ren et al., 2019).

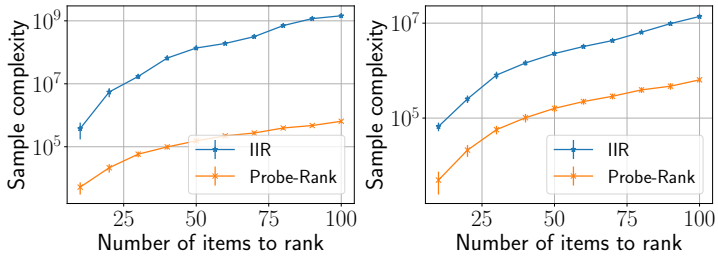
We study different settings where SST is satisfied, not guaranteed, or violated, but WST always holds, which is consistent with our theory. Specifically, we want to rank n items with the true ranking $\sigma_1 \succ \sigma_2 \succ \cdots \succ \sigma_n$, where n varies over $[10, 100]$. The probabilistic comparison model p_{ij} is generated in different ways to satisfy different assumptions. Note that Δ and Δ_d are tuning parameters in all the following settings.

- `SST`: SST is satisfied. Comparison probabilities p_{ij} are generated from the MNL model, where $p_{\sigma_i,\sigma_j} = (\exp(s_{\sigma_i} - s_{\sigma_j}) + 1)^{-1}$, and $s_{\sigma_1}, \ldots, s_{\sigma_n}$ is a decreasing sequence where $s_{\sigma_i} = 100\Delta_d \cdot \frac{(n+1-i)}{n}$.
- `WST`: SST does not necessarily hold. Let $p_{i,j} \sim Uni(\frac{1}{2} + \Delta_d, 1)$ for all items $i \succ j$.
- `NON-SST`: SST does not hold. For adjacent items, we have $p_{\sigma_i,\sigma_{i+1}} \sim Uni\left(\frac{1}{2} + \Delta_d, 1\right)$. Otherwise, we have $p_{\sigma_i,\sigma_j} \sim Uni\left(\frac{1}{2} + \frac{\Delta_d}{10}, \frac{1}{2} + \Delta_d\right)$ for $j > i+1$.



(a) SST: $\Delta_d = 0.3$ (b) WST: $\Delta_d = 0.3$ (c) NON-SST: $\Delta_d = 0.3$

Figure 4.2: Comparison of sample complexities of `Probe-Rank` and `IIR` under various settings. In each subfigure, Δ_d is fixed while the number of items varies.



(a) `ADJ-ASYM`: $\Delta_d = 0.3$, $\alpha = 1$ (b) `ADJ-ASYM`: $\Delta_d = 0.3, \alpha = 0.5$

Figure 4.3: Relationship between n and gap Δ_d

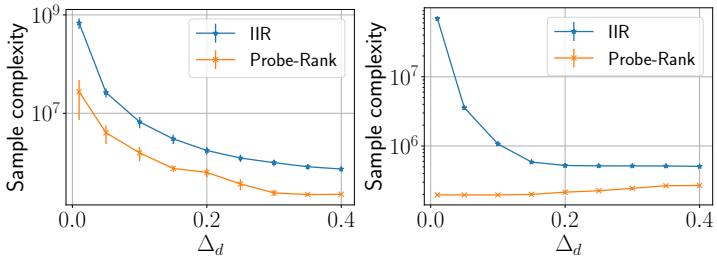
- `ADJ-ASYM`: SST does not hold. This setting is used to verify the asymptotic analysis in Section 4.4.1. For adjacent items, we set $p_{\sigma_i,\sigma_{i+1}} = \frac{1}{2} + \Delta_d$. Otherwise, we set $p_{\sigma_i,\sigma_j} = \frac{1}{2} + \frac{\Delta_d}{n^\alpha}$ for $j > i+1$. We consider cases where α equals 0.5 or 1.
- `ADJ-CNST`: SST does not hold. For adjacent items, we set $p_{\sigma_i,\sigma_{i+1}} = \frac{1}{2} + \Delta$. Otherwise $p_{\sigma_i,\sigma_j} = \frac{1}{2} + \Delta_d$ for $j > i+1$. Here $\Delta > \Delta_d$.

Two variants of the proposed algorithm and one baseline are compared. `IIR`: the baseline algorithm proposed in (Ren et al., 2019). `ProbeSort`: the proposed Algorithm 4.2 in previous section. `ProbeSortOpt` is an optimized version described in Section 4.3.1.

All experiments are averaged over 100 independent trials. For each trial, the ground truth ranking σ is generated uniformly at random and the comparison probabilities are assigned accordingly. The confidence level δ is fixed to be 0.1. Throughout the experiment, every trial for every algorithm successfully recovered the correct ranking.

We use internal clusters of intel "Skylake" generation CPUs. Each job contains a single model type for item numbers ranging from 10 to 100 with a step size of 10. Models are generated from a job unique random seed shared among the two algorithms. Most jobs with sample complexity smaller than 10^7 terminate in 3 minutes. For $\Delta_d = 0.1$ under the `ADJ-ASYM` model, 3 hours are needed due to high sample complexity. Due to the space limit, more detailed experimental setups and thorough ablation studies can be found in Section 4.5.1.

Performance comparison Fig. 4.2 with y-axis in log-scale shows comparison of `IIR` and `Probe-Rank`ing under the `SST`, `WST` and `NON-SST` settings. The parameter Δ_d is set to be 0.3. It can be seen that under the `SST` and `WST` settings (Figs. 4.2a and 4.2b), `Probe-Rank` consumes less samples than `IIR` for small n. As n gets larger, however, `IIR` becomes more sample-efficient due to that `Probe-Rank` has an additional factor of n in its sample complexity compared with `IIR` for instances satisfy SST. However, under the `NON-SST` setting where SST does not hold, `Probe-Rank` has a clear advantage over `IIR`, as shown in Fig. 4.2c.



(a) NON-SST: $n = 80$ (b) ADJ-CNST: $n = 80$, $\Delta = 0.4$

Figure 4.4: Ablation study on the dependence of the sample complexity on the probability gap Δ_d.

Dependence on n and the probability gaps Following Theorem 4.4.1, we verify that the sample complexity of `Probe-Rank` is lower than `IIR` when the number of items n gets larger. We use the `ADJ-ASYM` setting to simulate situations where nonadjacent items can be much more difficult to compare. In particular, we choose $\alpha = 1$ (see Fig. 4.3a) and $\alpha = 1/2$ (see Fig. 4.3b). It can be seen from Fig. 4.3a that as the number of items n gets larger, the gap between the two curves also gets larger. This matches our analysis that when $\widetilde{\Delta}_i/\Delta_i = \omega(\sqrt{n})$, then the sample complexity of `IIR` is of higher order than that of `Probe-Rank`. When $\widetilde{\Delta}_i/\Delta_i = \Theta(\sqrt{n})$, Fig. 4.3b shows that the gap between the two sample complexities varies little as n increases. Our analysis also suggests that sample complexities of two algorithms are of the same order.

Furthermore, we show through the `NON-SST` and `ADJ-CNST` settings that when the probability gaps of nonadjacent item pairs decrease, the advantage of our algorithm will be more and more prominent.

In Fig. 4.4, we fix $n = 80$ and let Δ_d vary. Clearly, `Probe-Rank` has an advantage over `IIR` in both settings. In particular, Fig. 4.4b shows the comparison of two algorithms in the `ADJ-CNST` setting with the probability gaps between adjacent items Δ fixed as 0.4. As the probability gap between nonadjacent items Δ_d varies from 0.01 to 0.4, it can be seen that the sample complexity of `Probe-Rank` does not vary much. However, the sample complexity of `IIR` has a positive correlation with $\frac{1}{\Delta_d^2}$. This numerical result matches our analysis that `Probe-Rank`ing is not affected by the comparison probability of nonadjacent items, which does not hold for `IIR`.

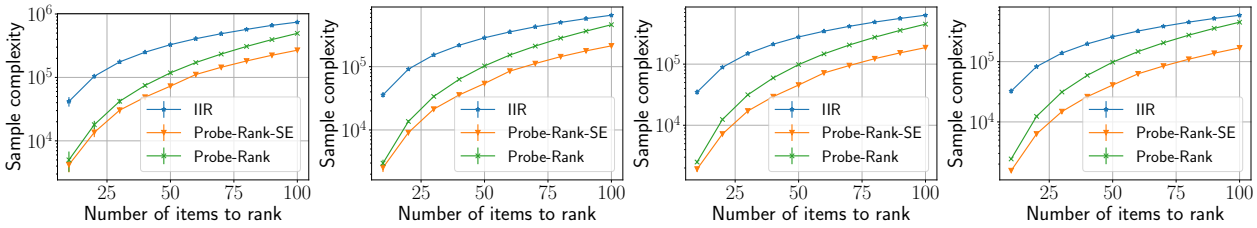
All trials are performed with confidence parameter $\delta = 0.1$. For the same setting, every algorithm is repeated 100 times. In each repeat, a ground truth ranking of n items is generated at random. Then the probabilities are generated from the ground truth ranking according to the method for each setting above. In each setting, the number of items to be ranked ranges from 10 to 100. With this confidence parameter, all algorithms are able to recover exactly the ground truth ranking. We use internal clusters of intel "Skylake" generation CPUs. Each job contains a single model type for item numbers ranging from 10 to 100 with a step size of 10. Models are generated from a job unique random seed shared among 3 algorithms. One job takes about 3-10 minutes depending on the difficulty parameter. And all jobs are repeated 100 times.

4.5.1 Detailed Experiments

In this section, we present more detailed numerical experiments comparing the sample complexities of `Probe-Rank`, `Probe-Rank-SE` and the state-of-the-art algorithm `IIR` by Ren et al. (2019). In particular, we focus on the `WST`, `SST`, `NON-SST` and `ADJ-ASYM` settings and perform these three algorithms with various parameters. Same as the results presented in Section 4.5, all experiments are averaged over 100 independent trials. For each trial, the ground truth ranking σ is generated uniformly at random and the comparison probabilities are assigned according to the chosen setting. The confidence level δ is fixed to be 0.1. Throughout the experiment, every trial for every algorithm successfully recovered the correct ranking. Moreover, for `IIR`, if the rank has not been recovered after the sample complexity reaches 10^9, we manually stop the ranking process and record the sample complexity as 10^9 to avoid extremely large running times. Note that the extreme cases happen in Figs. 4.8a to 4.8c and 4.12d.

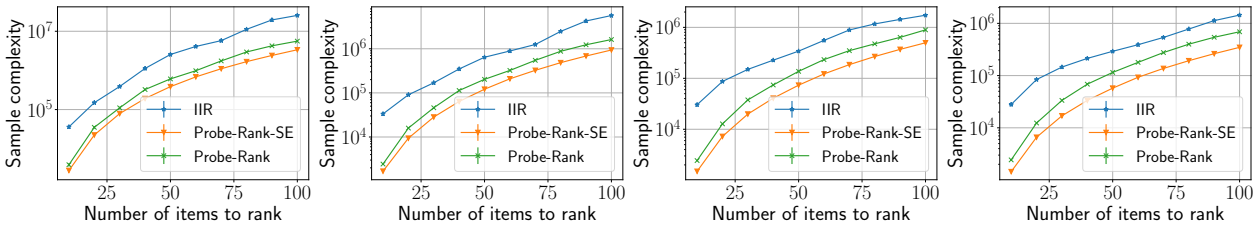
Figs. 4.5 to 4.8 compare the three algorithms under different settings where the difficulty parameter Δ_d is fixed and the number of items n varies from 10 to 100. Figs. 4.9 to 4.12 compare the three algorithms

under different settings where the number of items n is fixed and the difficulty parameter Δ_d varies from 0.1 to 0.4. It can be seen that `Probe-Rank` and its variant always consume less samples than `IIR` to recover the true ranking. Note that in the WST setting, comparison probabilities are all identically distributed and thus on average, adjacent items are as hard as nonadjacent items to compare. When Δ_d is fixed, as n gets larger and larger, `IIR` will eventually outperform `Probe-Rank`. This is consistent with our theoretical results presented in Section 4.4.1. Moreover, as indicated by the experimental results, `Probe-Rank-SE` can further reduce the sample complexity compared with `Probe-Rank`.



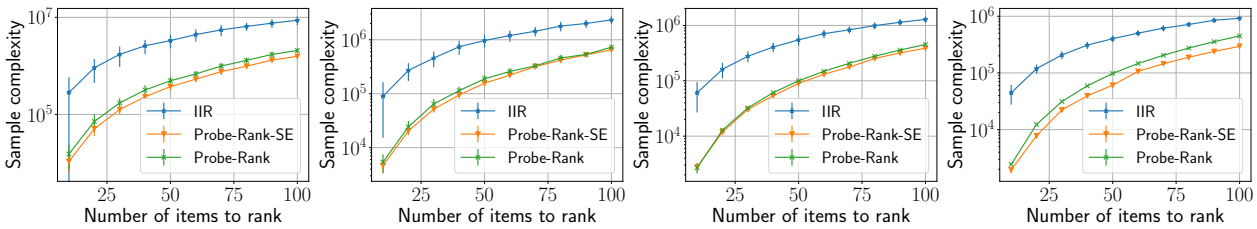
(a) WST: $\Delta_d = 0.1$ (b) WST: $\Delta_d = 0.2$ (c) WST: $\Delta_d = 0.3$ (d) WST: $\Delta_d = 0.4$

Figure 4.5: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the WST setting. In each subfigure, Δ_d is fixed while the number of items varies.



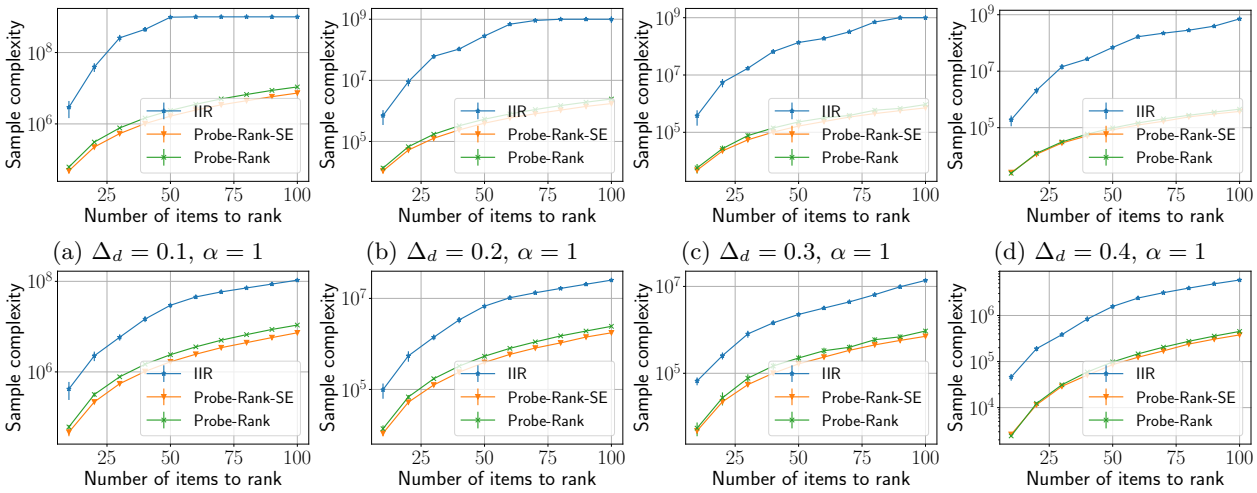
(a) SST: $\Delta_d = 0.1$ (b) SST: $\Delta_d = 0.2$ (c) SST: $\Delta_d = 0.3$ (d) SST: $\Delta_d = 0.4$

Figure 4.6: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the SST setting. In each subfigure, Δ_d is fixed while the number of items varies.



(a) NON-SST: $\Delta_d = 0.1$ (b) NON-SST: $\Delta_d = 0.2$ (c) NON-SST: $\Delta_d = 0.3$ (d) NON-SST: $\Delta_d = 0.4$

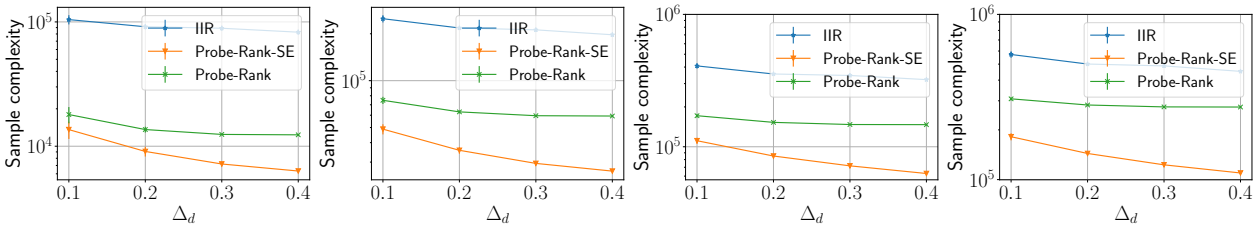
Figure 4.7: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the NON-SST setting. In each subfigure, Δ_d is fixed while the number of items varies.



(a) $\Delta_d = 0.1$, $\alpha = 1$ (b) $\Delta_d = 0.2$, $\alpha = 1$ (c) $\Delta_d = 0.3$, $\alpha = 1$ (d) $\Delta_d = 0.4$, $\alpha = 1$

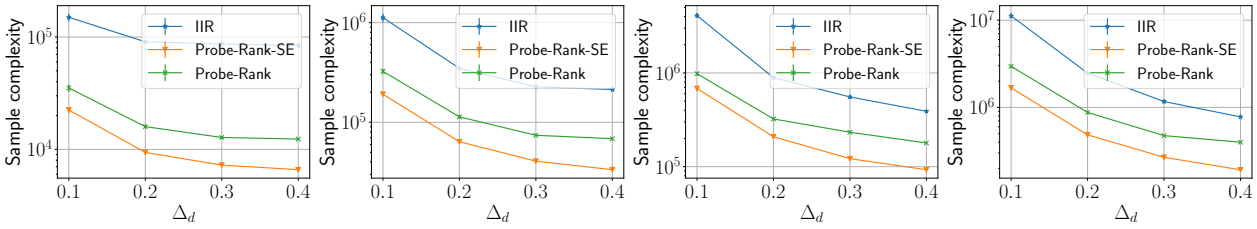
(e) $\Delta_d = 0.1$, $\alpha = 0.5$ (f) $\Delta_d = 0.2$, $\alpha = 0.5$ (g) $\Delta_d = 0.3$, $\alpha = 0.5$ (h) $\Delta_d = 0.4$, $\alpha = 0.5$

Figure 4.8: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the `ADJ-ASYM` setting. In each subfigure, Δ_d and α are fixed while the number of items varies.



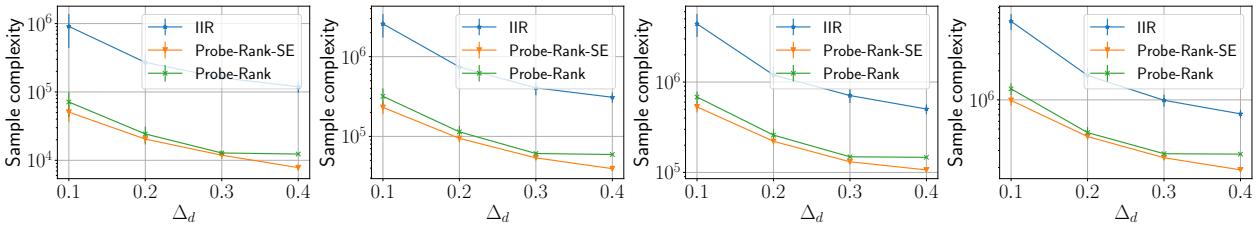
(a) WST, $n = 20$ (b) WST, $n = 40$ (c) WST, $n = 60$ (d) WST, $n = 80$

Figure 4.9: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the WST setting. In each subfigure, n is fixed while Δ_d varies.



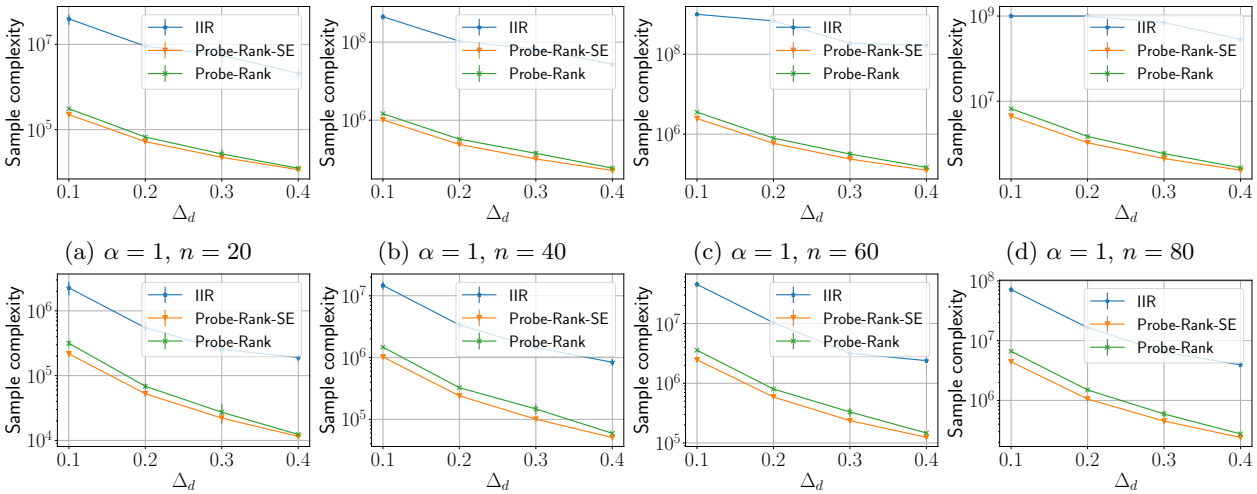
(a) SST, $n = 20$ (b) SST, $n = 40$ (c) SST, $n = 60$ (d) SST, $n = 80$

Figure 4.10: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the SST setting. In each subfigure, n is fixed while Δ_d varies.



(a) NON-SST, $n = 20$ (b) NON-SST, $n = 40$ (c) NON-SST, $n = 60$ (d) NON-SST, $n = 80$

Figure 4.11: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the NON-SST setting. In each subfigure, n is fixed while Δ_d varies.



(a) $\alpha = 1$, $n = 20$ (b) $\alpha = 1$, $n = 40$ (c) $\alpha = 1$, $n = 60$ (d) $\alpha = 1$, $n = 80$

(e) $\alpha = 1/2$, $n = 20$ (f) $\alpha = 1/2$, $n = 40$ (g) $\alpha = 1/2$, $n = 60$ (h) $\alpha = 1/2$, $n = 80$

Figure 4.12: Comparison of `Probe-Rank`, `Probe-Rank-SE` and `IIR` under the `ADJ-ASYM` setting. In each subfigure, n and α are fixed while Δ_d varies.

Chapter 5

Heterogeneous Active Ranking under Weak Stochastic Transitivity

5.1 Introduction

In many applications, the oracles[1] that provide preference feedback are usually human annotators, who may provide inherently noisy feedback. Moreover, oracles may show varying accuracy for different pairs of responses. For example, the performance gap between two LLMs may be negligible in one task but quite obvious in another. One natural question is then how to identify the underlying ranking of different LLMs efficiently, especially by taking advantage of those more accurate oracles (i.e., tasks). For example, when creating an LLM leaderboard, we have two different criteria or "tasks": honesty and helpfulness. When ranking two LLMs, we may ask a human annotator to rate which LLM is more honest and which LLM is more helpful. The two tasks/criteria may exhibit different preference behaviors.

Our goal is to accurately determine the ranking with as few queries as possible. Notably, Saad et al. (2023) solves the multi-oracle learning-to-rank tasks under a special case of the SST condition. In particular, they assume numerical feedback with sub-Gaussian noise, which satisfies the SST condition. This work established tight results for comparing two items, and directly swapped it with the deterministic comparison in a classic binary insertion sort algorithm to establish an upper bound for the noisy ranking problem.

Nevertheless, the application of SST can be overly restrictive in contexts where preference probabilities do not hinge on a single numerical attribute. Items can possess multifaceted attributes, leading people to evaluate different pairs based on different attributes. For example, LLM A is favored over LLM B because A produces longer responses, although both are equally informative. LLM B is preferred over LLM C because B's response is short and informative, while C's response is long and less informative. While human annotators can identify $A \succ B$ and $B \succ C$ easily, they may find it difficult to compare A and C for the long responses from both LLMs.

This phenomenon is pervasive in human behaviors and is defined as *Weak Stochastic Transitivity (WST)* introduced in Section 1.4. Therefore, a pair that is closely ranked might not always be more challenging to compare than one with a wider disparity. Driven by these real-world scenarios, this chapter mainly focuses on the challenge of identifying the complete ranking of N items in a broader context, where only WST is applicable and SST is not a requirement. Our primary goal is to minimize the number of comparisons while ensuring a high level of confidence.

5.2 Related work

Heterogeneous ranking and multi-task ranking. With the rise of crowdsourcing, data scientists are motivated to design algorithms adapted to this scenario to account for the variable quality of workers to achieve cost-efficient data acquisition rather than assigning tasks uniformly to workers regardless of their

[1] In this chapter, we refer to data sources as oracles

Table 5.1: A comparison among the related works and the proposed method. The table is divided into two major sections. The upper section mainly shows the sample complexities of three related algorithms under the *SST* condition. The lower section of the table shows the result under the *WST* condition.

Algorithm	Sample Complexity	Multi-Oracle
`IIR` (Ren et al., 2019)	$O\left(\sum_{i\in[N]} \Delta_i^{-2}\left(\log\log\left(\Delta_i^{-1}\right)+\log(N/\delta)\right)\right)$	No
`Binary-Search` (Saad et al., 2023)	$\widetilde{O}\left(\sum_{i\in[N]} H_i\left(\log\log(H_i)+\log^2(N)+\log(1/\delta)\right)\right)$	Yes
`Probe-Max` (Lou et al., 2022)	$\widetilde{O}\left(N\sum_{i=2}^{N}\Delta^{-2}_{\sigma^{-1}(i),\sigma^{-1}(i-1)}\right)$	No
`RMO-WST` (this work, Algorithm 5.1)	$\widetilde{O}\left(N\sum_{i=2}^{N} H_{\sigma^{-1}(i),\sigma^{-1}(i-1)}\right)$	Yes

performance (Niu et al., 2015). When data is already given and the algorithm is unable to affect the collection process, due to the varying precision of the workers, a model that considers or estimates the quality of the source while ranking shows a significant benefit over those that do not (Takanobu et al., 2019; Jin et al., 2020). In practical cases, the data collection has not happened yet; then an adaptive algorithm can be chosen to optimize query collection. Existing methods usually maintain two sets of estimates: one for ranking and one for worker quality (Wu et al., 2022; Saad et al., 2023). Low-accuracy workers are usually gradually eliminated, leaving high-quality responses to be collected more efficiently. A low-rank assumption can be made when the similarity between two parties within a subset of tasks can also be extrapolated to a broader set of tasks. Methods derived from probability matrix factorization are usually adopted in this case (Wang et al., 2016; Jun et al., 2019).

We summarize our contributions and compare them with the related work in Table 5.1. Due to the lengthy form of the exact result of `Binary-Search` and `RMO-SST` that does not fit in one line. We use $\widetilde{O}$ in the table to omit insignificant terms. In addition, we keep relevant log terms inside the $\widetilde{O}$ to showcase the improvement obtained by the proposed method.

5.3 Problem Setup and Preliminaries

In this chapter, we consider actively ranking N items, with M oracles (also known as data sources, users or experts). We assume there exists an ordering '$\succ$' over these items which is characterized as a mapping $\sigma(\cdot) : [N] \to [N]$ indicating the position of a given item in the ranking in descending order. Equivalently, the inverse mapping $\sigma^{-1}(\cdot)$ lists the items in order: $\sigma^{-1}(1) \succ \sigma^{-1}(2) \succ \cdots \succ \sigma^{-1}(N)$. For two items i and j, we assume that the result of the comparison is sampled independently from the Bernoulli distribution with mean $p^u_{i,j}$. More specifically, we denote $p^u_{i,j}$ as the probability that the response is "i is preferred over j" when a query is sent to oracle u. In particular, $p^u_{i,j} > \frac{1}{2}$ is considered as the item i is preferred over the item j by the oracle u. We will omit u in $p^u_{i,j}$ if the discussion is restricted to a single oracle.

For a fixed pair of items i and j, each oracle exhibits its preference, represented by the probability $p^u_{i,j}$. It is yet to decide how to aggregate them into a 'consensus preference'. If the consensus is defined as an average over $p^u_{i,j}$ or a majority vote over $\operatorname{sign}(p^u_{i,j} - 1/2)$, it is required that all oracles must be queried for each pair. In this case, there is no point to identify a more accurate expert to save queries. Instead, we make the following assumption that all oracles show a consistent preference for any item pair.

Assumption 5.3.1 (Consistency)**.** For any item pair (i, j), the preferences of all oracles are the same. More formally, for any two oracles u and v, we always have:

$$\operatorname{sign}(p^u_{i,j} - 1/2) = \operatorname{sign}(p^v_{i,j} - 1/2).$$

This assumption states that the oracles can show different levels of noise but must agree with the same underlying true ranking. This assumption also enables us to only select the more accurate oracles to recover the ranking with fewer comparisons. An equivalent assumption named the 'monotonicity' assumption is made by Saad et al. (2023).

5.3.1 Harndness Factor for Ranking two items

The hardness of estimating the preference of two items i and j under oracle u can be captured by the *gap* between their preferential probability and $1/2$: $\Delta_{i,j}^{(u)} = |p_{i,j}^u - 1/2|$. Intuitively, the closer the preferential probability to $1/2$, the harder it is to estimate the preference of the two items since the collected responses are more noisy. In our multi-oracle setting, a trivial method would be querying one oracle at a time in a uniformly random fashion and aggregating them as if from a single oracle, which leads to an *average gap* of

$$\bar{\Delta}_{i,j} := \sum_{u=1}^{M} \Delta_{i,j}^{(u)}/M. \tag{5.3.1}$$

In other words, any single-oracle algorithm can be trivially applied to the multi-oracle setting as if one oracle has a gap of $\bar{\Delta}_{i,j}$. It is easy to see a trivial solution is to construct an 'average' user by randomly sampling one user and query the pair (i, j). This will lead to a sample complexity of

$$\frac{\log(\delta^{-1})}{(\frac{1}{M}\sum_u \Delta_{i,j}^u)^2}.$$

5.4 Heterogeneous Ranking Algorithm under WST Condition

In this section, we propose an algorithm called *Rank-with-Multiple-Oracles (RMO)* under WST condition called `RMO-WST`, which is displayed in Algorithm 5.1 and has a bi-level design. At the high level, it calls `Probe-Max` (Algorithm 5.2) to select the maximal item from the pool of candidates repeatedly. At the low level, the `Compare` algorithm (Algorithm 5.3) perform comparisons that are necessary to rank a pair of items i and j, which also accounts for the heterogeneous quality of oracles that provide preferential feedback. `Try-Compare` (Algorithm 5.4) is where the actual comparison takes place and the order of pairs of items is determined.

In detail, `RMO-WST` (Algorithm 5.1, a variant of `Probe-Sort` in Lou et al. (2022)) takes a set of items labeled by $1, 2, \cdots, N$ as input and outputs a δ-correct ranking of them.

The set S_t contains the items to be ranked, each of which corresponds to a node in the directed graph T. It is also called "partial order preserving graph" in prior work (Lou et al., 2022). The graph starts with empty, where each node represents an item (Line 190). A directed edge is created between the two items, originating from the winning item, once the pair's order is determined. The maximal items are nodes of the current graph T such that there is no incoming edge towards them, which means they have not yet lose to any other items in comparison.

The WST assumption can also be employed to introduce additional edges during this process by getting the transitive closure of the graph. Furthermore, $\tau_{i,j}$ records a factor that determines the number of comparisons required to confidently determine the direction of a pair (i, j). It is initialized at 1 and its value will increase by one each time to determine the number of comparisons required throughout the algorithm. Inside each loop, the maximal item is found by `Probe-Max` with a confidence level of $2\delta/N^2$ and then removed from the graph T. This process repeats and finds the top items in the set of unranked items S_t sequentially.

Next, in `Probe-Max`, let U be the set that contains all the possible maximal items (i.e., all maximal items). Each item in U is paired with items whose order between them is not yet revealed (Line 2-Line 6). After revealing the order of a new pair using the `Compare` method, the losing item is removed from the set of possible maximal items U, and the graph T is also updated to include this directional edge and any other possible edges according to transitivity by running a standard method to compute the transitive closure of the graph (Line 9). After $N - 1$ rounds of finding the maximal item, the algorithm finds the ranking of the items. The sub-routine `Compare` (Algorithm 5.3, modified from Saad et al. (2023)), is designed to account for the multiple-oracles situation. The high-level idea is to enumerate different possible parameters: the subset size s_r and the gap width h_r. In Line 3-Line 10, the guessed subset size s_r and the gap width h_r will be sent to `Try-Compare` (Algorithm 5.4), until both reach the actual quantities. Then `Try-Compare` will return the correct comparison result with high probability (Line 8).

`Try-Compare` (Algorithm 5.4) is where the actual query and estimation for the pair direction takes place. In Line 2, a query size of m is determined by the subset size s given from the argument. Note that this value halves each time since s is doubled each time this subroutine is called on the same pair.

Then, a total number of $n_0 m$ comparisons is evenly assigned to the set of oracles that has not been eliminated yet (Line 6). Note that in our setting, the feedback is a binary indicator for a pair of items rather than a bounded scalar value for a single item as described in Saad et al. (2023). After comparison, a Bernoulli parameter estimate is calculated for each individual oracle as $\widehat{\mu}_{i,j}^{(u,\ell)}$ and a joint estimate as $\widehat{\mu}_{i,j}^{(\ell)}$ (Line 7). In Line 8-Line 9, the order of the pair is called when the confidence threshold is reached. If not, it continues to the elimination phase (Line 10), where oracles with accuracy lower than the medium of the group according to the estimate are removed from the active set S_ℓ^{ij} and S_ℓ^{ij}.

Algorithm 5.1 `RMO-WST` (N, δ): Rank-with-Multiple-Oracles

1: **input:** number of items to rank N, confidence level δ
2: **initialize**: $S_1 = [N], ans = [0]^N$, a directed graph T with N nodes and no edges, for $(i,j) \in S_1^2$ set $\tau_{i,j} = 1$
3: **define**: $\tau = \{\tau_{i,j}\}_{(i,j)\in[N]^2}$
4: **for** $t = 1$ to $N-1$ **do**
5: $\quad i_{\max}, T, \tau \leftarrow$ `Probe-Max`$(S_t, 2\delta/N^2, T, \tau)$
6: $\quad$ remove $i_{\max}$ from T
7: $\quad ans[t-1] = i_{\max}$
8: $\quad S_{t+1} = S_t \setminus \{i_{\max}\}$
9: **end for**
10: $ans[N-1] = S_N[0]$, **return** ans

Algorithm 5.2 `Probe-Max`(S, δ, T, τ)

1: **input**: set of unranked items S, confidence level δ, partial order preserving graph T, exponential factors $\{\tau_{i,j}\}_{(i,j)\in[N]^2}$ as τ.
2: Let U be the set of possible maximal items in T.
3: **while** $|U| > 1$ **do**
4: $\quad P = \{(i,j) | (i \in U \vee j \in U), (i,j) \in S^2, (i,j) \notin T\}$
5: $\quad$ **for** (i,j) in $\arg\min_{(x,y)\in P} \tau_{x,y}$ **do**
6: $\quad\quad ans =$ `Compare`$(i, j, \frac{6\delta}{\pi^2 \tau_{i,j}^2}, \tau_{i,j}), \tau_{i,j} = \tau_{i,j} + 1$
7: $\quad\quad$ **if** $ans \neq$ `unsure` **then**
8: $\quad\quad\quad w, l = ans, T = \text{TransClosure}(T \cup (w \succ l))$
9: $\quad\quad\quad$ **if** $|U| > 1$ and $l \in U$ **then** $U = U \setminus \{l\}$
10: $\quad\quad$ **end if**
11: $\quad$ **end for**
12: **end while**
13: return $U[0], T, \tau$

Algorithm 5.3 `Compare`(i, j, δ, τ)

1: **input**: pair (i,j), confidence level δ, precision factor τ
2: **initialize**: $r_{\max} = 1$, $\epsilon_\tau = 2^{-\tau}$, $ans =$ `unsure`.
3: **while** $ans =$ `unsure` and $\epsilon_\tau^2 < 4\log(2M)M2^{-r_{\max}}$ **do**
4: $\quad$ **for** $r = 0, \cdots, r_{\max}$ **do**
5: $\quad\quad s_r = \frac{2^r M}{2^{r_{\max}}}$, $h_r = 2^{-\frac{r}{2}}$
6: $\quad\quad \delta_{r_{\max}} = \delta/(10(r_{\max})^3 \log(M))$
7: $\quad\quad ans =$ `Try-Compare` $(i, j, \delta_{r_{\max}}, s_r, h_r)$
8: $\quad\quad$ if $ans \neq$ `unsure`, **break**.
9: $\quad$ **end for**
10: $\quad r_{\max} = r_{\max} + 1$
11: **end while**
12: return ans

Algorithm 5.4 `Try-Compare`(i, j, δ, s, h)

input: pair to query (i, j), confidence level δ, subset size s, estimated gap width h.
$m = 2^{\lceil \log_2(26 \log(1/\delta) M/s) \rceil}, n_0 = 64/h^2$.
Sample a set of m oracles with replacement from all M oracles as S_1.
Let $S_1^{ij} = S_1^{ji} = S_1$ and $L = \lceil \log_{4/3}(M/s) \rceil$.
for $\ell = 0, \cdots, L$ **do**
  Request $t_\ell = n_0 m / |S_\ell^{ij}|$ comparisons for pair (i, j) from each oracle $u \in S_\ell^{ij} \cup S_\ell^{ji}$. Denote c_u^{ij} as the number of times $i \succ j$.
  $\widehat{\mu}_{i,j}^{(u,\ell)} = \frac{c_{u,\ell}^{ij}}{t_\ell}$, $\widehat{\mu}_{j,i}^{(u,\ell)} = 1 - \widehat{\mu}_{i,j}^{(u,\ell)}$, $\widehat{\mu}_{i,j}^{(\ell)} = \frac{1}{|S_\ell^{ij}|} \sum_{u \in S_\ell^{ij}} \widehat{\mu}_{i,j}^{(u,\ell)}$.
  if $\widehat{\mu}_{i,j}^{(\ell)} - \frac{1}{2} \geq \sqrt{2 \log(2/\delta)/n_0 m}$ **then** return $i \succ j$
  if $\widehat{\mu}_{i,j}^{(\ell)} - \frac{1}{2} < -\sqrt{2 \log(2/\delta)/n_0 m}$ **then** return $i \prec j$
  $S_{\ell+1}^{ij} \leftarrow \{ v \in S_\ell^{ij} | \widehat{\mu}_{i,j}^{(v,\ell)} \geq$ medium of $\widehat{\mu}_{i,j}^{(u,\ell)}, u \in S_\ell^{ij} \}$
  $S_{\ell+1}^{ji} \leftarrow \{ v \in S_\ell^{ji} | \widehat{\mu}_{j,i}^{(v,\ell)} \geq$ medium of $\widehat{\mu}_{j,i}^{(u,\ell)}, u \in S_\ell^{ji} \}$
end for
return `unsure`.

5.5 Theoretical Analysis

5.5.1 Upper Bound of the Sample Complexity

Proof of Technical Lemmas

We first show that `Compare` (Algorithm 5.3) returns the correct outcomes between the given two items with high probability. In the following, without loss of generality, we assume the correct ordering between item i and item j is $i \succ j$. The proof follows those done similarly by (Saad et al., 2023), except that we deal with binary feedback representing the preference instead of numerical feedback for i and j respectively.

We first present the following lemma that characterizes the behavior of `Try-Compare` (Algorithm 5.4).

Lemma 5.5.1. In Algorithm 5.4, for each iteration ℓ, for any fixed pair of items $i \succ j$, we have that probability of getting incorrect probability estimation bounded as:

$$\mathbb{P}\left(\widehat{\mu}_{i,j}^{(\ell)} - \frac{1}{2} < -\sqrt{\frac{2 \log(1/\delta)}{|S_\ell| n_0}} \right) \leq \delta,$$

where S_ℓ is the subset of active oracles and n_0 is the number of repeated comparisons.

Proof. According to the assumption that $i \succ j$, so $\mathbb{E}[\widehat{\mu}_{i,j}^{(\ell)}] = \frac{1}{|S_\ell|} \sum_{u \in S_\ell} p_{i,j}^u \geq \frac{1}{2}$. Then we have

$$\mathbb{P}\left(\widehat{\mu}_{i,j}^{(\ell)} - \frac{1}{2} < -\sqrt{\frac{2 \log(1/\delta)}{|S_\ell| n_0}} \right) \leq \mathbb{P}\left(\widehat{\mu}_{i,j}^{(\ell)} - \mathbb{E}[\widehat{\mu}_{i,j}^{(\ell)}] < -\sqrt{\frac{2 \log(1/\delta)}{|S_\ell| n_0}} \right) \leq \delta.$$

The last inequality is due to Chernoff's inequality given that $\widehat{\mu}_{i,j}^{(\ell)}$ is a summation of $|S_\ell| n_0$ bounded independent random variables in $[0, 1]$. □

Then we have the following lemma stating the correctness of `Try-Compare` (Algorithm 5.4).

Lemma 5.5.2. In Algorithm 5.4, the probability of returning an incorrect result or order is bounded by:

$$\mathbb{P}(\text{return } i \prec j) \leq 1.75 \log(M) \delta.$$

Proof. If $i \prec j$ is returned, then in Algorithm 5.4, there exists some ℓ such that the condition in Line 9 is true. Formally, we have

$$\begin{aligned}\mathbb{P}(\text{return } i \prec j) &\leq \mathbb{P}\Big(\exists \ell \in [\log_{4/3}(M/m)] : \widehat{\mu}_{i,j}^{(\ell)} - 1/2 < -\sqrt{2\log(2/\delta)/n_0 m}\Big) \\ &\leq \log_{4/3}(M/m)\delta/2 \\ &\leq 1.75\log(M)\delta.\end{aligned}$$

The first inequality holds due to the reasoning above; the second inequality comes from the union bound and Theorem 5.5.1; in the last inequality we drop m and rearrange terms. □

Lemma 5.5.3. Assume $i \succ j$. In Algorithm 5.3, the wrong result will appear with probability $\mathbb{P}(\text{return } i \prec j) \leq 0.6\delta$.

Proof. Let $ans_{r_{\max},r}$ denote the output of Algorithm 5.4 when it is called with arguments $(\delta_{r_{\max}}, s_r, h_r)$. We have

$$\begin{aligned}\mathbb{P}(\text{return } i \prec j) &\leq \mathbb{P}\Big(\exists r_{\max} \geq 1, \exists r \leq r_{\max} : ans_{r_{\max},r} = (i \prec j)\Big) \\ &\leq \sum_{r_{\max}=1}^{\infty} \sum_{r=0}^{r_{\max}} \mathbb{P}(ans_{r_{\max},r} = (i \prec j)) \\ &\overset{(1)}{\leq} \sum_{r_{\max}=1}^{\infty} \sum_{r=0}^{r_{\max}} \frac{1.75\log(M)\delta}{10(r_{\max})^3\log(M)} \\ &= \sum_{r_{\max}=1}^{\infty} \frac{1.75\delta}{10} \frac{r_{\max}+1}{{r_{\max}}^3} \overset{(2)}{\leq} 0.6\delta,\end{aligned}$$

where (1) is due to Theorem 5.5.2 and the definition $\delta_{r_{\max}} = \delta/(10(r_{\max})^3\log(M))$, and (2) holds because $\sum_{r_{\max}=1}^{\infty} \frac{1+r_{\max}}{{r_{\max}}^3} \leq 3$. □

Proof of Subroutine: Theorem 5.5.4

To start with, we introduce the following theorem, which guarantees the performance of `Compare` (Algorithm 5.3).

Theorem 5.5.4. (Restatement of Theorem 4.1 from Saad et al. (2023)) For any given τ, δ and any pair (i, j), with probability at least $1 - \delta$, Algorithm 5.3 satisfies:

1. It outputs the correct order or `unsure` for any given τ and $\delta > 0$.
2. If $\tau > -\frac{1}{2}\log(M/H_{i,j})$, the correct order is returned.
3. When the correct order is returned, the sample complexity is $\widetilde{O}(\log(1/\delta)H_{i,j})$.

To obtain the above theorem, we replace the ϵ in the original theorem with $2^{-\tau}$ to suit our application. A detailed reasoning is available in **??**. This theorem guarantees the pairwise comparisons are correct with desired accuracy.

With a robust routine to return the correct order for each queried pair, a carefully designed ranking algorithm (Algorithm 5.1) orchestrates such pairwise comparisons to recover the ranking with high probability. In `RMO-WST`, the maximal item in the candidate set is identified and removed iteratively to rank all items. Thus, the total sample complexity is the summation of the sample complexity to identify each maximal item. And such cost can be upper bounded by N times the hardness to compare it with the item immediately smaller than it. We present the following theorem to characterize the total sample complexity upper bound of Algorithm 5.1. The detailed proof is deferred to **??**.

Proofs of Theorem 5.5.4. Our restatement also follows the structure of three conclusion claims towards the end of the proof by Saad et al. (2023, Theorem 4.1, Section B).

Note that one difference between our version of `Compare` (Algorithm 5.3) and their original algorithm is that their ϵ is replaced with τ by us to control the desired accuracy gap of one pairwise comparison. More specifically, in Line 2 of Algorithm 5.3, we calculated an equivalent ϵ as $\epsilon_\tau = 2^{-\tau}$ based on the input argument τ.

In their notation, d is the number of experts (oracles), which is equivalent to M in our notation.

The problem hardness factor $H_{i,j}$ is similarly defined. The setting is slightly different in that Saad et al. (2023) considered the difference of two 1-sub-Gaussian variables while we consider a Bernoulli variable shifted by 1/2. The central problem is to identify the sign of the expectation of the said random variables (which determines the order between i and j), and thus the problem hardness factors are defined in the same spirit.

Now, we are ready to restate the theorem.

Claim 1: The first part of the theorem can be directly derived from Theorem 5.5.3 or Saad et al. (2023, Lemma B.3). Indeed, since Theorem 5.5.3 states that Algorithm 5.3 will return the wrong result with probability at most 0.6δ. Therefore, Algorithm 5.3 returns either `unsure` or the correct order with probability at least $1-\delta$.

Claim 2: The second part of the theorem states that when τ is sufficiently large, the algorithm will return the correct result with a high probability of at least $1-\delta$.

To see this, when $\epsilon_\tau^2 = 2^{-2\tau} < M/H_{i,j}$, that is, when $\tau > -\frac{1}{2}\log(M/H_{i,j})$, the algorithm returns `unsure` with probability less than 0.4δ by the same argument as in Saad et al. (2023, Eq.5 and Lemma B.8). And by Theorem 5.5.3, it returns the wrong order with probability less than 0.6δ. In total, the probability of returning the incorrect result is less than δ.

Claim 3: The third part deals with the sample complexity. It is calculated under two conditions regarding the relationship between τ (hence ϵ_τ) and $M/H_{i,j}$, where c_1 is a constant. According to the end of the proof of Saad et al. (2023, Theorem 4.1, Section B) and replace it with our notation we have the following two cases:

1. When $\epsilon_\tau^2 \geq M/H_{i,j}$, Algorithm 5.3 finishes with a sample complexity of

$$c_1 \log^2(2M) \log\left(\frac{\log(2M)}{2^{-2\tau}}\right) \log\left(\frac{2\log(2M/2^{-2\tau})}{\delta}\right) \frac{M}{2^{-2\tau}} = \widetilde{O}\left(\log(1/\delta)\frac{M}{2^{-2\tau}}\right). \tag{5.5.1}$$

2. When $\epsilon_\tau^2 < M/H_{i,j}$, the total sample complexity is:

$$c_1 \log^2(M) \log(H_{i,j}) \log\left(\log(H_{i,j})\log(M)/\delta\right) H_{i,j} = \widetilde{O}(\log(1/\delta)H_{i,j}). \tag{5.5.2}$$

□

Proof of Main Result: Theorem 5.5.5

Theorem 5.5.5. (Instance-dependent sample complexity upper bound for `RMO-WST`) For a given set of items $[N]$ and desired confidence level δ, Algorithm 5.1 terminates with sample complexity bounded by

$$\widetilde{O}\Big(N\textstyle\sum_{i=2}^{N} H_{\sigma^{-1}(i),\sigma^{-1}(i-1)}\Big).$$

With probability at least $1-\delta$, Algorithm 5.1 will output a ranking that exactly matches the true ranking.

Proof of Theorem 5.5.5. As stated in Theorem 5.5.4, for any pair (i,j) and τ, with probability $1-\delta$, the following event will hold for `Compare`(i,j,δ,τ):

1. `Compare`(i,j,δ,τ) outputs the correct order or `unsure`.
2. If $\tau > -\frac{1}{2}\log(M/H_{i,j})$, `Compare`$(i,j,\delta,\tau)$ outputs the correct order.

3. The sample complexity is $\widetilde{O}(H_{i,j})$.

For each i, j and τ, denote $\mathcal{E}_{i,j}(\tau)$ as the high-probability event described above regarding $\texttt{Compare}(i, j, 6\delta/\pi^2\tau_{i,j}^2, \tau_{i,j})$ in Algorithm 5.4, which is called within $\texttt{Probe-Max}(S_t, 2\delta/N^2, T, \tau)$ in Algorithm 5.1. Then, we have that

$$\mathbb{P}(\mathcal{E}_{i,j}(\tau)) \geq 1 - \frac{12\delta}{\pi^2 N^2 \tau_{i,j}^2}.$$

By union bound, the probability that there exists one pair (i, j) that is compared wrongly by $\texttt{Compare}(i, j, \delta/\tau_{i,j}^2, \tau_{i,j})$ for some $\tau_{i,j}$ is

$$\mathbb{P}\left(\bigcup_{(i,j)\in[N]^2}\bigcup_{\tau=1}^{\infty}\overline{\mathcal{E}_{i,j}(\tau)}\right) \leq \frac{N^2}{2}\sum_{\tau=1}^{\infty}\frac{12\delta}{\pi^2 N^2\tau^2} \leq \delta, \tag{5.5.3}$$

where the last inequality comes from $\sum_{\tau=1}^{\infty}\tau^{-2} = \pi^2/6$.

In the following proof, we assume that $\texttt{Compare}(i, j, 6\delta/\pi^2\tau_{i,j}^2, \tau_{i,j})$ always runs successfully. Now, using Eq. (5.5.1), Eq. (5.5.2), define the following two terms:

1. When $\epsilon_\tau^2 \geq M/H_{i,j}$, for any $t > 0$,

$$n^{(t)} := c_1 \log^2(2M)\log\left(\frac{\log(2M)}{4^{-t}}\right)\log\left(\frac{2\pi^2N^2t^2\log(M/4^{-t})}{12\delta}\right)\frac{M}{4^{-t}} = \widetilde{O}(4^t M), \tag{5.5.4}$$

2. When $\epsilon_\tau^2 < M/H_{i,j}$, that is $2^{-\tau_{i,j}} < M/H_{i,j}$, which is applied in the first inequality below:

$$n_{i,j}^{(*)} := c_1\log^2(M)\log(H_{i,j})\log\left(\frac{\tau_{i,j}^2\log(H_{i,j})\log(M)N^2\pi^2}{12\delta}\right)H_{i,j} \leq \widetilde{O}(H_{i,j}). \tag{5.5.5}$$

Given the fact that `RMO-WST` only compares the pair contains at least one maximal element. In this case, for every call of `Compare` on pair (i, j) if i is maximal, we say that item i *initializes* the comparison, and the number of comparisons is charged to i. If both i, j are maximal, then the cost is charged to both items. We denote the total number of charged comparisons to i as $c(i), i \in [N]$. And the sample complexity of `RMO-WST` is at most $\sum_{i\in[N]} c(i)$.

Without loss of generality, assume the true ranking of items is $1 \succ 2 \succ \cdots \succ N$. Given $i \in [N]$, we use τ_i° to denote the value of $\tau_{i,i-1}$ when the order between i and $i-1$ is revealed. Let $\tau_1^\circ = 0$. The order of adjacent items of i under WST condition can only be revealed when $\texttt{Compare}(i, i-1, 2\delta/N^2, \tau_i^\circ)$ returns a value other than `unsure`. According to Theorem 5.5.3, $\tau_i^\circ \leq \lceil \frac{1}{2}\log\frac{H_{i,i-1}}{M}\rceil$.

Define $b_{i,j}^{(\tau)}$ as follows, where $n^{(\tau)}$ is defined in Eq. (5.5.4):

$$b_{i,j}^{(\tau)} = \begin{cases} n^{(\tau)}, & \text{if } \tau < \tau_{i,j}^\circ \\ \sum_{t=1}^{\tau-1} n^{(t)} + n_{i,j}^{(*)}, & \text{otherwise} \end{cases}$$

For each $j \neq i$, let $\tau_{i,j}^*$ be the value of $\tau_{i,j}$ when last time `Compare` is initialized by i and called before $\texttt{Probe-Max}(S_i, 2\delta/N^2)$. For any $\tau > \tau_{i,j}^*$, if $\texttt{Compare}(i, j, 2\delta/N^2, \tau)$ is called in $\texttt{Probe-Max}(S_t, 2\delta/N^2)$ for some $t < i$, then it must not be initialized by i. In light of this, let $\tau_{i,j}^t$ be the value of $\tau_{i,j}$ after completion of $\texttt{Probe-Max}(S_t, 2\delta/N^2)$. We break down $c(i)$ into two parts as follows:

$$c(i) \leq \sum_{j\neq i}\sum_{\tau=1}^{\tau_{i,j}^*} b_{i,j}^{(\tau)} + \sum_{j\neq i}\sum_{\tau=\tau_{i,j}^{i-1}+1}^{\tau_{i,j}^i} b_{i,j}^{(\tau)} \tag{5.5.6}$$

We now move on to bound the first summation term on the right-hand side of Eq. (5.5.6). Before $\texttt{Probe-Max}(S_{i-1}, 2\delta/N^2)$ terminates, item $i-1$ is in T. Therefore, whenever i is a maximal item, the order between i and $i-1$ is not revealed. So when i initializes the comparison $\texttt{Compare}(i, j, 2\delta/N^2, \tau_{i,j}^*)$, the item

pair $(i, i-1)$ is also in the set of legitimate pairs P. Therefore, $\tau^*_{i,j}$ is no larger than the value of $\tau_{i,i-1}$ at that point, and is further no larget than τ°_i:

$$\sum_{j\neq i}\sum_{\tau=1}^{\tau^*_{i,j}} b^{(\tau)}_{i,j} \leq N \sum_{\tau=1}^{\tau_{i,i-1}} b_\tau \leq N \sum_{\tau=1}^{\tau^\circ_i} b^{(\tau)}_{i,i-1}. \tag{5.5.7}$$

We then continue to bound the second summation term in $c(i)$ in Eq. (5.5.6). Consider the last group of `Compare` called in `Probe-Max`$(S_i, 2\delta/N^2)$, here the groups mean that there might be multiple item pairs whose values τ are the minimum in P. Denote their τ values by τ^i. There must be some `Compare`$(a_i, b_i, 2\delta/N^2, \tau^i)$ returning $b_i \succ a_i$ such that a_i is a maximal item, otherwise no maximal item is removed from U and `Probe-Max` will not terminate. When every `Compare` call is returning the correct order, a_i is not the maximal in S_i so $a_i > i$. Thus, item $a_i - 1$ is also in S_i and before the call of `Compare`$(a_i, b_i, 2\delta/N^2, \tau^i)$, the order between a_i and $a_i - 1$ is not revealed, that is, $\tau^i \leq \tau^\circ_{a_i}$. Moreover, $\tau^i_{i,j} \leq \tau^i$ because we always compare pairs of items with the smallest τ values, it follows that

$$\sum_{j\neq i}\sum_{\tau=\tau^{i-1}_{i,j}+1}^{\tau^i_{i,j}} b^{(\tau)}_{i,j} \leq N \sum_{\tau=1}^{\tau^\circ_i} b^{(\tau)}_{i,i-1}. \tag{5.5.8}$$

In summary, the total sample complexity is

$$\sum_{i=1}^{N} c(i) \leq 2N \sum_{i=1}^{N}\sum_{\tau=1}^{\tau^\circ_i} b^{(\tau)}_{i,i-1} = 2N \sum_{i=2}^{N}\sum_{\tau=1}^{\tau^\circ_i} b^{(\tau)}_{i,i-1}, \tag{5.5.9}$$

where the last equality is due to $\tau^\circ_1 = 0$. Plug in $\tau^\circ_i \leq \lceil \frac{1}{2}\log\frac{H_{i,i-1}}{M} \rceil$ into the above equation to get:

$$2N \sum_{i=2}^{N}\sum_{\tau=1}^{\tau^\circ_i} b^{(\tau)}_{i,j} = N\Bigg[\sum_{i=2}^{N} O\left(\log^2(M)\log(H_{i,i-1})\log\left(\log(H_{i,i-1})\log(M)\right)H_{i,i-1}\right) \tag{5.5.10}$$

$$+ \sum_{i=2}^{N} \log^2(2M)\log\left(\log(2M)\frac{H_{i,i-1}}{M}\right)\log\left(4N^2\log(\frac{H_{i,i-1}}{M}M)/\delta\right) O(\frac{H_{i,i-1}}{M}M)\Bigg] \tag{5.5.11}$$

$$= \widetilde{O}\left(N\sum_{i=2}^{N} H_{i,i-1}\right) \tag{5.5.12}$$

Given we assumed w.l.o.g. that the correct ranking is $1 \succ 2 \succ 3 \succ \cdots N$ and the sample complexity is Eq. (5.5.12). Now we conclude without this assumption the sample complexity would be

$$\widetilde{O}\left(N\sum_{i=2}^{N} H_{\sigma^{-1}(i),\sigma^{-1}(i-1)}\right).$$

□

Remark 5.5.6. A baseline algorithm is to uniformly randomly choose one oracle and apply the `Probe-Rank` algorithm with the average oracle. This leads to an averaged oracle with average gap $\bar{\Delta}_{i,j}$ or average hardness $\bar{H}_{i,j}$ as described near the end of **??**. The resulting sample complexity is $\widetilde{O}\big(N\sum_{i=2}^{N}\bar{H}_{\sigma^{-1}(i),\sigma^{-1}(i-1)}\big)$, which is strictly worse than our sample complexity, since $H_{i,j} \leq \bar{H}_{i,j}$.

5.5.2 Lower Bound of the Sample Complexity for Multiple Oracles

We define the following multi-oracle problem class:

Definition 5.5.7 ($\mathcal{I}_{\text{WST"}}$)**.** Consider N items with an underlying ordering 'σ'. For any items i, j and oracle u,

$$p_u^\sigma(i,j) = \begin{cases} \frac{1}{2} + \frac{\epsilon}{\sqrt{M}}, & \text{if } \sigma(i) = 1 \text{ and } \sigma(j) = 2, \\ \frac{1}{2}, & \text{otherwise.} \end{cases}$$

For any (ϵ, δ)-correct ranking algorithm that outputs a ϵ-correct ranking under $\mathcal{I}_{\text{WST"}}$ with probability at least $1 - \delta$, we have that the algorithm must correctly rank between the largest item $\sigma^{-1}(1)$ and the second-largest one $\sigma^{-1}(2)$.

Because all oracles have the same comparison probability, the problem is equivalent to ranking with a single oracle, with the lower bound being $\Omega\Big(\frac{N^2 M \log(1/\delta)}{\epsilon^2}\Big)$.

Further, we define the (ϵ, δ)-correctness for multiple oracles:

Definition 5.5.8. An algorithm $\mathcal{A}$ is called (ϵ, δ)-correct, if with probability at least $1 - \delta$, $\mathcal{A}$ will output a ranking $\widehat{\sigma}$ such that for all $i \succ_{\widehat{\sigma}} j$ but $j \succ_\sigma i$, $\sum_{u=1}^M (\Delta_{i,j}^u)^2 < \epsilon^2$.

Intuitively, the equivalent probability margin $\sqrt{\sum_{u=1}^M (\Delta_{i,j}^u)^2}$ must be small for any mis-ranked pair (i, j).

We have the following result:

Theorem 5.5.9. For any (ϵ, δ)-correct algorithm $\mathcal{A}$, there exist a ranking σ and corresponding $\{p^u(i,j)\}_{u \in [M]}$ such that with probability at least δ,

$$\textstyle\sum_{i,j} C_{i,j} = \Omega\big(N^2 M \log(1/\delta)/\epsilon^2\big) = \widetilde{\Omega}\big(N^2 H_{\sigma^{-1}(1),\sigma^{-1}(2)}\big),$$

where $C_{i,j}$ denotes the queries made at (i, j).

Again, the lower bound is tight and can be reached by allocating the comparison budget evenly to each pair and call Algorithm 5.3.

5.6 Experiments

5.6.1 Improved Algorithm for Practical Use

In this section, we present the algorithm that is modified for practical usage. Instead of using an estimator $\widehat{\mu}_{i,j}$ that only depends on the data collected in the same iteration of the for loop. A global estimator is derived from the statistics collected from multiple iterations to save sample complexity (Line 6).

We study the practical performance of the proposed algorithm and compare it with existing methods. We compare two methods in the experiment:

`Probe-Max`: the main algorithm proposed (Lou et al., 2022), however, their algorithm does not account for multiple oracles. In this case, as a naive implementation, whenever a pair is requested, it chooses an oracle from $[M]$ uniformly at random.

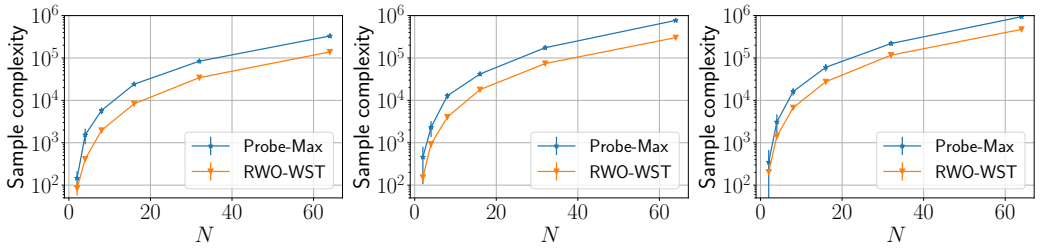
`RMO-WST`: Algorithm 5.1 proposed in this work. However, we notice that due to multiple uses of the union bound, excessive sampling can occur, which is unrealistic in real-world scenarios.

For instance, repetitive sampling of m tasks as seen in Algorithm 5.4 at Line 2 can enhance the precision of the overall estimate $\widehat{\mu}_{i,j}^{(u,\ell)}$. However, since s diminishes at an exponential rate, the quantities m and t_l also increase exponentially. Our hypothesis is that setting $S_1 = [M]$—in other words, maintaining a constant size of m at M—can lead to greater efficiency. Additionally, the distribution of the accuracy of active candidates is the same with or without the sampling without replacement in Line 6. Furthermore, note that the confidence interval used from Line 8 to Line 9 still holds after the change. We also notice that, in Line 7 the individual estimate and the global estimate depend only on the samples collected within a single iteration of the for loop starting Line 4. This can also be improved by reusing the statistics collected in previous rounds. We present the improved algorithm in Algorithm 5.5.

To start with, we randomly generate the comparison matrix according to the following rules: a) There are N items to rank and a random permutation of $[N]$ is generated as the ground truth ranking. There

Algorithm 5.5 (Improved version for practical adoption) `Try-Compare`(i, j, δ, s, h)

input: pair to query (i, j), confidence level δ, subset size s, estimated gap width h.
$m = M$.
$S_1 = [M]$.
Let $S_1^{ij} = S_1^{ji} = S_1$ and $L = \lceil \log_{4/3}(M/s) \rceil$.
for $\ell = 0, \cdots, L$ **do**
  Request $t_\ell = n_0 m / |S_\ell^{ij}|$ comparisons for pair (i, j) from each oracle $u \in S_\ell^{ij} \cup S_\ell^{ji}$. Denote c_u^{ij} as the number of times $i \succ j$.
  $\widehat{\mu}_{i,j}^{(u,\ell)} = \frac{\sum_{\ell' \in [\ell]} c_{u,\ell'}^{ij}}{\sum_{\ell' \in [\ell]} t_{\ell'}}$, $\widehat{\mu}_{j,i}^{(u,\ell)} = 1 - \widehat{\mu}_{i,j}^{(u,\ell)}$, $\widehat{\mu}_{i,j}^{(\ell)} = \frac{1}{|S_\ell^{ij}|} \sum_{u \in S_\ell^{ij}} \widehat{\mu}_{i,j}^{(u,\ell)}$.
  if $\widehat{\mu}_{i,j}^{(\ell)} - \frac{1}{2} \geq \sqrt{2 \log(2/\delta)/n_0 m}$ **then**
    return $i \succ j$
  end if
  if $\widehat{\mu}_{i,j}^{(\ell)} - \frac{1}{2} < -\sqrt{2 \log(2/\delta)/n_0 m}$ **then**
    return $i \prec j$
  end if
  $S_{\ell+1}^{ij} \leftarrow \{ v \in S_\ell^{ij} | \widehat{\mu}_{i,j}^{(v,\ell)} \geq$ medium of $\widehat{\mu}_{i,j}^{(u,\ell)}, u \in S_\ell^{ij} \}$
  $S_{\ell+1}^{ji} \leftarrow \{ v \in S_\ell^{ji} | \widehat{\mu}_{j,i}^{(v,\ell)} \geq$ medium of $\widehat{\mu}_{j,i}^{(u,\ell)}, u \in S_\ell^{ji} \}$
end for
return `unsure`.



(a) $M = 2$ (b) $M = 4$ (c) $M = 6$

Figure 5.1: Sample complexities of ranking $N \in \{2, 4, 8, 16, 32, 64\}$ items with $M - 1$ oracles have low accuracy and one oracle has high accuracy.

are M oracles that can conduct pairwise evaluation each with a comparison matrix for $p_{i,j}$. b) One of the M oracles is very accurate and we assign a probability value for $p_{i,j}$ sampled from $[0.85, 0.95]$ uniformly at random if $i \succ j$ for every pair of items. c) For the rest of $M - 1$ oracles, we assign a value to $p_{i,j}$ sampled uniformly at random from $[0.55, 0.65]$ if $i \succ j$.

We tested $N \in \{2, 4, 8, 16, 32, 64\}$ to explore how the samples grow with the size of the problem. The mix of accurate responses can also affect the performance gain. The average sample complexity of 32 runs with one standard deviation error bar calculated by Python `numpy` library is plotted in Fig. 5.1a, Fig. 5.1b and Fig. 5.1c for the case where $M = 2$, $M = 4$ and $M = 6$, respectively. In general, it is harder to derive estimated rankings while the majority of the information is noisy ($M = 6$), as in this case the sample complexity is much higher for both algorithms. However, regardless of the noisiness of oracles, the proposed method always beats the baseline. In addition, if the proportion of the noisy oracles are high, then our proposed method benefits more which is illustrated by the wider gap compared to the baseline method from $M = 2$ to $M = 6$.

Part III

Efficient Rank Aggregation in Contextual Dueling Bandits

Chapter 6

Contextual Borda Dueling Bandits

6.1 Introduction

Multi-armed bandits (MAB) (Lattimore and Szepesvári, 2020) is an interactive game where in each round, an agent chooses an arm to pull and receives a noisy reward as feedback. In contrast to numerical feedback considered in classic MAB settings, preferential feedback is more natural in various online learning tasks including information retrieval Yue and Joachims (2009), recommendation systems Sui and Burdick (2014), ranking Minka et al. (2018), crowdsourcing Chen et al. (2013), etc. Moreover, numerical feedback is also more difficult to gauge and prone to errors in many real-world applications. For example, when provided with items to shop or movies to watch, it is more natural for a customer to pick a preferred one than scoring the options. This motivates *Dueling Bandits* (Yue and Joachims, 2009), where the agent repeatedly pulls two arms at a time and is provided with feedback being the binary outcome of "duels" between the two arms.

In dueling bandits problems, the outcome of duels is commonly modeled as Bernoulli random variables due to their binary nature. In each round, suppose the agent chooses to compare arm i and j, then the binary feedback is assumed to be sampled independently from a Bernoulli distribution. For a dueling bandits instance with K arms, the probabilistic model of the instance can be fully characterized by a $K \times K$ preference probability matrix with each entry being: $p_{i,j} = \mathbb{P}(\text{arm } i \text{ is chosen over arm } j)$.

In a broader range of applications such as ranking, "arms" are often referred to as "items". We will use these two terms interchangeably in the rest of this chapter. One central goal of dueling bandits is to devise a strategy to identify the "optimal" item as quickly as possible, measured by either sample complexity or cumulative regret. However, the notion of optimality for dueling bandits is way harder to define than for multi-armed bandits. The latter can simply define the arm with the highest numerical feedback as the optimal arm, while for dueling bandits there is no obvious definition solely dependent on $\{p_{i,j} | i, j \in [K]\}$.

The first few works on dueling bandits imposed strong assumptions on $p_{i,j}$. For example, Yue et al. (2012) assumed that there exists a true ranking that is coherent among all items, and the preference probabilities must satisfy both strong stochastic transitivity (SST) and stochastic triangle inequality (STI). While relaxations like weak stochastic transitivity (Falahatgar et al., 2018) or relaxed stochastic transitivity (Yue and Joachims, 2011) exist, they typically still assume the true ranking exists and the preference probabilities are consistent, i.e., $p_{i,j} > \frac{1}{2}$ if and only if i is ranked higher than j. In reality, the existence of such coherent ranking aligned with item preferences is rarely the case. For example, $p_{i,j}$ may be interpreted as the probability of one basketball team i beating another team j, and there can be a circle among the match advantage relations.

In this chapter, we do not assume such coherent ranking exists and solely rely on the *Borda score* based on preference probabilities. The Borda score $B(i)$ of an item i is the probability that it is preferred when compared with another random item, namely $B(i) := \frac{1}{K-1} \sum_{j \neq i} p_{i,j}$. The item with the highest Borda score is called the *Borda winner*. The Borda winner is intuitively appealing and always well-defined for any set of preferential probabilities. The Borda score also does not require the problem instance to obey any consistency or transitivity, and it is considered one of the most general criteria.

To identify the Borda winner, estimations of the Borda scores are needed. Since estimating the Borda

score for one item requires comparing it with every other items, the sample complexity is prohibitively high when there are numerous items. On the other hand, in many real-world applications, the agent has access to side information that can assist the evaluation of $p_{i,j}$. For instance, an e-commerce item carries its category as well as many other attributes, and the user might have a preference for a certain category (Wang et al., 2018). For a movie, the genre and the plot as well as the directors and actors can also be taken into consideration when making choices (Liu et al., 2017).

Based on the above motivation, we consider *Generalized Linear Dueling Bandits*. In each round, the agent selects two items from a finite set of items and receives a comparison result of the preferred item. The comparisons depend on known intrinsic contexts/features associated with each pair of items. The contexts can be obtained from upstream tasks, such as topic modeling (Zhu et al., 2012) or embedding (Vasile et al., 2016). Our goal is to adaptively select items and minimize the regret with respect to the optimal item (i.e., Borda winner). Our main contributions are summarized as follows:

- We show a hardness result regarding the Borda regret minimization for the (generalized) linear model. We prove a worst-case regret lower bound $\Omega(d^{2/3}T^{2/3})$ for our dueling bandit model, showing that even in the stochastic setting, minimizing the Borda regret is difficult. The construction and proof of the lower bound are new and might be of independent interest.
- We propose an explore-then-commit type algorithm under the stochastic setting, which can achieve a nearly matching upper bound $\widetilde{O}(d^{2/3}T^{2/3})$. When the number of items K is small, the algorithm can also be configured to achieve a smaller regret $\widetilde{O}\big((d\log K)^{1/3}T^{2/3}\big)$.
- We propose an EXP3 type algorithm for linear dueling bandits under the adversarial setting, which can achieve a nearly matching upper bound $\widetilde{O}\big((d\log K)^{1/3}T^{2/3}\big)$.
- We conduct empirical studies to verify the correctness of our theoretical claims. Under both synthetic and real-world data settings, our algorithms can outperform all the baselines in terms of cumulative regret.

6.2 Related Work

Multi-armed and Contextual Bandits Multi-armed bandit is a problem of identifying the best choice in a sequential decision-making system. It has been studied in numerous ways with a wide range of applications (Even-Dar et al., 2002; Lai et al., 1985; Kuleshov and Precup, 2014). Contextual linear bandit is a special type of bandit problem where the agent is provided with side information, i.e., contexts, and rewards are assumed to have a linear structure. Various algorithms (Rusmevichientong and Tsitsiklis, 2010; Filippi et al., 2010; Abbasi-Yadkori et al., 2011; Li et al., 2017; Jun et al., 2017) have been proposed to utilize this contextual information.

Dueling Bandits and Its Performance Metrics Dueling bandits is a variant of MAB with preferential feedback (Yue et al., 2012; Zoghi et al., 2014, 2015). A comprehensive survey can be found at Bengs et al. (2021). As discussed previously, the probabilistic structure of a dueling bandits problem is governed by the preference probabilities, over which an optimal item needs to be defined. Optimality under the *Borda score* criteria has been adopted by several previous works (Jamieson et al., 2015; Falahatgar et al., 2017a; Heckel et al., 2018; Saha et al., 2021). The most relevant work to ours is Saha et al. (2021), where they studied the problem of regret minimization for adversarial dueling bandits and proved a T-round Borda regret upper bound $\widetilde{O}(K^{1/3}T^{2/3})$. They also provide an $\Omega(K^{1/3}T^{2/3})$ lower bound for stationary dueling bandits using Borda regret.

Apart from the Borda score, *Copeland score* is also a widely used criteria (Urvoy et al., 2013; Zoghi et al., 2015, 2014; Wu and Liu, 2016; Komiyama et al., 2016). It is defined as $C(i) := \frac{1}{K-1}\sum_{j\neq i}\mathbb{1}\{p_{i,j} > 1/2\}$. A Copeland winner is the item that beats the most number of other items. It can be viewed as a "thresholded" version of Borda winner. In addition to Borda and Copeland winners, optimality notions such as a von Neumann winner were also studied in Ramamohan et al. (2016); Dudík et al. (2015); Balsubramani et al. (2016).

Another line of work focuses on identifying the optimal item or the total ranking, assuming the preference probabilities are consistent. Common consistency conditions include Strong Stochastic Transitivity (Yue et al., 2012; Falahatgar et al., 2017a,b), Weak Stochastic Transitivity (Falahatgar et al., 2018; Ren et al., 2019; Wu et al., 2022; Lou et al., 2022), Relaxed Stochastic Transitivity (Yue and Joachims, 2011) and Stochastic Triangle Inequality. Sometimes the aforementioned transitivity can also be implied by some

structured models like the Bradley–Terry model. We emphasize that these consistency conditions are not assumed or implicitly implied in our setting.

Contextual Dueling Bandits In Dudík et al. (2015), contextual information is incorporated in the dueling bandits framework. Later, Saha (2021) studied a structured contextual dueling bandits setting where each item i has its own contextual vector $\mathbf{x}_i$ (sometimes called Linear Stochastic Transitivity). Each item then has an intrinsic score v_i equal to the linear product of an unknown parameter vector $\boldsymbol{\theta}^*$ and its contextual vector $\mathbf{x}_i$. The preference probability between two items i and j is assumed to be $\mu(v_i - v_j)$ where $\mu(\cdot)$ is the logistic function. These intrinsic scores of items naturally define a ranking over items. The regret is also computed as the gap between the scores of pulled items and the best item. While in this chapter, we assume that the contextual vectors are associated with item pairs and define regret on the Borda score. In Section 6.3.2, we provide a more detailed discussion showing that the setting considered in Saha (2021) can be viewed as a special case of our model.

6.3 Problem Setup and Preliminaries

We first consider the stochastic preferential feedback model with K items in the fixed time horizon setting. We denote the item set by $[K]$ and let T be the total number of rounds. in each round t, the agent can pick any pair of items (i_t, j_t) to compare and receive stochastic feedback about whether item i_t is preferred over item j_t, (denoted by $i_t \succ j_t$). We denote the probability of seeing the event $i \succ j$ as $p_{i,j} \in [0, 1]$. Naturally, we assume $p_{i,j} + p_{j,i} = 1$, and $p_{i,i} = 1/2$.

In this chapter, we are concerned with the generalized linear model (GLM), where there is assumed to exist an *unknown* parameter $\boldsymbol{\theta}^* \in \mathbb{R}^d$, and each pair of items (i, j) has its own *known* contextual/feature vector $\boldsymbol{\phi}_{i,j} \in \mathbb{R}^d$ with $\|\boldsymbol{\phi}_{i,j}\| \leq 1$. There is also a fixed known link function (sometimes called comparison function) $\mu(\cdot)$ that is monotonically increasing and satisfies $\mu(x) + \mu(-x) = 1$, e.g. a linear function or the logistic function $\mu(x) = 1/(1 + e^{-x})$. The preference probability is defined as $p_{i,j} = \mu(\boldsymbol{\phi}_{i,j}^\top \boldsymbol{\theta}^*)$. In each round, denote $r_t = \mathbb{1}\{i_t \succ j_t\}$, then we have

$$\mathbb{E}[r_t | i_t, j_t] = p_{i_t, j_t} = \mu(\boldsymbol{\phi}_{i_t, j_t}^\top \boldsymbol{\theta}^*).$$

Then our model can also be written as

$$r_t = \mu(\boldsymbol{\phi}_{i_t, j_t}^\top \boldsymbol{\theta}^*) + \epsilon_t,$$

where the noises $\{\epsilon_t\}_{t \in [T]}$ are zero-mean, 1-sub-Gaussian and assumed independent from each other. Note that, given the constraint $p_{i,j} + p_{j,i} = 1$, it is implied that $\boldsymbol{\phi}_{i,j} = -\boldsymbol{\phi}_{j,i}$ for any $i \in [K], j \in [K]$.

The agent's goal is to maximize the cumulative Borda score. The (slightly modified [1]) Borda score of item i is defined as $B(i) = \frac{1}{K} \sum_{j=1}^{K} p_{i,j}$, and the Borda winner is defined as $i^* = \operatorname{argmax}_{i \in [K]} B(i)$. The problem of merely identifying the Borda winner was deemed trivial (Zoghi et al., 2014; Bengs et al., 2021) because for a fixed item i, uniformly random sampling j and receiving feedback $r_{i,j} = \text{Bernoulli}(p_{i,j})$ yield a Bernoulli random variable with its expectation being the Borda score $B(i)$. This so-called *Borda reduction* trick makes identifying the Borda winner as easy as the best-arm identification for K-armed bandits. Moreover, if the regret is defined as $\text{Regret}(T) = \sum_{t=1}^{T} (B(i^*) - B(i_t))$, then any optimal algorithms for multi-arm bandits can achieve $\widetilde{O}(\sqrt{T})$ regret.

However, the above definition of regret does not respect the fact that a pair of items is selected in each round. When the agent chooses two items to compare, it is natural to define the regret so that both items contribute equally. A commonly used regret, e.g., in Saha et al. (2021), has the following form:

$$\text{Regret}(T) = \sum_{t=1}^{T} \big(2B(i^*) - B(i_t) - B(j_t)\big), \tag{6.3.1}$$

where the regret is defined as the sum of the sub-optimality of both selected arms. Sub-optimality is measured by the gap between the Borda scores of the compared items and the Borda winner. This form of regret deems

[1] Previous works define Borda score as $B'_i = \frac{1}{K-1} \sum_{j \neq i} p_{i,j}$, excluding the diagonal term $p_{i,i} = 1/2$. Our definition is equivalent since the difference between two items satisfies $B(i) - B_j = \frac{K-1}{K}(B'_i - B'_j)$. Therefore, the regret will be in the same order for both definitions.

any classical multi-arm bandit algorithm with Borda reduction vacuous because taking j_t into consideration will invoke $\Theta(T)$ regret.

Adversarial Setting Saha et al. (2021) considered an adversarial setting for the multi-armed case, where in each round t, the comparison follows a potentially different probability model, denoted by $\{p_{i,j}^t\}_{i,j\in[K]}$. In this chapter, we consider its contextual counterpart. Formally, we assume there is an underlying parameter $\boldsymbol{\theta}_t^*$, and in round t, the preference probability is defined as $p_{i,j}^t = \mu(\boldsymbol{\phi}_{i,j}^\top \boldsymbol{\theta}_t^*)$.

The Borda score of item $i \in [K]$ in round t is defined as $B_t(i) = \frac{1}{K}\sum_{j=1}^K p_{i,j}^t$, and the Borda winner in round T is defined as $i^* = \operatorname{argmax}_{i\in[K]} \sum_{t=1}^T B_t(i)$. The T-round regret is thus defined as $\mathrm{Regret}(T) = \sum_{t=1}^T \big(2B_t(i^*) - B_t(i_t) - B_t(j_t)\big)$.

6.3.1 Assumptions

In this section, we present the assumptions required for establishing theoretical guarantees. Due to the fact that the analysis technique is largely extracted from Li et al. (2017), we follow them to make assumptions to enable regret minimization for generalized linear dueling bandits.

We make a regularity assumption about the distribution of the contextual vectors:

Assumption 6.3.1. There exists a constant $\lambda_0 > 0$ such that $\lambda_{\min}\big(\frac{1}{K^2}\sum_{i=1}^K\sum_{j=1}^K \boldsymbol{\phi}_{i,j}\boldsymbol{\phi}_{i,j}^\top\big) \geq \lambda_0$.

This assumption is only utilized to initialize the design matrix $\mathbf{V}_\tau = \sum_{t=1}^\tau \boldsymbol{\phi}_{i_t,j_t}\boldsymbol{\phi}_{i_t,j_t}^\top$ so that the minimum eigenvalue is large enough. We follow Li et al. (2017) to deem λ_0 as a constant.

We also need the following assumption regarding the link function $\mu(\cdot)$:

Assumption 6.3.2. Let $\dot{\mu}$ be the first-order derivative of μ. We have $\kappa := \inf_{\|\mathbf{x}\|\leq 1, \|\boldsymbol{\theta}-\boldsymbol{\theta}^*\|\leq 1} \dot{\mu}(\mathbf{x}^\top\boldsymbol{\theta}) > 0$.

Assuming $\kappa > 0$ is necessary to ensure the maximum log-likelihood estimator can converge to the true parameter $\boldsymbol{\theta}^*$ (Li et al., 2017, Section 3). This type of assumption is commonly made in previous works for generalized linear models (Filippi et al., 2010; Li et al., 2017; Faury et al., 2020).

Another common assumption is regarding the continuity and smoothness of the link function.

Assumption 6.3.3. μ is twice differentiable. Its first and second-order derivatives are upper-bounded by constants L_μ and M_μ respectively.

This is a very mild assumption. For example, it is easy to verify that the logistic link function satisfies Theorem 6.3.3 with $L_\mu = M_\mu = 1/4$.

6.3.2 Existing Results for Structured Contexts

A structural assumption made by some previous works (Saha, 2021) is that $\boldsymbol{\phi}_{i,j} = \mathbf{x}_i - \mathbf{x}_j$, where $\mathbf{x}_i$ can be seen as some feature vectors tied to the item. In this work, we do not consider minimizing the Borda regret under the structural assumption.

The immediate reason is that, when $p_{i,j} = \mu(\mathbf{x}_i^\top\boldsymbol{\theta}^* - \mathbf{x}_j^\top\boldsymbol{\theta}^*)$, with $\mu(\cdot)$ being the logistic function, the probability model $p_{i,j}$ effectively becomes (a linear version of) the well-known Bradley-Terry model. Namely, each item is tied to a value $v_i = \mathbf{x}_i^\top\boldsymbol{\theta}^*$, and the comparison probability follows $p_{i,j} = \frac{e^{v_i}}{e^{v_i}+e^{v_j}}$. More importantly, this kind of model satisfies both the strong stochastic transitivity (SST) and the stochastic triangle inequality (STI), which are unlikely to satisfy in reality.

Furthermore, when stochastic transitivity holds, there is a true ranking among the items, determined by $\mathbf{x}_i^\top\boldsymbol{\theta}^*$. A true ranking renders concepts like the Borda winner or Copeland winner redundant because the rank-one item will always be the winner in every sense. When $\boldsymbol{\phi}_{i,j} = \mathbf{x}_i - \mathbf{x}_j$, Saha (2021) proposed algorithms that can achieve nearly optimal regret $\widetilde{O}(d\sqrt{T})$, with regret being defined as

$$\mathrm{Regret}(T) = \sum_{t=1}^T 2\langle \mathbf{x}_{i^*}, \boldsymbol{\theta}^*\rangle - \langle \mathbf{x}_{i_t}, \boldsymbol{\theta}^*\rangle - \langle \mathbf{x}_{j_t}, \boldsymbol{\theta}^*\rangle, \tag{6.3.2}$$

where $i^* = \text{argmax}_i \langle \mathbf{x}_i, \boldsymbol{\theta}^* \rangle$, which also happens to be the Borda winner. Meanwhile, by Theorem 6.3.3,

$$\begin{aligned} & B(i^*) - B(j) \\ &= \frac{1}{K} \sum_{k=1}^{K} \left[\mu(\langle \mathbf{x}_{i^*} - \mathbf{x}_k, \boldsymbol{\theta}^* \rangle) - \mu(\langle \mathbf{x}_j - \mathbf{x}_k, \boldsymbol{\theta}^* \rangle) \right] \\ &\leq L_\mu \cdot \langle \mathbf{x}_{i^*} - \mathbf{x}_j, \boldsymbol{\theta}^* \rangle, \end{aligned}$$

where L_μ is the upper bound on the derivative of $\mu(\cdot)$. For logistic function $L_\mu = 1/4$. The Borda regret (6.3.1) is thus at most a constant multiple of (6.3.2). This shows Borda regret minimization can be sufficiently solved by Saha (2021) when structured contexts are present. We consider the most general case where the only restriction is the implicit assumption that $\boldsymbol{\phi}_{i,j} = -\boldsymbol{\phi}_{j,i}$.

6.4 Proposed Algorithm for Generalized Contextual Dueling Bandits

Algorithm 6.1 BETC-GLM

1: **Input:** time horizon T, number of items K, feature dimension d, feature vectors $\boldsymbol{\phi}_{i,j}$ for $i \in [K]$, $j \in [K]$, exploration rounds τ, error tolerance ϵ, failure probability δ.
2: **for** $t = 1, 2, \ldots, \tau$ **do**
3: sample $i_t \sim \text{Uniform}([K])$, $j_t \sim \text{Uniform}([K])$
4: query pair (i_t, j_t) and receive feedback r_t
5: **end for**
6: Find the G-optimal design $\pi(i, j)$ based on $\boldsymbol{\phi}_{i,j}$ for $i \in [K]$, $j \in [K]$
7: Let $N(i,j) = \left\lceil \frac{d\pi(i,j)}{\epsilon^2} \right\rceil$ for any $(i,j) \in \text{supp}(\pi)$, denote $N = \sum_{i=1}^{K} \sum_{j=1}^{K} N(i,j)$
8: **for** $i \in [K]$, $j \in [K]$, $s \in [N(i,j)]$ **do**
9: set $t \leftarrow t + 1$, set $(i_t, j_t) = (i, j)$
10: query pair (i_t, j_t) and receive feedback r_t
11: **end for**
12: Calculate the empirical MLE estimator $\widehat{\boldsymbol{\theta}}_{\tau+N}$ based on all $\tau + N$ samples via (6.4.1)
13: Estimate the Borda score for each item:

$$\widehat{B}(i) = \frac{1}{K} \sum_{j=1}^{K} \mu(\boldsymbol{\phi}_{i,j}^\top \widehat{\boldsymbol{\theta}}_{\tau+N}), \qquad \widehat{i} = \underset{i \in [K]}{\text{argmax}}\, \widehat{B}(i)$$

14: Keep querying $(\widehat{i}, \widehat{i})$ for the rest of the time.

We propose an algorithm named Borda Explore-Then-Commit for Generalized Linear Models (BETC-GLM), presented in Algorithm 6.1. Our algorithm is inspired by the algorithm for generalized linear models proposed by Li et al. (2017).

At the high level, Algorithm 6.1 can be divided into two phases: the exploration phase (Line 2-11) and the exploitation phase (Line 12-14). The exploration phase ensures that the MLE estimator $\widehat{\boldsymbol{\theta}}$ is accurate enough so that the estimated Borda score is within $\widetilde{O}(\epsilon)$-range of the true Borda score (ignoring other quantities). Then the exploitation phase simply chooses the empirical Borda winner to incur small regret.

During the exploration phase, the algorithm first performs "pure exploration" (Line 2-5), which can be seen as an initialization step for the algorithm. The purpose of this step is to ensure the design matrix $\mathbf{V}_{\tau+N} = \sum_{t=1}^{\tau+N} \boldsymbol{\phi}_{i_t,j_t} \boldsymbol{\phi}_{i_t,j_t}^\top$ is positive definite.

After that, the algorithm will perform the "designed exploration". Line 6 will find the G-optimal design, which minimizes the objective function $g(\pi) = \max_{i,j} \|\boldsymbol{\phi}_{i,j}\|^2_{\mathbf{V}(\pi)^{-1}}$, where $\mathbf{V}(\pi) := \sum_{i,j} \pi(i,j) \boldsymbol{\phi}_{i,j} \boldsymbol{\phi}_{i,j}^\top$. The G-optimal design $\pi^*(\cdot)$ satisfies $\|\boldsymbol{\phi}_{i,j}\|^2_{\mathbf{V}(\pi^*)^{-1}} \leq d$, and can be efficiently approximated by the Frank-Wolfe

algorithm (See Theorem 6.4.4 for a detailed discussion). Then the algorithm will follow $\pi(\cdot)$ found at Line 6 to determine how many samples (Line 7) are needed. At Line 8-11, there are in total $N = \sum_{i=1}^{K}\sum_{j=1}^{K} N(i,j)$ samples queried, and the algorithm shall index them by $t = \tau+1, \tau+2, \ldots, \tau+N$.

At Line 12, the algorithm collects all the $\tau+N$ samples and performs the maximum likelihood estimation (MLE). For the generalized linear model, the MLE estimator $\widehat{\boldsymbol{\theta}}_{\tau+N}$ satisfies:

$$\sum_{t=1}^{\tau+N} \mu(\boldsymbol{\phi}_{i_t,j_t}^\top \widehat{\boldsymbol{\theta}}_{\tau+N})\boldsymbol{\phi}_{i_t,j_t} = \sum_{t=1}^{\tau+N} r_t \boldsymbol{\phi}_{i_t,j_t}, \tag{6.4.1}$$

or equivalently, it can be determined by solving a strongly concave optimization problem:

$$\widehat{\boldsymbol{\theta}}_{\tau+N} \in \operatorname*{argmax}_{\boldsymbol{\theta}} \sum_{t=1}^{\tau+N} \Big(r_t \boldsymbol{\phi}_{i_t,j_t}^\top \boldsymbol{\theta} - m(\boldsymbol{\phi}_{i_t,j_t}^\top \boldsymbol{\theta}) \Big),$$

where $\dot{m}(\cdot) = \mu(\cdot)$. For the logistic link function, $m(x) = \log(1+e^x)$. As a special case of our generalized linear model, the linear model has a closed-form solution for (6.4.1). For example, if $\mu(x) = \frac{1}{2} + x$, i.e. $p_{i,j} = \frac{1}{2} + \boldsymbol{\phi}_{i,j}^\top \boldsymbol{\theta}^*$, then (6.4.1) becomes:

$$\widehat{\boldsymbol{\theta}}_{\tau+N} = \mathbf{V}_{\tau+N}^{-1} \sum_{t=1}^{\tau+N} (r_t - 1/2)\boldsymbol{\phi}_{i_t,j_t},$$

where $\mathbf{V}_{\tau+N} = \sum_{t=1}^{\tau+N} \boldsymbol{\phi}_{i_t,j_t}\boldsymbol{\phi}_{i_t,j_t}^\top$.

After the MLE estimator is obtained, Line 13 will calculate the estimated Borda score $\widehat{B}(i)$ for each item based on $\widehat{\boldsymbol{\theta}}_{\tau+N}$, and pick the empirically best one.

The theoretical guarantee of G-optimal design is provided below: given an action set $\mathcal{X} \subseteq \mathbb{R}^d$ that is compact and $\operatorname{span}(\mathcal{X}) = \mathbb{R}^d$. A fixed design $\pi(\cdot) : \mathcal{X} \to [0,1]$ satisfies $\sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x}) = 1$. Define $\mathbf{V}(\pi) := \sum_{\mathbf{x}\in\mathcal{X}} \pi(\mathbf{x})\mathbf{x}\mathbf{x}^\top$ and $g(\pi) := \max_{\mathbf{x}\in\mathcal{X}} \|\mathbf{x}\|^2_{\mathbf{V}(\pi)^{-1}}$.

Lemma 6.4.1 (The Kiefer–Wolfowitz Theorem, Section 21.1, Lattimore and Szepesvári (2020))**.** There exists an optimal design $\pi^*(\cdot)$ such that $|\operatorname{supp}(\pi)| \le d(d+1)/2$, and satisfies:

1. $g(\pi^*) = d$.
2. π^* is the minimizer of $g(\cdot)$.

Remark 6.4.2 (Regret for Fewer Arms)**.** In typical scenarios, the number of items K is not exponentially large in the dimension d. In this case, we can choose a different parameter set of τ and ϵ such that Algorithm 6.1 can achieve a smaller regret bound $\widetilde{O}\big(\kappa^{-1}(d\log K)^{1/3}T^{2/3}\big)$ with smaller dependence on the dimension d.

Remark 6.4.3 (Regret for Infinitely Many Arms)**.** In most practical scenarios of dueling bandits, it is adequate to consider a finite number K of items (e.g., ranking items). Nonetheless, BETC-GLM can be easily adapted to accommodate infinitely many arms in terms of regret. We can construct a covering over all $\boldsymbol{\phi}_{i,j}$ and perform optimal design and exploration on the covering set. The resulting regret will be the same as our upper bound, i.e., $\widetilde{O}(d^{2/3}T^{2/3})$ up to some error caused by the epsilon net argument.

Remark 6.4.4 (Approximate G-optimal Design)**.** Algorithm 6.1 assumes an exact G-optimal design π is obtained. In the experiments, we use the Frank-Wolfe algorithm to solve the constraint optimization problem (See Algorithm 6.5, Section 6.7.3). To find a policy π such that $g(\pi) \le (1+\varepsilon)g(\pi^*)$, roughly $O(d/\varepsilon)$ optimization steps are needed. Such a near-optimal design will introduce a factor of $(1+\varepsilon)^{1/3}$ into the upper bounds.

Remark 6.4.5 (Computational Complexity)**.** While the regret or sample complexity does not rely on the number of arms K, the computation complexity of any algorithm will inevitably suffer from large K. It is clear that $\Omega(K^2)$ operations are necessary to at least traverse over all contextual vectors $\boldsymbol{\phi}_{i,j}$. For Algorithm 6.1, the most computation-intensive part is the G-optimal design, where each optimization step will take $O(d^2K^2)$ basic operations. A breakdown of this cost is available in Section 6.7.3. To solve (6.4.1), gradient descent cab be used and each step requires $O(dK^2)$ operations.

6.5 Proposed Algorithm for Adversarial Contextual Dueling Bandit

This section addresses Borda regret minimization under the adversarial setting. As we introduced in Section 6.3, the unknown parameter $\boldsymbol{\theta}_t$ can vary for each round t, while the contextual vectors $\boldsymbol{\phi}_{i,j}$ are fixed.

Our proposed algorithm, BEXP3, is designed for the contextual linear model. Formally, in round t and given pair (i, j), we have $p^t_{i,j} = \frac{1}{2} + \langle \boldsymbol{\phi}_{i,j}, \boldsymbol{\theta}^*_t \rangle$.

6.5.1 Algorithm Description

Algorithm 6.2 BEXP3

1: **Input:** time horizon T, number of items K, feature dimension d, feature vectors $\boldsymbol{\phi}_{i,j}$ for $i \in [K]$, $j \in [K]$, learning rate η, exploration parameter γ.
2: **Initialize:** $q_1(i) = \frac{1}{K}$.
3: **for** $t = 1, \ldots, T$ **do**
4: Sample items $i_t \sim q_t$, $j_t \sim q_t$.
5: Query pair (i_t, j_t) and receive feedback r_t
6: Calculate $Q_t = \sum_{i\in[K]} \sum_{j\in[K]} q_t(i) q_t(j) \boldsymbol{\phi}_{i,j} \boldsymbol{\phi}_{i,j}^\top$, $\widehat{\boldsymbol{\theta}}_t = Q_t^{-1} \boldsymbol{\phi}_{i_t,j_t} r_t$.
7: Calculate the (shifted) Borda score estimates $\widehat{B}_t(i) = \langle \frac{1}{K} \sum_{j\in[K]} \boldsymbol{\phi}_{i,j}, \widehat{\boldsymbol{\theta}}_t \rangle$.
8: Update for all $i \in [K]$, set

$$\widetilde{q}_{t+1}(i) = \frac{\exp(\eta \sum_{l=1}^t \widehat{B}_l(i))}{\sum_{j\in[K]} \exp(\eta \sum_{l=1}^t \widehat{B}_l(j))}; \qquad q_{t+1}(i) = (1-\gamma)\widetilde{q}_{t+1}(i) + \frac{\gamma}{K}.$$

9: **end for**

Algorithm 6.2 is adapted from the DEXP3 algorithm in Saha et al. (2021), which deals with the adversarial multi-armed dueling bandit. Algorithm 6.2 maintains a distribution $q_t(\cdot)$ over $[K]$, initialized as uniform distribution (Line 2). In every round t, two items are chosen following q_t independently. Then Line 6 calculates the one-sample unbiased estimate $\widehat{\boldsymbol{\theta}}_t$ of the true underlying parameter $\boldsymbol{\theta}^*_t$. Line 7 further calculates the unbiased estimate of the (shifted) Borda score. Note that the true Borda score in round t satisfies $B_t(i) = \frac{1}{2} + \langle \frac{1}{K} \sum_{j\in[K]} \boldsymbol{\phi}_{i,j}, \boldsymbol{\theta}^*_t \rangle$. $\widehat{B}_t$ instead only estimates the second term of the Borda score. This is a choice to simplify the proof. The cumulative estimated score $\sum_{l=1}^t \widehat{B}_l(i)$ can be seen as the estimated cumulative reward of item i in round t. In Line 8, q_{t+1} is defined by the classic exponential weight update, along with a uniform exploration policy controlled by γ.

6.6 Construction of Hardness Cases

This specific construction emphasizes the intrinsic hardness of Borda regret minimization: to differentiate the best item from its close competitors, the algorithm must query the bad items to gain information.

The construction of this hard instance for linear dueling bandits is inspired by the worst-case lower bound for the stochastic linear bandit (Dani et al., 2008), which has the order $\Omega(d\sqrt{T})$, while ours is $\Omega(d^{2/3}T^{2/3})$. The difference is that for the linear or multi-armed stochastic bandit, eliminating bad arms can make further exploration less expensive. But in our case, any amount of exploration will not reduce the cost of further exploration. This essentially means that exploration and exploitation must be separate, which is also supported by the fact that a simple explore-then-commit algorithm shown in Section 6.4 can be nearly optimal.

For any $d > 0$, we construct a hard instance with 2^{d+1} items (indexed from 0 to $2^{d+1} - 1$). We construct

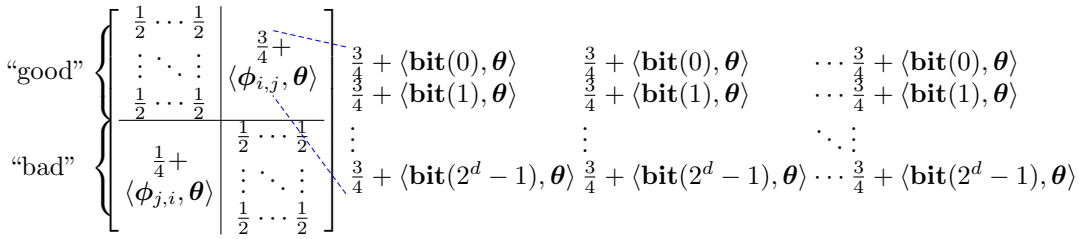


Figure 6.1: Illustration of the hard-to-learn preference probability matrix $\{p^{\boldsymbol{\theta}}_{i,j}\}_{i\in[K],j\in[K]}$. There are $K = 2^{d+1}$ items in total. The first 2^d items are "good" items with higher Borda scores, and the last 2^d items are "bad" items. The upper right block $\{p_{i,j}\}_{i<2^d,j\geq 2^d}$ is defined as shown in the blue bubble. The lower left block satisfies $p_{i,j} = 1 - p_{j,i}$. For any $\boldsymbol{\theta}$, there exist one and only best item i such that $\mathbf{bit}(i) = \mathbf{sign}(\boldsymbol{\theta})$.

the hard instance $p^{\boldsymbol{\theta}}_{i,j}$ for any $\boldsymbol{\theta} \in \{-\Delta, +\Delta\}^d$ as:

$$p^{\boldsymbol{\theta}}_{i,j} = \begin{cases} \frac{1}{2}, & \text{if } i < 2^d, j < 2^d \\ \frac{1}{2}, & \text{if } i \geq 2^d, j \geq 2^d \\ \frac{3}{4}, & \text{if } i < 2^d, j \geq 2^d \\ \frac{1}{4}, & \text{if } i \geq 2^d, j < 2^d \end{cases} + \langle \boldsymbol{\phi}_{i,j}, \boldsymbol{\theta} \rangle, \tag{6.6.1}$$

where the feature vectors $\boldsymbol{\phi}_{i,j}$ and the parameter $\boldsymbol{\theta}$ are of dimension d, and have the following forms:

$$\boldsymbol{\phi}_{i,j} = \begin{cases} \mathbf{0}, & \text{if } i < 2^d, j < 2^d \\ \mathbf{0}, & \text{if } i \geq 2^d, j \geq 2^d \\ \mathbf{bit}(i), & \text{if } i < 2^d, j \geq 2^d \\ -\mathbf{bit}(j), & \text{if } i \geq 2^d, j < 2^d, \end{cases}$$

where $\mathbf{bit}(\cdot)$ is the (shifted) bit representation of non-negative integers, i.e., suppose $x = b_0 \times 2^0 + b_1 \times 2^1 + \cdots + b_{d-1} \times 2^{d-1}$, then $\mathbf{bit}(x) = 2\boldsymbol{b} - 1$. Note that $\mathbf{bit}(\cdot) \in \{-1, +1\}^d$, and $\boldsymbol{\phi}_{i,j} = -\boldsymbol{\phi}_{j,i}$.

We rewrite (6.6.1) as:

$$p^{\boldsymbol{\theta}}_{i,j} = \begin{cases} \frac{1}{2}, & \text{if } i < 2^d, j < 2^d \\ \frac{1}{2}, & \text{if } i \geq 2^d, j \geq 2^d \\ \frac{3}{4}, & \text{if } i < 2^d, j \geq 2^d \\ \frac{1}{4}, & \text{if } i \geq 2^d, j < 2^d \end{cases} + \begin{cases} 0, & \text{if } i < 2^d, j < 2^d \\ 0, & \text{if } i \geq 2^d, j \geq 2^d \\ \langle \mathbf{bit}(i), \boldsymbol{\theta} \rangle, & \text{if } i < 2^d, j \geq 2^d \\ -\langle \mathbf{bit}(j), \boldsymbol{\theta} \rangle, & \text{if } i \geq 2^d, j < 2^d, \end{cases} \tag{6.6.2}$$

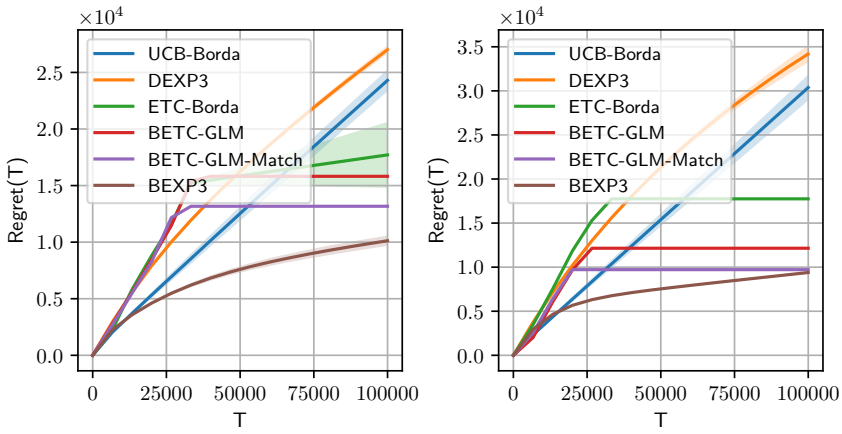
and the Borda scores are:

$$B^{\boldsymbol{\theta}}(i) = \begin{cases} \frac{5}{8} + \frac{1}{2}\langle \mathbf{bit}(i), \boldsymbol{\theta} \rangle, & \text{if } i < 2^d, \\ \frac{3}{8}, & \text{if } i \geq 2^d. \end{cases}$$

Intuitively, the former half arms indexed from 0 to $2^d - 1$ are "good" arms (one among them is optimal), while the latter half arms are "bad" arms. It is clear that choosing a "bad" arm i will incur regret $B(i^*) - B(i) \geq 1/4$.

6.7 Experiments

This section compares the proposed algorithm BETC-GLM with existing ones that are capable of minimizing Borda regret. We use random responses (generated from fixed preferential matrices) to interact with all tested algorithms. Each algorithm is run for 50 times over a time horizon of $T = 10^6$. We report both the mean



(a) Generated Hard Case (b) EventTime

Figure 6.2: The regret of the proposed algorithms (BETC-GLM, BEXP3) and the baseline algorithms (UCB-Borda, DEXP3, ETC-Borda).

and the standard deviation of the cumulative Borda regret and supply some analysis. The following list summarizes all methods we study in this section: BETC-GLM(-Match): Algorithm 6.1 proposed in this chapter. For general link function, to find $\widehat{\boldsymbol{\theta}}$ by MLE in (6.4.1), 100 rounds of gradient descent are performed. The failure probability is set to $\delta = 1/T$.

UCB-Borda: The UCB algorithm (Auer et al., 2002) using *Borda reduction* technique mentioned by Bengs et al. (2021). The complete listing is displayed in Algorithm 6.3.

DEXP3: Dueling-Exp3 is an adversarial Borda bandit algorithm developed by Saha et al. (2021), which also applies to our stationary bandit case. Relevant tuning parameters are set according to their upper-bound proof.

ETC-Borda: We devise a simple explore-then-commit algorithm, named ETC-Borda. Like DEXP3, ETC-Borda does not take any contextual information into account. The complete procedure of ETC-Borda is displayed in Algorithm 6.4, Section 6.7.3. The failure probability δ is optimized as $1/T$.

BEXP3: The proposed method for adversarial Borda bandits displayed in Algorithm 6.2.

6.7.1 Simulated Study: Generated Hard Case

We first test the algorithms on the hard instances constructed in Section 6.6. We generate $\boldsymbol{\theta}^*$ randomly from $\{-\Delta, +\Delta\}^d$ with $\Delta = \frac{1}{4d}$ so that the comparison probabilities $p^{\boldsymbol{\theta}^*}_{i,j} \in [0,1]$ for all $i, j \in [K]$. We pick the dimension $d = 6$ and the number of arms is therefore $K = 2^{d+1} = 128$. Note the dual usage of d in our construction and the model setup in Section 6.3.

As depicted in Fig. 6.2a, the proposed algorithms (BETC-GLM, BEXP3) outperform the baseline algorithms in terms of cumulative regret when reaching the end of time horizon T. For UCB-Borda, since it is not tailored for the dueling regret definition, it suffers from a linear regret as its second arm is always sampled uniformly at random, leading to a constant regret per round. DEXP3 and ETC-Borda are two algorithms designed for K-armed dueling bandits. Both are unable to utilize contextual information and thus demand more exploration. As expected, their regrets are higher than BETC-GLM or BEXP3.

In Fig. 6.3 we show that under the same experimental setting, tuning the error tolerance ϵ in BETC can further reduce its total regret up to a constant factor, showing that under suitable hyper-parameter choices, BETC can outperform BEXP3.

6.7.2 Real-world Data Experiments

To showcase the performance of the algorithms in a real-world setting, we use the EventTime dataset (Zhang et al., 2016). In this dataset, $K = 100$ historical events are compared in a pairwise fashion by crowd-sourced workers. We first calculate the empirical preference probabilities $\widetilde{p}_{i,j}$ from the collected responses,

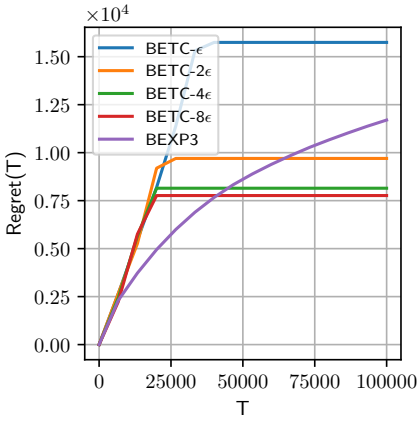


Figure 6.3: The performance of BETC under different choices of error tolerance ϵ, compared with BEXP3. We examined BETC with ϵ, 2ϵ, 4ϵ, 8ϵ where $\epsilon = d^{1/6}T^{-1/3}$.

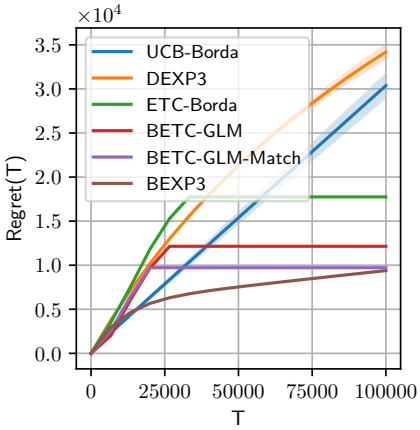


Figure 6.4: EventTime

Figure 6.5: The regret of the proposed algorithm (BETC-GLM,BEXP3) and the baseline algorithms (UCB-Borda, DEXP3, ETC-Borda).

and construct a generalized linear model based on the empirical preference probabilities. The algorithms are tested under this generalized linear model. Due to space limitations, more details are deferred to Section 6.7.2.

As depicted in Fig. 6.2b, the proposed algorithm BETC-GLM outperforms the baseline algorithms in terms of cumulative regret when reaching the end of time horizon T. The other proposed algorithm BEXP3 performs equally well even when misspecified (the algorithm is designed for the linear setting, while the comparison probability follows a logistic model).

We first calculate the empirical preference probabilities $\widetilde{p}_{i,j}$ from the collected responses. During simulation, $\widetilde{p}_{i,j}$ is the parameter of the Bernoulli distribution that is used to generate the responses whenever a pair (i, j) is queried. The contextual vectors $\boldsymbol{\phi}_{i,j}$ are generated randomly from $\{-1, +1\}^5$. For simplicity, we assign the item pairs that have the same probability value with the same contextual vector, i.e., if $\widetilde{p}_{i,j} = \widetilde{p}_{k,l}$ then $\boldsymbol{\phi}_{i,j} = \boldsymbol{\phi}_{k,l}$. The MLE estimator $\widehat{\boldsymbol{\theta}}$ in (6.4.1) is obtained to construct the recovered preference probability $\widehat{p}_{i,j} := \mu(\boldsymbol{\phi}_{i,j}^\top \widehat{\boldsymbol{\theta}})$ where $\mu(x) = 1/(1 + e^{-x})$ is the logistic function. We ensure that the recovered preference probability $\widehat{p}_{i,j}$ is close to $\widetilde{p}_{i,j}$, so that $\boldsymbol{\phi}_{i,j}$ are informative enough. As shown in Fig. 6.4, our algorithm outperforms the baseline methods as expected. In particular, the gap between our algorithm and the baselines is even larger than that under the generated hard case. In both settings, our algorithms demonstrated a stable performance with negligible variance.

6.7.3 Additional Information for Experiments

The UCB-Borda Algorithm

The UCB-Borda procedure, displayed in Algorithm 6.3 is a UCB algorithm with Borda reduction only capable of minimization of regret in the following form:

$$\text{Regret}(T) = \sum_{t=1}^{T} \big(B(i^*) - B(i_t)\big).$$

Let $\mathbf{n}_i$ be the number of times arm $i \in [K]$ has been queried. Let $\mathbf{w}_i$ be the number of times arm i wins the duel. $\widehat{B}(i)$ is the estimated Borda score. α is set to 0.3 in all experiments.

Algorithm 6.3 UCB-Borda

Input: time horizon T, number of items K, exploration parameter α.
Initialize: $\mathbf{n} = \mathbf{w} = \{0\}^K$, $\widehat{B}(i) = \frac{1}{2}, i \in [K]$
for $t = 1, 2, \ldots, T$ **do**
  $i_t = \text{argmax}_{k \in [K]} \left(\widehat{B}_k + \sqrt{\frac{\alpha \log(t)}{\mathbf{n}_k}}\right)$
  sample $j_t \sim \text{Uniform}([K])$
  query pair (i_t, j_t) and receive feedback $r_t \sim \text{Bernoulli}(p_{i_t, j_t})$
  $\mathbf{n}_{i_t} = \mathbf{n}_{i_t} + 1$, $\mathbf{w}_{i_t} = \mathbf{w}_{i_t} + r_t$, $\widehat{B}(i_t) = \frac{\mathbf{w}_{i_t}}{\mathbf{n}_{i_t}}$
end for

The ETC-Borda Algorithm

The ETC-Borda procedure, displayed in Algorithm 6.4 is an explore-then-commit type algorithm capable of minimizing the Borda dueling regret. It can be shown that the regret of Algorithm 6.4 is $\widetilde{O}(K^{1/3}T^{2/3})$.

Algorithm 6.4 ETC-Borda

Input: time horizon T, number of items K, target failure probability δ
Initialize: $\mathbf{n} = \mathbf{w} = \{0\}^K$, $\widehat{B}(i) = \frac{1}{2}, i \in [K]$
Set $N = \lceil K^{-2/3} T^{2/3} \log(K/\delta)^{1/3} \rceil$
for $t = 1, 2, \ldots, T$ **do**
  Choose action $i_t \leftarrow \begin{cases} 1 + (t-1) \bmod K, & \text{if } t \le KN, \\ \text{argmax}_{i \in [K]} \widehat{B}(i), & \text{if } t > KN. \end{cases}$
  Choose action $j_t = \begin{cases} \text{Uniform}([K]), & \text{if } t \le KN, \\ \text{argmax}_{i \in [K]} \widehat{B}(i), & \text{if } t > KN. \end{cases}$
  query pair (i_t, j_t) and receive feedback $r_t \sim \text{Bernoulli}(p_{i_t, j_t})$
  if $t \le N$ **then**
    $\mathbf{n}_{i_t} = \mathbf{n}_{i_t} + 1$, $\mathbf{w}_{i_t} = \mathbf{w}_{i_t} + r_t$, $\widehat{B}(i_t) = \frac{\mathbf{w}_{i_t}}{\mathbf{n}_{i_t}}$
  end if
end for

Frank-Wolfe algorithm used to find approximate solution for G-optimal design

In order to find a solution for the G-optimal design problem, we resort to the Frank-Wolfe algorithm to find an approximate solution. The detailed procedure is listed in Algorithm 6.5. In Line 4, each outer product costs d^2 multiplications, K^2 such matrices are scaled and summed into a d-by-d matrix $\mathbf{V}(\pi)$, which costs $O(K^2 d^2)$ operations in total. In Line 5, one matrix inversion costs approximately $O(d^3)$. The weighted norm requires $O(d^2)$ and the maximum is taken over K^2 such calculated values. The scaling and update in the following lines only require $O(K^2)$. In summary, the algorithm is dominated by the calculation in Line 5 which costs $O(d^2 K^2)$.

In experiments, the G-optimal design $\pi(i,j)$ is approximated by running 20 iterations of Frank-Wolfe algorithm, which is more than enough for its convergence given our particular problem instance. (See Note 21.2 in (Lattimore and Szepesvári, 2020)).

Algorithm 6.5 G-OPTIMAL DESIGN BY FRANK-WOLFE

Input: number of items K, contextual vectors $\boldsymbol{\phi}_{i,j}, i \in [K], j \in [K]$, number of iterations R
Initialize: $\pi_1(i,j) = 1/K^2$
for $r = 1, 2, \cdots, R$ **do**
  $\mathbf{V}(\pi_r) = \sum_{i,j} \pi_r(i,j) \boldsymbol{\phi}_{i,j} \boldsymbol{\phi}_{i,j}^\top$
  $i_r^*, j_r^* = \text{argmax}_{(i,j) \in [K] \times [K]} ||\boldsymbol{\phi}_{i,j}||_{\mathbf{V}(\pi_r)^{-1}}$
  $g_r = ||\boldsymbol{\phi}_{i_r^*, j_r^*}||_{\mathbf{V}(\pi_r)^{-1}}$
  $\gamma_r = \frac{g_r - 1/d}{g_r - 1}$
  $\pi_{r+1}(i,j) = (1 - \gamma_r)\pi_r(i,j) + \gamma_r \mathbb{1}(i_r^* = i)\mathbb{1}(j_r^* = j)$
end for
Output: Approximate G-optimal design solution $\pi_{R+1}(i,j)$

Chapter 7

Variance-Aware Contextual Dueling Bandits

7.1 Introduction

Intuitively, the variance of the noise in the feedback signal determines the difficulty of the problem. To illustrate, consider an extreme case, where the feedback of a linear contextual bandit is noiseless (i.e., the variance is zero). A learner can recover the underlying reward function precisely by exploring each dimension only once, and suffer a $\widetilde{O}(d)$ regret in total, where d is the dimension of the context vector. This motivates a series of works on establishing variance-aware regret bounds for multi-armed bandits, e.g. (Audibert et al., 2009; Mukherjee et al., 2017) and contextual bandits, e.g. (Zhou et al., 2021; Zhang et al., 2021; Kim et al., 2022; Zhao et al., 2023b,a). This observation also remains valid when applied to the dueling bandit scenario. In particular, the binary preferential feedback is typically assumed to adhere to a Bernoulli distribution, with the mean value denoted by p. The variance reaches its maximum when p is close to $1/2$, a situation that is undesirable in human feedback applications, as it indicates a high level of disagreement or indecision. Therefore, maintaining a low variance in comparisons is usually preferred, and variance-dependent dueling algorithms are desirable because they can potentially perform better than those algorithms that only have worst-case regret guarantees.

In this chapter, We propose a new algorithm, named `VACDB`, to obtain a variance-aware regret guarantee. This algorithm is built upon several innovative designs, including (1) adaptation of multi-layered estimators to generalized linear models where the mean and variance are coupled (i.e., Bernoulli distribution), (2) symmetric arm selection that naturally aligns with the actual reward maximization objective in dueling bandits.

We prove that our algorithm enjoys a variance-aware regret bound $\widetilde{O}\big(d\sqrt{\sum_{t=1}^{T}\sigma_t^2}+d\big)$, where σ_t is the variance of the comparison in round t. Our algorithm is computationally efficient and does not require any prior knowledge of the variance level, which is available in the dueling bandit scenario. In the deterministic case, our regret bound becomes $\widetilde{O}(d)$, showcasing a remarkable improvement over previous works. When the variances of the pairwise comparison are the same across different pairs of arms, our regret reduces to the worst-case regret of $\widetilde{O}\big(d\sqrt{T}\big)$, which matches the lower bound $\Omega(d\sqrt{T})$ proved in Bengs et al. (2022)

7.2 Related Work

It has been shown empirically that leveraging variance information in multi-armed bandit algorithms can enjoy performance benefits (Auer et al., 2002). In light of this, Audibert et al. (2009) proposed an algorithm, named `UCBV`, which is based on Bernstein's inequality equipped with empirical variance. It provided the first analysis of variance-aware algorithms, demonstrating an improved regret bound. `EUCBV` Mukherjee et al. (2017) is another variance-aware algorithm that employs an elimination strategy. It incorporates variance estimates to determine the confidence bounds of the arms. For linear bandits, Zhou et al. (2021) proposed

a Bernstein-type concentration inequality for self-normalized martingales and designed an algorithm named `Weighted OFUL`. This approach used a weighted ridge regression scheme, using variance to discount each sample's contribution to the estimator. In particular, they proved a variance-dependent regret upper bound, which was later improved by Zhou and Gu (2022). These two works assumed the knowledge of variance information. Without knowing the variances, Zhang et al. (2021) and Kim et al. (2022) obtained the variance-dependent regret bound by constructing variance-aware confidence sets. (Zhao et al., 2023b) proposed an algorithm named `MOR-UCB` with the idea of partitioning the observed data into several layers and grouping samples with similar variance into the same layer. A similar idea was used in Zhao et al. (2023a) to design a SupLin-type algorithm `SAVE`. It assigns collected samples to L layers according to their estimated variances, where each layer has twice the variance upper bound as the one at one level lower. In this way, for each layer, the estimated variance of one sample is at most twice as the others. Their algorithm is computationally tractable with a variance-dependent regret bound based on a Freedman-type concentration inequality and adaptive variance-aware exploration.

7.3 Problem Setup

In this work, we consider a preferential feedback model with contextual information. In this model, an agent learns through sequential interactions with its environment over a series of rounds indexed by t, where $t \in [T]$ and T is the total number of rounds. In each round t, the agent is presented with a finite set of alternatives, with each alternative being characterized by its associated feature in the contextual set $\mathcal{A}_t \subseteq \mathbb{R}^d$. Following the convention in bandit theory, we refer to these alternatives as *arms*. Both the number of alternatives and the contextual set $\mathcal{A}_t$ can vary with the round index t. Afterward, the agent selects a pair of arms, with features $(\mathbf{x}_t, \mathbf{y}_t)$ respectively. The environment then compares the two selected arms and returns a stochastic feedback o_t, which takes a value from the set $\{0, 1\}$. This feedback informs the agent which arm is preferred: When $o_t = 1$ (resp. $o_t = 0$), the arm with feature $\mathbf{x}_t$ (resp. $\mathbf{y}_t$) wins.

We assume that stochastic feedback o_t follows a Bernoulli distribution, where the expected value p_t is determined by a generalized linear model (GLM). To be more specific, let $\mu(\cdot)$ be a fixed link function that is increasing monotonically and satisfies $\mu(x) + \mu(-x) = 1$. We assume the existence of an *unknown* parameter $\boldsymbol{\theta}^* \in \mathbb{R}^d$ which generates the preference probability when two contextual vectors are given, i.e.

$$\mathbb{P}(o_t = 1) = \mathbb{P}(\text{arm with } \mathbf{x}_t \text{ is preferred over arm with } \mathbf{y}_t) = p_t = \mu((\mathbf{x}_t - \mathbf{y}_t)^\top \boldsymbol{\theta}^*).$$

This model is the same as the linear stochastic transitivity (LST) model in Bengs et al. (2022), which includes the Bradley-Terry-Luce (BTL) model (Hunter, 2004; Luce, 1959), Thurstone-Mosteller model (Thurstone, 1927) and the exponential noise model as special examples. Please refer to Bengs et al. (2022) for details. The preference model studied in Saha (2021) can be treated as a special case where the link function is logistic.

We make the assumption on the boundness of the true parameter $\boldsymbol{\theta}^*$ and the feature vector.

Assumption 7.3.1. $\|\boldsymbol{\theta}^*\|_2 \leq 1$. There exists a constant $A > 0$ such that for all $t \in [T]$ and all $\mathbf{x} \in \mathcal{A}_t$, $\|\mathbf{x}\|_2 \leq A$.

Additionally, we make the following assumption on the link function μ, which is common in the study of generalized linear contextual bandits (Filippi et al., 2010; Li et al., 2017).

Assumption 7.3.2. The link function μ is differentiable. Furthermore, the first derivative $\dot{\mu}$ satisfies $\kappa_\mu \leq \dot{\mu}(\cdot) \leq L_\mu$ for some constants $L_\mu, \kappa_\mu > 0$.

We define the random noise $\epsilon_t = o_t - p_t$. Since the stochastic feedback o_t adheres to the Bernoulli distribution with expected value p_t, $\epsilon_t \in \{-p_t, 1 - p_t\}$. From the definition of ϵ_t, we can see that $|\epsilon_t| \leq 1$. Furthermore, we make the following assumptions:

$$\mathbb{E}[\epsilon_t | \mathbf{x}_{1:t}, \mathbf{y}_{1:t}, \epsilon_{1:t-1}] = 0, \mathbb{E}[\epsilon_t^2 | \mathbf{x}_{1:t}, \mathbf{y}_{1:t}, \epsilon_{1:t-1}] = \sigma_t^2.$$

Intuitively, σ_t reflects the difficulty associated with comparing the two arms:

- When p_t is around $1/2$, it suggests that the arms are quite similar, making the comparison challenging. Under this circumstance, the variance σ_t tends toward a constant, reaching a maximum value of $1/4$.
- On the contrary, as p_t approaches 0 or 1, it signals that one arm is distinctly preferable over the other, thus simplifying the comparison. In such scenarios, the variance σ_t decreases significantly toward 0.

The learning objective is to minimize the cumulative average regret defined as

$$\text{Regret}(T) = \frac{1}{2}\sum_{t=1}^{T}\left[2\mathbf{x}_t^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_t + \mathbf{y}_t)^\top\boldsymbol{\theta}^*\right], \tag{7.3.1}$$

where $\mathbf{x}_t^* = \arg\max_{\mathbf{x}\in\mathcal{A}_t} \mathbf{x}^\top\boldsymbol{\theta}^*$ is the contextual/feature vector of the optimal arm in round t. This definition is the same as the average regret studied in (Saha, 2021; Bengs et al., 2022). Note that in Bengs et al. (2022), besides the average regret, they also studied another type of regret, called weak regret. Since the weak regret is smaller than the average regret, the regret bound proved in our paper can immediately imply a regret bound defined by the weak regret.

7.4 Algorithm

7.4.1 Overview of the Algorithm

In this section, we present our algorithm named `VACDB` in Algorithm 7.1. Our algorithm shares a similar structure with `Sta'D` in Saha (2021) and `SupCoLSTIM` in Bengs et al. (2022). The core of our algorithm involves a sequential arm elimination process: from Line 6 to Line 18, our algorithm conducts arm selection with a layered elimination procedure. Arms are progressively eliminated across layers, with increased exploration precision in the subsequent layers. Starting at layer $\ell = 1$, our algorithm incorporates a loop comprising three primary conditional phases: Exploitation (Lines 7-9), Elimination (Lines 10-12) and Exploration (Lines 14-16). When all arm pairs within a particular layer have low uncertainty, the elimination procedure begins, dropping the arms with suboptimal estimated values. This elimination process applies an adaptive bonus radius based on variance information. A more comprehensive discussion can be found in Section 7.4.3. Subsequently, it advances to a higher layer, where exploration is conducted over the eliminated set. Upon encountering a layer with arm pairs of higher uncertainty than desired, our algorithm explores them and receives the feedback. Once comprehensive exploration has been achieved across layers and the uncertainty for all remaining arm pairs is small enough, our algorithm leverages the estimated parameters in the last layer to select the best arm from the remaining arms. For a detailed discussion of the selection policy, please refer to Section 7.4.4. After arm selection in the exploration phase, the estimator of the current layer is updated (Lines 19-22) using the regularized MLE, which will be discussed in more details in Section 7.4.2. Note that our algorithm maintains an index set $\Psi_{t,\ell}$ for each layer, comprising all rounds before round t when the algorithm conducts exploration in layer ℓ. As a result, for each exploration step, only one of the estimators $\widehat{\boldsymbol{\theta}}_{t,\ell}$ needs to be updated. Furthermore, our algorithm updates the covariance matrix $\widehat{\boldsymbol{\Sigma}}_{t,\ell}$ used to estimate uncertainty (Line 19).

7.4.2 Regularized MLE

Most of the previous work adopted standard MLE techniques to maintain an estimator of $\boldsymbol{\theta}^*$ in the generalized linear bandit model (Filippi et al., 2010; Li et al., 2017), which requires an initial exploration phase to ensure a balanced input dataset across $\mathbb{R}^d$ for the MLE. In the dueling bandits setting, where the feedback in each round can be seen as a generalized linear reward, Saha (2021); Bengs et al. (2022) also applied a similar MLE in their algorithms. As a result, a random initial exploration phase is also inherited to ensure that the MLE equation has a unique solution. However, in our setting, where the decision set varies among rounds and is even arbitrarily decided by the environment, this initial exploration phase cannot be directly applied to control the minimum eigenvalue of the covariance matrix.

To resolve this issue, we introduce a regularized MLE for contextual dueling bandits, which is more well-behaved in the face of extreme input data and does not require an additional exploration phase at the

Algorithm 7.1 Variance-Aware Contextual Dueling Bandit (`VACDB`)

Require: $\alpha > 0$, $L \leftarrow \lceil \log_2(1/\alpha) \rceil$, κ_μ, L_μ.

Initialize: For $\ell \in [L]$, $\widehat{\boldsymbol{\Sigma}}_{1,\ell} \leftarrow 2^{-2\ell}\mathbf{I}$, $\widehat{\boldsymbol{\theta}}_{1,\ell} \leftarrow \mathbf{0}$, $\boldsymbol{\Psi}_{1,\ell} \leftarrow \emptyset$, $\widehat{\beta}_{1,\ell} \leftarrow 2^{-\ell}(1 + 1/\kappa_\mu)$

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

  Observe $\mathcal{A}_t$

  Let $\mathcal{A}_{t,1} \leftarrow \mathcal{A}_t$, $\ell \leftarrow 1$.

  while $\mathbf{x}_t, \mathbf{y}_t$ are not specified **do**

    if $\|\mathbf{x} - \mathbf{y}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \le \alpha$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{A}_{t,\ell}$ **then**

      Choose $\mathbf{x}_t, \mathbf{y}_t = \operatorname{argmax}_{\mathbf{x},\mathbf{y} \in \mathcal{A}_{t,\ell}} \left\{ (\mathbf{x} + \mathbf{y})^\top \widehat{\boldsymbol{\theta}}_{t,\ell} + \widehat{\beta}_{t,\ell} \|\mathbf{x} - \mathbf{y}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \right\}$

      and observe $o_t = \mathbb{1}(\mathbf{x}_t \succ \mathbf{y}_t)$ `//Exploitation (Lines 7-9)`

      Keep the same index sets at all layers: $\boldsymbol{\Psi}_{t+1,\ell'} \leftarrow \boldsymbol{\Psi}_{t,\ell'}$ for all $\ell' \in [L]$

    else if $\|\mathbf{x} - \mathbf{y}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \le 2^{-\ell}$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{A}_{t,\ell}$ **then**

      $\mathcal{A}_{t,\ell+1} \leftarrow \left\{ \mathbf{x} \in \mathcal{A}_{t,\ell} \mid \mathbf{x}^\top \widehat{\boldsymbol{\theta}}_{t,\ell} \ge \max_{\mathbf{x}' \in \mathcal{A}_{t,\ell}} \mathbf{x}'^\top \widehat{\boldsymbol{\theta}}_{t,\ell} - 2^{-\ell} \widehat{\beta}_{t,\ell} \right\}$

      $\ell = \ell + 1$ `//Elimination (Lines 10-12)`

    else

      Choose $\mathbf{x}_t, \mathbf{y}_t$ such that $\|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} > 2^{-\ell}$

      and observe $o_t = \mathbb{1}(\mathbf{x}_t \succ \mathbf{y}_t)$ `//Exploration (Lines 14-16)`

      Compute the weight $w_t \leftarrow 2^{-\ell} / \|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}}$

      Update the index sets $\boldsymbol{\Psi}_{t+1,\ell} \leftarrow \boldsymbol{\Psi}_{t,\ell} \cup \{t\}$ and $\boldsymbol{\Psi}_{t+1,\ell'} \leftarrow \boldsymbol{\Psi}_{t,\ell'}$ for all $\ell' \in [L]/\{\ell\}$

    end if

  end while

  For $\ell \in [L]$ such that $\boldsymbol{\Psi}_{t+1,\ell} \neq \boldsymbol{\Psi}_{t,\ell}$, update $\widehat{\boldsymbol{\Sigma}}_{t+1,\ell} \leftarrow \widehat{\boldsymbol{\Sigma}}_{t,\ell} + w_t^2 (\mathbf{x}_t - \mathbf{y}_t)(\mathbf{x}_t - \mathbf{y}_t)^\top$

  Calculate the MLE $\widehat{\boldsymbol{\theta}}_{t+1,\ell}$ by solving the equation:

$$2^{-2\ell}\kappa_\mu \boldsymbol{\theta} + \sum_{s \in \boldsymbol{\Psi}_{t+1,\ell}} w_s^2 \Big(\mu\big((\mathbf{x}_s - \mathbf{y}_s)^\top \boldsymbol{\theta}\big) - o_s \Big)(\mathbf{x}_s - \mathbf{y}_s) = \mathbf{0}$$

  Compute $\widehat{\beta}_{t+1,\ell}$ according to (7.4.3)

  For $\ell \in [L]$ such that $\boldsymbol{\Psi}_{t+1,\ell} = \boldsymbol{\Psi}_{t,\ell}$, let $\widehat{\boldsymbol{\Sigma}}_{t+1,\ell} = \widehat{\boldsymbol{\Sigma}}_{t,\ell}$, $\widehat{\boldsymbol{\theta}}_{t+1,\ell} \leftarrow \widehat{\boldsymbol{\theta}}_{t,\ell}, \widehat{\beta}_{t+1,\ell} \leftarrow \widehat{\beta}_{t,\ell}$

end for

starting rounds. Specifically, the regularized MLE is the solution of the following equation:

$$\lambda\boldsymbol{\theta} + \sum_{s} w_s^2 \Big(\mu\big((\mathbf{x}_s - \mathbf{y}_s)^\top \boldsymbol{\theta}\big) - o_s\Big)(\mathbf{x}_s - \mathbf{y}_s) = \mathbf{0}, \tag{7.4.1}$$

where we add the additional regularization term $\lambda\boldsymbol{\theta}$ to make sure that the estimator will change mildly. From the theoretical viewpoint, our proposed regularization term leads to a non-singularity guarantee for the covariance matrix. Additionally, we add some weights here to obtain a tighter concentration inequality. Concretely, with a suitable choice of the parameters in each layer and a Freedman-type inequality first introduced in Zhao et al. (2023a), we can prove a concentration inequality for the estimator in the ℓ-th layer:

$$\left\|\boldsymbol{\theta}^* - \widehat{\boldsymbol{\theta}}_{t,\ell}\right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}} \leq \frac{2^{-\ell}}{\kappa_\mu}\left[16\sqrt{\textstyle\sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2\sigma_s^2 \log(4t^2L/\delta)} + 6\log(4t^2L/\delta)\right] + 2^{-\ell}. \tag{7.4.2}$$

This upper bound scales with $2^{-\ell}$, which arises from our choice of the weights.

The regularized MLE can be formulated as a finite-sum offline optimization problem. For many widely used models, such as the Bradley-Terry-Luce (BTL) model (Hunter, 2004; Luce, 1959), the regularized MLE is a strongly convex and smooth optimization problem. We can solve it using accelerated gradient descent (Nesterov, 2003) and SVRG (Johnson and Zhang, 2013), both of which achieve a linear rate of convergence. This can mitigate the scalability issues caused by the increasing number of iterations. The regularized MLE can also be solved by an online learning algorithm such as in Jun et al. (2017) and Zhao et al. (2023b), where additional effort is required for the analysis.

7.4.3 Multi-layer Structure with Variance-Aware Confidence Radius

Due to the multi-layered structure of our algorithm, the construction of the confidence set is of paramount importance. Our algorithm distinguishes itself from prior multi-layered algorithms (Saha, 2021; Bengs et al., 2022) primarily through a variance-aware adaptive selection of the confidence radius, which helps to achieve a variance-aware regret bound. Intuitively, we should choose the confidence radius $\widehat{\beta}_{t,\ell}$ based on the concentration inequality (7.4.2). However, it depends on the true variance σ_s, of which we do not have prior knowledge. To address this issue, we estimate it using the estimator $\widehat{\boldsymbol{\theta}}_{t,\ell}$. We choose

$$\begin{aligned}\widehat{\beta}_{t,\ell} := {} & \frac{16\cdot 2^{-\ell}}{\kappa_\mu}\sqrt{\Big(8\widehat{\mathrm{Var}}_{t,\ell} + 18\log(4(t+1)^2L/\delta)\Big)\log(4t^2L/\delta)} \\ & + \frac{6\cdot 2^{-\ell}}{\kappa_\mu}\log(4t^2L/\delta) + 2^{-\ell+1},\end{aligned} \tag{7.4.3}$$

where

$$\widehat{\mathrm{Var}}_{t,\ell} := \begin{cases} \sum_{s\in\Psi_{t,\ell}} w_s^2\Big(o_s - \mu((\mathbf{x}_s - \mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell})\Big)^2, & 2^\ell \geq 64(L_\mu/\kappa_\mu)\sqrt{\log(4(t+1)^2L/\delta)}, \\ |\Psi_{t,\ell}|, & \text{otherwise.}\end{cases}$$

The varied selections of $\widehat{\mathrm{Var}}_{t,\ell}$ arise from the fact that our variance estimator becomes more accurate at higher layers. For those low layers, we employ the natural upper bound $\sigma_i \leq 1$. Note that this situation arises only $\Theta(\log\log(T/\delta))$ times, which is a small portion of the total layers $L = \Theta(\log T)$. In our proof, we deal with two cases separately. Due to the limited space available here, the full proof can be found in Section 7.5.2.

7.4.4 Symmetric Arm Selection

In this subsection, we focus on the arm selection policy described in Line 9. To our knowledge, this policy is new and has never been studied in prior work for the (generalized) linear dueling bandit problem. In detail, suppose that we have an estimator $\widehat{\boldsymbol{\theta}}_t$ in round t that lies in a high probability confidence set:

$$\big\{\boldsymbol{\theta} : \big\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\big\|_{\widehat{\boldsymbol{\Sigma}}_t} \leq \beta_t\big\},$$

where $\widehat{\mathbf{\Sigma}}_t = \lambda \mathbf{I} + \sum_{i=1}^{t-1} (\mathbf{x}_i - \mathbf{y}_i)(\mathbf{x}_i - \mathbf{y}_i)^\top$. Our choice of arms can be written as

$$\mathbf{x}_t, \mathbf{y}_t = \underset{\mathbf{x},\mathbf{y} \in \mathcal{A}_t}{\operatorname{argmax}} \left[(\mathbf{x} + \mathbf{y})^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x} - \mathbf{y}\|_{\widehat{\mathbf{\Sigma}}_t^{-1}} \right]. \tag{7.4.4}$$

Intuitively, we utilize $(\mathbf{x}+\mathbf{y})^\top \widehat{\boldsymbol{\theta}}_t$ as the estimated score and incorporate an exploration bonus dependent on $\|\mathbf{x} - \mathbf{y}\|_{\widehat{\mathbf{\Sigma}}_t^{-1}}$. Our symmetric selection of arms aligns with the nature of dueling bandits where the order of arms does not matter. Here we compare it with several alternative arm selection criteria that have appeared in previous works.

The `MaxInP` algorithm in Saha (2021) builds the so-called "promising" set that includes the optimal arm:

$$\mathcal{C}_t = \left\{ \mathbf{x} \in \mathcal{A}_t \mid (\mathbf{x} - \mathbf{y})^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x} - \mathbf{y}\|_{\widehat{\mathbf{\Sigma}}_t^{-1}} \geq 0, \forall \mathbf{y} \in \mathcal{A}_t \right\}.$$

It chooses the symmetric arm pair from the set $\mathcal{C}_t$ that has the highest pairwise score variance (maximum informative pair), i.e.,

$$\mathbf{x}_t, \mathbf{y}_t = \underset{\mathbf{x},\mathbf{y} \in \mathcal{C}_t}{\operatorname{argmax}} \|\mathbf{x} - \mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}}.$$

The `Sta'D` algorithm in Saha (2021) uses an asymmetric arm selection criterion, which selects the first arm with the highest estimated score, i.e.,

$$\mathbf{x}_t = \underset{\mathbf{x} \in \mathcal{A}_t}{\operatorname{argmax}} \, \mathbf{x}^\top \widehat{\boldsymbol{\theta}}_t.$$

Following this, it selects the second arm as the toughest competitor to the arm $\mathbf{x}_t$, with a bonus term related to $\|\mathbf{x}_t - \mathbf{y}\|_{\Sigma_t^{-1}}$, i.e.,

$$\mathbf{y}_t = \underset{\mathbf{y} \in \mathcal{A}_t}{\operatorname{argmax}} \, \mathbf{y}^\top \widehat{\boldsymbol{\theta}}_t + 2\beta_t \|\mathbf{x}_t - \mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}}. \tag{7.4.5}$$

Similar arm selection criterion has also been used in the `CoLSTIM` algorithm (Bengs et al., 2022). We can show that these two alternative arm selection policies result in comparable regret decomposition and can establish similar regret upper bound.

We assume that in round t, we have an estimator $\widehat{\boldsymbol{\theta}}_t$, a covariance matrix $\mathbf{\Sigma}_t = \lambda \mathbf{I} + \sum_{i=1}^{t-1} (\mathbf{x}_i - \mathbf{y}_i)(\mathbf{x}_i - \mathbf{y}_i)^\top$ and a concentration inequality with confidence radius β_t,

$$\|\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^*\|_{\mathbf{\Sigma}_t} \leq \beta_t. \tag{7.4.6}$$

The three arm selection methods can be described as follows:

Method 1: Following Saha (2021), let $\mathcal{C}_t$ be

$$\mathcal{C}_t = \{ \mathbf{x} \in \mathcal{A}_t \mid (\mathbf{x} - \mathbf{y})^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x} - \mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}} \geq 0, \forall \mathbf{y} \in \mathcal{A}_t \}.$$

Then $\mathbf{x}_t^* \in \mathcal{C}_t$ because for any $\mathbf{y} \in \mathcal{A}_t$

$$\begin{aligned}
(\mathbf{x}_t^* - \mathbf{y})^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x}_t^* - \mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}} &= (\mathbf{x}_t^* - \mathbf{y})^\top (\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^*) + (\mathbf{x}_t^* - \mathbf{y})^\top \boldsymbol{\theta}^* + \beta_t \|\mathbf{x}_t^* - \mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}} \\
&\geq \beta_t \|\mathbf{x}_t^* - \mathbf{y}\|_{\Sigma_t^{-1}} - \|\mathbf{x}_t^* - \mathbf{y}\|_{\Sigma_t^{-1}}^\top \|\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^*\|_{\mathbf{\Sigma}_t} \\
&\geq 0,
\end{aligned}$$

where the first inequality holds due to Cauchy-Schwarz inequality and $\mathbf{x}_t^*$ is the optimal arm in round t. The second inequality holds due to (7.4.6).

The arms selected in round t are $\mathbf{x}_t, \mathbf{y}_t = \operatorname{argmax}_{\mathbf{x},\mathbf{y}\in\mathcal{C}_t} \|\mathbf{x}-\mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}}$ Then the regret in round t can be decomposed as

$$\begin{aligned}
2r_t &= 2\mathbf{x}_t^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_t+\mathbf{y}_t)^\top\boldsymbol{\theta}^* \\
&= (\mathbf{x}_t^*-\mathbf{x}_t)^\top\boldsymbol{\theta}^* + (\mathbf{x}_t^*-\mathbf{y}_t)^\top\boldsymbol{\theta}^* \\
&= (\mathbf{x}_t^*-\mathbf{x}_t)^\top(\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t) + (\mathbf{x}_t^*-\mathbf{x}_t)^\top\widehat{\boldsymbol{\theta}}_t + (\mathbf{x}_t^*-\mathbf{y}_t)^\top(\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t) + (\mathbf{x}_t^*-\mathbf{y}_t)^\top\widehat{\boldsymbol{\theta}}_t \\
&\le (\mathbf{x}_t^*-\mathbf{x}_t)^\top(\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t) + \beta_t\|\mathbf{x}_t^*-\mathbf{x}_t\|_{\mathbf{\Sigma}_t^{-1}} + (\mathbf{x}_t^*-\mathbf{y}_t)^\top(\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t) + \beta_t\|\mathbf{x}_t^*-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}} \\
&\le \|\mathbf{x}_t^*-\mathbf{x}_t\|_{\mathbf{\Sigma}_t^{-1}}\|\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t\|_{\mathbf{\Sigma}_t} + \beta_t\|\mathbf{x}_t^*-\mathbf{x}_t\|_{\mathbf{\Sigma}_t^{-1}} \\
&\quad + \|\mathbf{x}_t^*-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}}\|\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t\|_{\mathbf{\Sigma}_t} + \beta_t\|\mathbf{x}_t^*-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}} \\
&\le 2\beta_t\|\mathbf{x}_t^*-\mathbf{x}_t\|_{\mathbf{\Sigma}_t^{-1}} + 2\beta_t\|\mathbf{x}_t^*-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}} \\
&\le 4\beta_t\|\mathbf{x}_t-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}},
\end{aligned}$$

where the first inequality holds because the choice $\mathbf{x}_t, \mathbf{y}_t \in \mathcal{C}_t$. The second inequality holds due to Cauchy-Schwarz inequality. The third inequality holds due to (7.4.6). The last inequality holds due to $\mathbf{x}_t^* \in \mathcal{C}_t, \mathbf{x}_t, \mathbf{y}_t = \operatorname{argmax}_{\mathbf{x},\mathbf{y}\in\mathcal{C}_t} \|\mathbf{x}-\mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}}$.

Method 2: Following Bengs et al. (2022), we choose the first arm as

$$\mathbf{x}_t = \operatorname*{argmax}_{\mathbf{x}\in\mathcal{A}_t} \mathbf{x}^\top\widehat{\boldsymbol{\theta}}_t.$$

Then choose the second arm as

$$\mathbf{y}_t = \operatorname*{argmax}_{\mathbf{y}\in\mathcal{A}_t} \mathbf{y}^\top\widehat{\boldsymbol{\theta}}_t + 2\beta_t\|\mathbf{x}_t-\mathbf{y}\|_{\mathbf{\Sigma}_t^{-1}},$$

The regret in round t can be decomposed as

$$\begin{aligned}
2r_t &= 2\mathbf{x}_t^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_t+\mathbf{y}_t)^\top\boldsymbol{\theta}^* \\
&= 2(\mathbf{x}_t^*-\mathbf{x}_t)^\top\boldsymbol{\theta}^* + (\mathbf{x}_t-\mathbf{y}_t)^\top\boldsymbol{\theta}^* \\
&= 2(\mathbf{x}_t^*-\mathbf{x}_t)^\top(\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t) + 2(\mathbf{x}_t^*-\mathbf{x}_t)^\top\widehat{\boldsymbol{\theta}}_t + (\mathbf{x}_t-\mathbf{y}_t)^\top(\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t) + (\mathbf{x}_t-\mathbf{y}_t)^\top\widehat{\boldsymbol{\theta}}_t \\
&\le 2\|\mathbf{x}_t^*-\mathbf{x}_t\|_{\mathbf{\Sigma}_t^{-1}}\|\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t\|_{\mathbf{\Sigma}_t} + (\mathbf{x}_t^*-\mathbf{x}_t)^\top\widehat{\boldsymbol{\theta}}_t \\
&\quad + \|\mathbf{x}_t-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}}\|\boldsymbol{\theta}^*-\widehat{\boldsymbol{\theta}}_t\|_{\mathbf{\Sigma}_t} + (\mathbf{x}_t-\mathbf{y}_t)^\top\widehat{\boldsymbol{\theta}}_t \\
&\le 2\beta_t\|\mathbf{x}_t^*-\mathbf{x}_t\|_{\mathbf{\Sigma}_t^{-1}} + (\mathbf{x}_t^*-\mathbf{y}_t)^\top\widehat{\boldsymbol{\theta}}_t + \beta_t\|\mathbf{x}_t-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}} \\
&\le \mathbf{y}_t^\top\widehat{\boldsymbol{\theta}}_t + 2\beta_t\|\mathbf{x}_t-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}} - \mathbf{x}_t^{*\top}\widehat{\boldsymbol{\theta}}_t + (\mathbf{x}_t^*-\mathbf{y}_t)^\top\widehat{\boldsymbol{\theta}}_t + \beta_t\|\mathbf{x}_t-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}} \\
&= 3\beta_t\|\mathbf{x}_t-\mathbf{y}_t\|_{\mathbf{\Sigma}_t^{-1}},
\end{aligned}$$

where the first inequality holds due to the Cauchy-Schwarz inequality and $\mathbf{x}_t^\top\widehat{\boldsymbol{\theta}}_t \ge \mathbf{x}_t^*\widehat{\boldsymbol{\theta}}_t$. The second inequality holds due to the Cauchy-Schwarz inequality. The third inequality holds due to $\mathbf{y}_t = \operatorname{argmax}_{\mathbf{y}\in\mathcal{A}_t} \mathbf{y}^\top\widehat{\boldsymbol{\theta}}_t + 2\beta_t\|\mathbf{x}_t-\mathbf{y}\|_{\Sigma_t^{-1}}$.

Method 3: In this method, we choose two arms as

$$\mathbf{x}_t, \mathbf{y}_t = \operatorname*{argmax}_{\mathbf{x},\mathbf{y}\in\mathcal{A}_t} \left[(\mathbf{x}+\mathbf{y})^\top\widehat{\boldsymbol{\theta}}_t + \beta_t\|\mathbf{x}-\mathbf{y}\|_{\widehat{\mathbf{\Sigma}}_t^{-1}}\right] \tag{7.4.7}$$

Then the regret can be decomposed as

$$\begin{aligned}
2r_t &= 2\mathbf{x}_t^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_t + \mathbf{y}_t)^\top \boldsymbol{\theta}^* \\
&= (\mathbf{x}_t^* - \mathbf{x}_t)^\top \boldsymbol{\theta}^* + (\mathbf{x}_t^* - \mathbf{y}_t)^\top \boldsymbol{\theta}^* \\
&= (\mathbf{x}_t^* - \mathbf{x}_t)^\top (\boldsymbol{\theta}^* - \widehat{\boldsymbol{\theta}}_t) + (\mathbf{x}_t^* - \mathbf{y}_t)^\top (\boldsymbol{\theta}^* - \widehat{\boldsymbol{\theta}}_t) + (2\mathbf{x}_t^* - \mathbf{x}_t - \mathbf{y}_t)^\top \widehat{\boldsymbol{\theta}}_t \\
&\leq \|\mathbf{x}_t^* - \mathbf{x}_t\|_{\boldsymbol{\Sigma}_t^{-1}} \|\boldsymbol{\theta}^* - \widehat{\boldsymbol{\theta}}_t\|_{\boldsymbol{\Sigma}_t} + \|\mathbf{x}_t^* - \mathbf{y}_t\|_{\boldsymbol{\Sigma}_t^{-1}} \|\boldsymbol{\theta}^* - \widehat{\boldsymbol{\theta}}_t\|_{\boldsymbol{\Sigma}_t} + (2\mathbf{x}_t^* - \mathbf{x}_t - \mathbf{y}_t)^\top \widehat{\boldsymbol{\theta}}_t \\
&\leq \beta_t \|\mathbf{x}_t^* - \mathbf{x}_t\|_{\boldsymbol{\Sigma}_t^{-1}} + \beta_t \|\mathbf{x}_t^* - \mathbf{y}_t\|_{\boldsymbol{\Sigma}_t^{-1}} + (2\mathbf{x}_t^* - \mathbf{x}_t - \mathbf{y}_t)^\top \widehat{\boldsymbol{\theta}}_t,
\end{aligned}$$

where the first inequality holds due to the Cauchy-Schwarz inequality. The second inequality holds due to (7.4.6). Using (7.4.7), we have

$$\begin{aligned}
(\mathbf{x}_t^* + \mathbf{x}_t)^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x}_t^* - \mathbf{x}_t\|_{\widehat{\boldsymbol{\Sigma}}_t^{-1}} &\leq (\mathbf{x}_t + \mathbf{y}_t)^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_t^{-1}} \\
(\mathbf{x}_t^* + \mathbf{y}_t)^\top \widehat{\boldsymbol{\theta}}_{t,\ell} + \beta_t \|\mathbf{x}_t^* - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_t^{-1}} &\leq (\mathbf{x}_t + \mathbf{y}_t)^\top \widehat{\boldsymbol{\theta}}_t + \beta_t \|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_t^{-1}}.
\end{aligned}$$

Adding the above two inequalities, we have

$$\beta_t \|\mathbf{x}_t^* - \mathbf{x}_t\|_{\boldsymbol{\Sigma}_t^{-1}} + \beta_t \|\mathbf{x}_t^* - \mathbf{y}_t\|_{\boldsymbol{\Sigma}_t^{-1}} \leq (\mathbf{x}_t + \mathbf{y}_t - 2\mathbf{x}_t^*)^\top \widehat{\boldsymbol{\theta}}_t + 2\beta_t \|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_t^{-1}}.$$

Therefore, we prove that the regret can be upper bounded by

$$2r_t \leq 2\beta_t \|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}_t^{-1}}.$$

In conclusion, we can prove similar inequalities for the above three arm selection policies. To get an upper bound of regret, we can sum up the instantaneous regret in each round and use Lemma 7.7.1 to obtain the final result.

7.5 Variance-aware Regret Bound

In this section, we summarize our main results in the following theorem.

Theorem 7.5.1. If we set $\alpha = 1/(T^{3/2})$, then with probability at least $1 - 2\delta$, the regret of Algorithm 7.1 is bounded as

$$\text{Regret}(T) = \widetilde{O}\left(\frac{d}{\kappa_\mu}\sqrt{\sum_{t=1}^{T}\sigma_t^2} + d\Big(\frac{L_\mu^2}{\kappa_\mu^2} + \frac{1}{\kappa_\mu}\Big)\right).$$

This regret can be divided into two parts, corresponding to the regret incurred from the exploration steps (Line 14) and the exploitation steps (Line 8). The exploitation-induced regret is always $\widetilde{O}(1)$ as shown in (7.5.1), and thus omitted by the big-O notation. The total regret is dominated by the exploration-induced regret, which mainly depends on the total variance $\sum_{t=1}^{T}\sigma_t^2$. Note that the comparisons during the exploration steps only happen between non-identical arms ($\mathbf{x}_t \neq \mathbf{y}_t$).

Remark 7.5.2. To show the advantage of variance awareness, consider the extreme case where the comparisons are deterministic. More specifically, for any two arms with contextual vectors $\mathbf{x}$ and $\mathbf{y}$, the comparison between arm $\mathbf{x}$ and item $\mathbf{y}$ is determined by $o_t = \mathbb{1}\{\mathbf{x}_t^\top \boldsymbol{\theta}^* > \mathbf{y}_t^\top \boldsymbol{\theta}^*\}$, and thus has zero variance. Our algorithm can account for the zero variance, and the regret becomes $\widetilde{O}(d)$, which is optimal since recovering the parameter $\boldsymbol{\theta}^* \in \mathbb{R}^d$ requires exploring each dimension.

Remark 7.5.3. The setting we study is quite general, where the arm set is time-varying, and therefore, the variance of arms can vary with respect to time and arms. When we restrict our setting to a special case with uniform variances for all pairwise comparisons, i.e., $\sigma_t^2 = \sigma^2$ for all t, our upper bound becomes $\widetilde{O}(\sigma d\sqrt{T})$. This results in a regret bound that does not depend on the random variable σ_t^2.

Remark 7.5.4. In the worst-case scenario, the variance of the arm comparison is upper bounded by $1/4$, our regret upper bound becomes $\widetilde{O}(d\sqrt{T})$, which matches the regret lower bound $\Omega(d\sqrt{T})$ for dueling bandits with exponentially many arms proved in Bengs et al. (2022), up to logarithmic factors. This regret bound also recovers the regret bounds of `MaxInP` (Saha, 2021) and `CoLSTIM` (Bengs et al., 2022). Compared with `Sta'D` (Saha, 2021) and `SupCoLSTIM` (Bengs et al., 2022), our regret bound is on par with their regret bounds provided the number of arms K is large. More specifically, their regret upper bounds are $\widetilde{O}(\sqrt{dT\log K})$. When K is exponential in d, their regret bound becomes $\widetilde{O}(d\sqrt{T})$, which is of the same order as our regret bound.

Remark 7.5.5. Notably, in Bengs et al. (2022), they made an assumption that the context vectors can span the total d-dimensional Euclidean space, which is essential in their initial exploration phase. In our work, we replace the initial exploration phase with a regularizer, thus relaxing their assumption.

7.5.1 Proof Sketch of Theorem 7.5.1

As we describe in Section 7.4, the arm selection is specified in two places, the exploration part (Lines 14 - 16) and the exploitation part (Lines 8 - 9). Given the update rule of the index set, each step within the exploration part will be included by the final index set $\Psi_{T+1,\ell}$ of a singular layer ℓ. Conversely, steps within the exploitation part get into $T/\cup_{\ell\in[L]}\Psi_{T+1,\ell}$. Using this division, we can decompose the regret into :

$$\begin{aligned}\mathrm{Regret}(T) = \frac{1}{2}\Bigg[&\underbrace{\textstyle\sum_{s\in[T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})}\Big(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\Big)}_{\text{exploitation}}\\ &+\underbrace{\textstyle\sum_{\ell\in[L]}\sum_{s\in\Psi_{T+1,\ell}}\Big(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\Big)}_{\text{exploration}}\Bigg].\end{aligned}$$

We bound the incurred regret of each part separately.

For any round $s \in T/\cup_{\ell\in[L]}\Psi_{T+1,\ell}$, the given condition for exploitation indicates the existence of a layer ℓ_s such that $\|\mathbf{x}_s - \mathbf{y}_s\|_{\widehat{\boldsymbol{\Sigma}}_{s,\ell}^{-1}} \le \alpha$ for all $\mathbf{x}_s, \mathbf{y}_s \in \mathcal{A}_{s,\ell}$. Using the Cauchy inequality and the MLE described in Section 7.4.2, we can show that the regret incurred in round s is smaller than $3\widehat{\beta}_{s,\ell_s}\cdot\alpha$. Considering the simple upper bound $\widehat{\beta}_{s,\ell_s} \le \widetilde{O}(\sqrt{T})$ and $\alpha = T^{-3/2}$, the regret for one exploitation round does not exceed $\widetilde{O}(1/T)$. Consequently, the cumulative regret is

$$\textstyle\sum_{s\in[T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})}\Big(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\Big) \le \widetilde{O}(1)., \tag{7.5.1}$$

which is a low-order term in total regret.

In the exploration part, the regret is the cumulative regret encountered within each layer. We analyze the low layers and high layers distinctly. For $\ell \le \ell^* = \Big\lceil \log_2\Big(64(L_\mu/\kappa_\mu)\sqrt{\log(4(T+1)^2L/\delta)}\Big)\Big\rceil$, the incurred regret can be upper bounded by the number of rounds in this layer

$$\textstyle\sum_{s\in\Psi_{T+1,\ell}}\big(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\big) \le 4|\Psi_{T+1,\ell}|.$$

Moreover, $|\Psi_{T+1,\ell}|$ can be upper bounded by

$$|\Psi_{T+1,\ell}| \le 2^{2\ell}d\log\left(1+2^{2\ell}AT/d\right) \le O\Bigg(\frac{L_\mu^2}{\kappa_\mu^2}d\log\left(1+2^{2\ell^*}AT/d\right)\log\left(4(T+1)^2L/\delta\right)\Bigg). \tag{7.5.2}$$

Thus the total regret for layers $\ell \le \ell^*$ is bounded by $\widetilde{O}(d)$. For $\ell > \ell^*$, we can bound the cumulative regret incurred in each layer with

Lemma 7.5.6. With high probability, for all $\ell \in [L]\setminus\{1\}$, the regret incurred by the index set $\Psi_{T+1,\ell}$ is bounded by

$$\textstyle\sum_{s\in\Psi_{T+1,\ell}}\Big(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - \big(\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*\big)\Big) \le \widetilde{O}\Big(d\cdot 2^\ell\widehat{\beta}_{T,\ell-1}\Big).$$

By summing up the regret of all the layers, we can upper bound the total regret for layers $\ell > \ell^*$ as

$$\sum_{\ell \in [L]/[\ell^*]} \sum_{s \in \Psi_{T+1,\ell}} \left(2\mathbf{x}_s^{*\top} \boldsymbol{\theta}^* - \left(\mathbf{x}_s^\top \boldsymbol{\theta}^* + \mathbf{y}_s^\top \boldsymbol{\theta}^* \right) \right) \leq \widetilde{O}\left(\frac{d}{\kappa_\mu} \sqrt{\sum_{t=1}^T \sigma_t^2} + \frac{d}{\kappa_\mu} \right),$$

We can complete the proof of Theorem 7.5.1 by combining the regret in different parts together.

7.5.2 Proof of Theorem 7.5.1

We first need the concentration inequality for the MLE.

Lemma 7.5.7. With probability at least $1-\delta$, the following concentration inequality holds for all round $t \geq 2$ and layer $\ell \in [L]$ simultaneously:

$$\left\| \widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^* \right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}} \leq \frac{2^{-\ell}}{\kappa_\mu} \left[16 \sqrt{\sum_{s \in \boldsymbol{\Psi}_{t,\ell}} w_s^2 \sigma_s^2 \log(4t^2 L/\delta)} + 6 \log(4t^2 L/\delta) \right] + 2^{-\ell}.$$

With this lemma, we have the following event holds with high probability:

$$\mathcal{E} = \left\{ \left\| \widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^* \right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}} \leq \frac{2^{-\ell}}{\kappa_\mu} \left[16 \sqrt{\sum_{s \in \boldsymbol{\Psi}_{t,\ell}} w_s^2 \sigma_s^2 \log(4t^2 L/\delta)} + 6 \log(4t^2 L/\delta) \right] + 2^{-\ell} \text{ for all } t, \ell \right\}.$$

Lemma 7.5.7 shows that $\mathbb{P}[\mathcal{E}] \geq 1-\delta$. For our choice of $\widehat{\beta}_{t,\ell}$ defined in (7.4.3), we define the following event:

$$\mathcal{E}^{\text{bonus}} = \left\{ \widehat{\beta}_{t,\ell} \geq \frac{2^{-\ell}}{\kappa} \left[16 \sqrt{\sum_{s \in \boldsymbol{\Psi}_{t,\ell}} w_s^2 \sigma_s^2 \log(4t^2 L/\delta)} + 6 \log(4t^2 L/\delta) \right] + 2^{-\ell}, \text{ for all } t, \ell \right\}.$$

The following two lemmas show that the event $\mathcal{E}_\ell^{\text{bonus}}$ holds with high probability.

Lemma 7.5.8. With probability at least $1-\delta$, for all $t \geq 2$, $\ell \in [L]$, the following two inequalties hold simultaneously.

$$\sum_{s \in \Psi_{t,\ell}} w_s^2 \sigma_s^2 \leq 2 \sum_{s \in \Psi_{t,\ell}} w_s^2 \epsilon_s^2 + \frac{14}{3} \log(4t^2 L/\delta).$$

$$\sum_{s \in \Psi_{t,\ell}} w_s^2 \epsilon_s^2 \leq \frac{3}{2} \sum_{s \in \Psi_{t,\ell}} w_s^2 \sigma_s^2 + \frac{7}{3} \log(4t^2 L/\delta).$$

Lemma 7.5.9. Suppose that the inequalities in Lemma 7.5.8 and the event $\mathcal{E}$ hold. For all $t \geq 2$ and $\ell \in [L]$ such that $2^\ell \geq 64(L_\mu/\kappa_\mu)\sqrt{\log(4(T+1)^2 L/\delta)}$, the following inequalities hold

$$\sum_{s \in \Psi_{t,\ell}} w_s^2 \sigma_s^2 \leq 8 \sum_{s \in \Psi_{t,\ell}} w_s^2 \left(o_s - \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell} \right) \right)^2 + 18 \log(4(t+1)^2 L/\delta).$$

$$\sum_{s \in \Psi_{t,\ell}} w_s^2 \left(o_s - \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell} \right) \right)^2 \leq 4 \sum_{s \in \Psi_{t,\ell}} w_s^2 \sigma_s^2 + 8 \log(4(t+1)^2 L/\delta).$$

Recall that with our choice of $\widehat{\beta}_{t,\ell}$ in (7.4.3), the inequality in $\mathcal{E}^{\text{bonus}}$ holds naturally when $2^\ell < 64(L_\mu/\kappa_\mu)\sqrt{\log(4(T+1)^2 L/\delta)}$. Combining Lemma 7.5.8, Lemma 7.5.9 and $\mathbb{P}[\mathcal{E}] \geq 1-\delta$, after taking a union bound, we have proved $\mathbb{P}[\mathcal{E}^{\text{bonus}} \cap \mathcal{E}] \geq 1-2\delta$.

Lemma 7.5.10. Suppose the high probability events $\mathcal{E}^{\text{bonus}}$ and $\mathcal{E}$ holds. Then for all $t \geq 1$ and $\ell \in [L]$ such that the set $\mathcal{A}_{t,\ell}$ is defined, the contextual vector of the optimal arm $\mathbf{x}_t^*$ lies in $\mathcal{A}_{t,\ell}$.

Then we can bound the regret incurred in each layer separately.

Lemma 7.5.11. Suppose the the high probability events $\mathcal{E}^{\text{bonus}}$ and $\mathcal{E}$ holds. Then for all $\ell \in [L]/1$, the regret incurred by the index set $\Psi_{T+1,\ell}$ is bounded by

$$\sum_{s\in\Psi_{T+1,\ell}} \left(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\right) \le \widetilde{O}\left(d\cdot 2^\ell \widehat{\beta}_{T,\ell-1}\right).$$

With all these lemmas, we can prove Theorem 7.5.1.

Proof of Theorem 7.5.1. Conditioned on $\mathcal{E}^{\text{bonus}} \cap \mathcal{E}$, let

$$\ell^* = \left\lceil \log_2(64(L_\mu/\kappa_\mu)\sqrt{\log(4(T+1)^2L/\delta)}) \right\rceil.$$

Using the high probability event $\mathcal{E}^{\text{bonus}}$, Lemma 7.5.10 and Lemma 7.5.11, for any $\ell > \ell^*$, we have

$$\begin{aligned}
&\sum_{s\in\Psi_{T+1,\ell}} \left(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\right) \\
&\quad\le \widetilde{O}\left(d\cdot 2^\ell \widehat{\beta}_{T,\ell-1}\right) \\
&\quad\le \widetilde{O}\left(\frac{d}{\kappa_\mu}\sqrt{\sum_{s\in\Psi_{T+1,\ell}} w_s^2\left(o_s - \mu((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{T+1,\ell})\right)^2 + 1} + 1\right) \\
&\quad\le \widetilde{O}\left(\frac{d}{\kappa_\mu}\sqrt{\sum_{t=1}^{T}\sigma_t^2} + \frac{d}{\kappa_\mu} + 1\right),
\end{aligned} \tag{7.5.3}$$

where the first inequality holds due to Lemma 7.5.11. The second inequality holds due to the definition 7.4.3. The last inequality holds due to Lemma 7.5.9 and $w_s \le 1$.

For $\ell \in [\ell^*]$, we have

$$\begin{aligned}
&\sum_{s\in\Psi_{T+1,\ell}} \left(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\right) \\
&\quad\le 4|\Psi_{T+1,\ell}| \\
&\quad= 2^{2\ell+2}\sum_{s\in\Psi_{T+1,\ell}} \|w_s(\mathbf{x}_s-\mathbf{y}_s)\|^2_{\widehat{\boldsymbol{\Sigma}}_{s,\ell}} \\
&\quad\le 2^{2\ell+3} d\log(1+T/(d\lambda)) \\
&\quad= \widetilde{O}\left(\frac{dL_\mu^2}{\kappa_\mu^2}\right),
\end{aligned} \tag{7.5.4}$$

where the first equality holds due to our choice of w_s such that $\|w_s(\mathbf{x}_s-\mathbf{y}_s)\|^2_{\widehat{\boldsymbol{\Sigma}}_{s,\ell}}$. The second inequality holds due to Lemma 7.7.1. The last equality holds due to $\ell \le \ell^*$

For any $s \in [T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})$, we set ℓ_s as the value of layer such that $\|\mathbf{x}_s - \mathbf{y}_s\|_{\widehat{\boldsymbol{\Sigma}}^{-1}_{s,\ell}} \le \alpha$ for all $\mathbf{x}_s, \mathbf{y}_s \in \mathcal{A}_{s,\ell}$ and then the while loop ends. By the choice of $\mathbf{x}_s, \mathbf{y}_s$ and $\mathbf{x}_s^* \in \mathcal{A}_{s,\ell_s}$ (Lemma 7.5.10), we have

$$\begin{aligned}
2\mathbf{x}_s^{*\top}\widehat{\boldsymbol{\theta}}_{s,\ell_s} &\le \mathbf{x}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} + \mathbf{y}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} + \widehat{\beta}_{s,\ell_s}\|\mathbf{x}_s-\mathbf{y}_s\|_{\widehat{\boldsymbol{\Sigma}}^{-1}_{s,\ell_s}} \\
&\le \mathbf{x}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} + \mathbf{y}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} + \widehat{\beta}_{s,\ell_s}\alpha,
\end{aligned} \tag{7.5.5}$$

where the last inequality holds because $\|\mathbf{x}_s - \mathbf{y}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell}^{-1}} \leq \alpha$ for all $\mathbf{x}_s, \mathbf{y}_s \in \mathcal{A}_{s,\ell}$. Then we have

$$
\begin{aligned}
&\sum_{s\in[T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})} \left(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top\boldsymbol{\theta}^* + \mathbf{y}_s^\top\boldsymbol{\theta}^*)\right) \\
&= \sum_{s\in[T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})} \Bigg(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - 2\mathbf{x}_s^{*\top}\widehat{\boldsymbol{\theta}}_{s,\ell_s} + \left(\mathbf{x}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} - \mathbf{x}_s^\top\boldsymbol{\theta}^*\right) \\
&\quad + \left(\mathbf{y}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} - \mathbf{y}_s^\top\boldsymbol{\theta}^*\right) + \left(2\mathbf{x}_s^{*\top}\widehat{\boldsymbol{\theta}}_{s,\ell_s} - (\mathbf{x}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s} + \mathbf{y}_s^\top\widehat{\boldsymbol{\theta}}_{s,\ell_s})\right)\Bigg) \\
&\leq \sum_{s\in[T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})} \left(\|\mathbf{x}_s^* - \mathbf{x}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell_s}^{-1}} + \|\mathbf{x}_s^* - \mathbf{y}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell_s}^{-1}}\right)\|\boldsymbol{\theta}^* - \widehat{\boldsymbol{\theta}}_{s,\ell_s}\|_{\widehat{\Sigma}_{s,\ell_s}} + \widehat{\beta}_{s,\ell_s}\alpha \\
&\leq \sum_{s\in[T]/(\cup_{\ell\in[L]}\Psi_{T+1,\ell})} 3\widehat{\beta}_{s,\ell_s}\alpha \\
&\leq T\cdot\widetilde{O}\left(1/T\right) = \widetilde{O}(1),
\end{aligned} \tag{7.5.6}
$$

where the first inequality holds due to the Cauchy-Schwarz inequality and (7.5.5). The third inequality holds due to $\|\mathbf{x}_s - \mathbf{y}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell}^{-1}} \leq \alpha$ for all $\mathbf{x}_s, \mathbf{y}_s \in \mathcal{A}_{s,\ell_s}$, $\mathbf{x}_s^* \in \mathcal{A}_{s,\ell_s}$ (Lemma 7.5.10) and Lemma 7.5.7. The third inequality holds due to our choice of $\widehat{\beta}_{s,\ell_s} \leq \widetilde{O}(\sqrt{T})$ and $\alpha = 1/T^{3/2}$. Combining (7.5.3), (7.5.4), (7.5.6) together, we obtain

$$
\text{Regret}(T) = \widetilde{O}\left(\frac{d}{\kappa_\mu}\sqrt{\sum_{t=1}^{T}\sigma_t^2 + d\left(\frac{L_\mu^2}{\kappa_\mu^2} + \frac{1}{\kappa_\mu}\right)}\right).
$$

□

7.6 Experiments

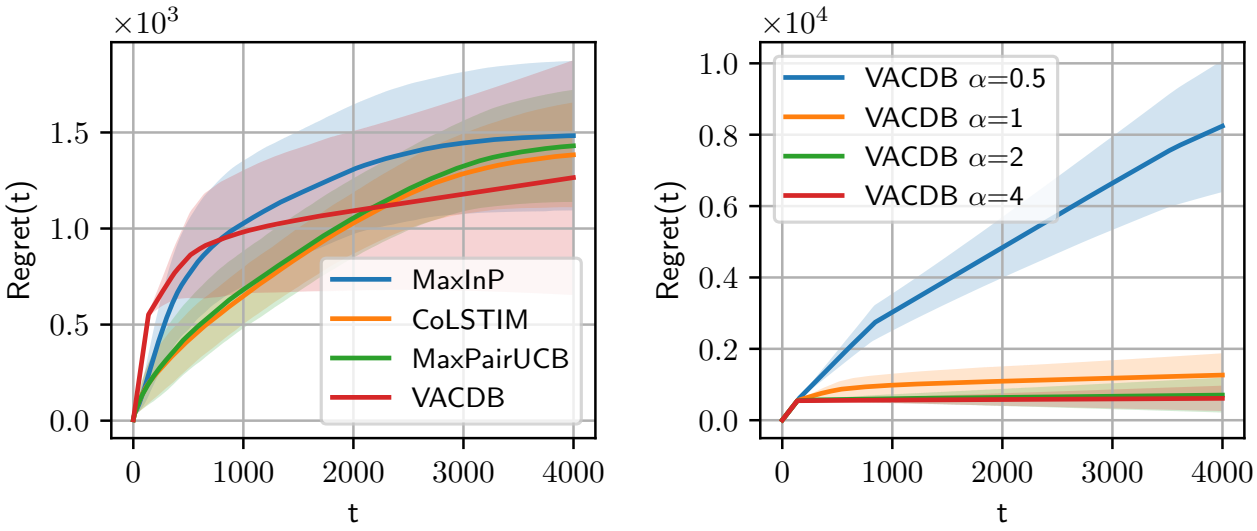
Experiment Setup. We study the proposed algorithm in simulation to compare it with those that are also designed for contextual dueling bandits. Each experiment instance is simulated for $T = 4000$ rounds. The unknown parameter $\boldsymbol{\theta}^*$ to be estimated is generated at random and normalized to be a unit vector. The feature dimension is set to $d = 5$. A total of $|\mathcal{A}_t| = 2^d$ distinct contextual vectors are generated from $\{-1, 1\}^d$. In each round, given the arm pair selected by the algorithm, a response is generated according to the random process defined in Section 7.3. For each experiment, a total of 128 repeated runs are carried out. We tune the confidence radius of each algorithm to showcase the best performance. The average cumulative regret is reported in Fig. 7.1 along with the standard deviation in the shaded region. The link function $\mu(\cdot)$ is set to be the logistic function.

Algorithms. We list the algorithms studied in this section as follows:

- `MaxInP`: Maximum Informative Pair by Saha (2021). It maintains an active set of possible optimal arms each round. The pairs are chosen on the basis of the maximum uncertainty in the difference between the two arms. Instead of using a warm-up period τ_0 in their definition, we initialize $\mathbf{\Sigma}_0 = \lambda\mathbf{I}$ as regularization. When $\lambda = 0.001$ this approach empirically has no significant impact on regret performance compared to the warm-up method.

- `MaxPairUCB`: In this algorithm, we keep the MLE the same as `MaxInP`. However, we eliminate the need for an active set of arms, and the pair of arms that is picked is according to the term defined in (7.4.4).

- `CoLSTIM`: This method is from Bengs et al. (2022). First, they add randomly disturbed utilities to each arm and pick the arm that has the best estimation. They claim this step achieves better empirical performance. The second arm is chosen according to criteria as defined in (7.4.5).



(a) Compare proposed algorithm with baselines.

(b) Variance-awareness of the proposed algorithm.

Figure 7.1: Experiments showing regret performance in various settings.

- `VACDB`: The proposed variance-aware Algorithm 7.1 in this chapter. α is set to this theoretical value according to Theorem 7.5.1. However, we note that for this specific experiment, $L = 4$ is enough to eliminate all suboptimal arms. The estimated $\widehat{\boldsymbol{\theta}}$ in one layer below is used to initialize the MLE of the upper layer when it is first reached to provide a rough estimate since the data is not shared among layers.

Regret Comparison. In Fig. 7.1a we first notice that the proposed method `VACDB` has a better regret over other methods on average, demonstrating its efficiency. Second, the `MaxPairUCB` and `CoLSTIM` algorithm have a slight edge over the `MaxInP` algorithm empirically, which can be partially explained by the discussion in Section 7.4.4. The contributing factor for this could be that in `MaxInP` the chosen pair is solely based on uncertainty, while the other two methods choose at least one arm that maximizes the reward.

Variance-Awareness. In Fig. 7.1b, we show the variance awareness of our algorithm by scaling the unknown parameter $\boldsymbol{\theta}^*$. Note that the variance of the Bernoulli distribution with parameter p is $\sigma^2 = p(1-p)$. To generate high- and low-variance instances, we scale the parameter $\boldsymbol{\theta}^*$ by a ratio of $\alpha \in \{0.5, 1, 2, 4\}$. If $\alpha \geq 1$ then p will be closer to 0 or 1 which results in a lower variance instance, and vice versa. In this plot, we show the result under four cases where the scale is set in an increasing manner, which corresponds to reducing the variance of each arm. With decreasing variance, our algorithm suffers less regret, which corresponds to the decrease in the σ_t term in our main theorem.

7.6.1 Additional Experiment on Real-world Data

7.6.2 Comparison with Prior Works

In this section, we provide a detailed discussion of the layered design, drawing a comparison with `Sta'D` in Saha (2021) and `SupCoLSTIM` in Bengs et al. (2022). The general idea follows Auer (2002), which focuses on maintaining a set of "high confidence promising arms". The algorithm operates differently in two distinct scenarios. If there are some pairs $(\mathbf{x}_t, \mathbf{y}_t)$ in the current layer ℓ with high uncertainty, represented by $\|\mathbf{x}_t - \mathbf{y}_t\|_{\widehat{\boldsymbol{\Sigma}}^{-1}_{t,\ell}}$, we will explore those arm pairs. Conversely, when achieving the desired accuracy, we eliminate suboptimal arms using our confidence set and proceed to a subsequent layer demanding greater accuracy. This process continues until we reach a sufficiently accurate high layer, at which we make decisions based on the remaining arms in the confidence set and the estimated parameters $\widehat{\boldsymbol{\theta}}_{t,\ell}$.
In the final stage, `Sta'D` picks the first arm $\mathbf{x}_t$ as the one with the maximum estimated score, followed by choosing its strongest challenger $\mathbf{y}_t$, which has the highest optimistic opportunity to beat $\mathbf{x}_t$. `SupCoLSTIM` adopts a similar policy and distinguishes itself with a randomized learning strategy by generating additive

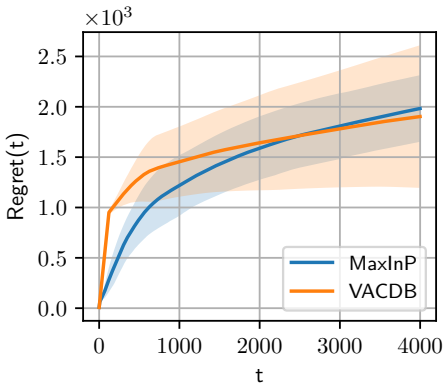


Figure 7.2: Regret comparison between `VACDB` and `MaxInP` on a real-world dataset.

noise terms from an underlying perturbation distribution. Our arm selection is based on the symmetric arm selection policy described in Section 7.4.4.

`Sta'D` and `SupCoLSTIM` choose the confidence set radius $\widehat{\beta}_{t,\ell}$ to be $2^{-\ell}$ in the ℓ-th layer. In comparison, our choice $\widehat{\beta}_{t,\ell}$ is defined in (7.4.3). As we mention in Section 7.4.3, apart from the $2^{-\ell}$ dependency on the layer ℓ, it also relies on the estimated variance. Such a variance-adaptive confidence set radius helps to achieve the variance-aware regret bound.

To showcase the performance of our algorithms in a real-world setting, we use EventTime dataset (Zhang et al., 2016). In this dataset, $K = 100$ historical events are compared in a pairwise fashion by crowd-sourced workers. The data contains binary response indicating which one of the events the worker thinks precedes the other. There is no side information

$$\mathcal{A} = \{\mathbf{x}_i, i \in [K]\},$$

or the true parameter $\boldsymbol{\theta}^*$ readily available in the dataset. Thus, we estimate them with pairwise comparison data. To achieve this, let $C_{ij}, i, j \in [K]$ be the number of times event j precedes event i labeled by the workers. The following MLE is used:

$$\underset{\{\mathbf{x}_i\},\boldsymbol{\theta}}{\operatorname{argmax}} \sum_{i\in[K]} \sum_{j\in[K]} C_{ij} \log\left(\sigma((\mathbf{x}_i - \mathbf{x}_j)^\top \boldsymbol{\theta})\right).$$

With the estimated $\mathcal{A}$ and $\boldsymbol{\theta}^*$, it is then possible to simulate the interactive process. We compared our algorithm `VACDB` with `MaxInP` in Fig. 7.2. We can see that after about 2500 rounds, our algorithm starts to outperform `MaxInP` in terms of cumulative regret.

7.7 Proof of Lemmas

7.7.1 Proof of Lemma 7.5.7

Proof of Lemma 7.5.7. For a fixed $\ell \in [L]$, let $t \in \Psi_{T+1,\ell}$, $t \geq 2$, we define some auxiliary quantities:

$$G_{t,\ell}(\boldsymbol{\theta}) = 2^{-2\ell}\kappa_\mu \boldsymbol{\theta} + \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \left[\mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \boldsymbol{\theta}\right) - \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \boldsymbol{\theta}^*\right)\right](\mathbf{x}_s - \mathbf{y}_s)$$

$$\epsilon_t = o_t - \mu\left((\mathbf{x}_t - \mathbf{y}_t)^\top \boldsymbol{\theta}^*\right)$$

$$Z_{t,\ell} = \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \epsilon_s (\mathbf{x}_s - \mathbf{y}_s).$$

Recall (7.4.1), $\widehat{\boldsymbol{\theta}}_{t,\ell}$ is the solution to

$$2^{-2\ell}\kappa_\mu \widehat{\boldsymbol{\theta}}_{t,\ell} + \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \left(\mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell}\right) - o_s\right)(\mathbf{x}_s - \mathbf{y}_s) = \mathbf{0}. \tag{7.7.1}$$

A simple transformation shows that (7.7.1) is equivalent to following equation,

$$\begin{aligned} G_{t,\ell}\left(\widehat{\boldsymbol{\theta}}_{t,\ell}\right) &= 2^{-2\ell}\kappa_\mu \widehat{\boldsymbol{\theta}}_{t,\ell} + \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \left[\mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell}\right) - \mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top \boldsymbol{\theta}^*\right)\right](\mathbf{x}_s-\mathbf{y}_s) \\ &= \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \left[o_s - \mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top \boldsymbol{\theta}^*\right)\right](\mathbf{x}_s-\mathbf{y}_s) \\ &= Z_{t,\ell}. \end{aligned}$$

It is assumed that $G_{t,\ell}$ is invertible and thus $\widehat{\boldsymbol{\theta}}_{t,\ell} = G_{t,\ell}^{-1}(Z_{t,\ell})$.

Moreover, we can see that $G_{t,\ell}(\boldsymbol{\theta}^*) = 2^{-2\ell}\kappa_\mu\boldsymbol{\theta}^*$. Recall $\widehat{\boldsymbol{\Sigma}}_{t,\ell} = 2^{-2\ell}\kappa_\mu\mathbf{I} + \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2(\mathbf{x}_s-\mathbf{y}_s)(\mathbf{x}_s-\mathbf{y}_s)^\top$. We have

$$\begin{aligned} \left\|G_{t,\ell}(\widehat{\boldsymbol{\theta}}_{t,\ell}) - G_{t,\ell}(\boldsymbol{\theta}^*)\right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}}^2 &= (\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*)^\top F(\bar{\boldsymbol{\theta}})\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}F(\bar{\boldsymbol{\theta}})(\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*) \\ &\geq \kappa_\mu^2(\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*)^\top\widehat{\boldsymbol{\Sigma}}_{t,\ell}(\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*) \\ &= \kappa_\mu^2\|\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}}^2, \end{aligned}$$

where the first inequality holds because $\dot{\mu}(\cdot) \geq \kappa_\mu > 0$ and thus $F(\bar{\boldsymbol{\theta}}) \succeq \kappa_\mu\widehat{\boldsymbol{\Sigma}}_{t,\ell}$. Using the triangle inequality, we have

$$\begin{aligned} \left\|\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*\right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}} &\leq 2^{-2\ell}\|\boldsymbol{\theta}^*\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} + \frac{1}{\kappa_\mu}\|Z_{t,\ell}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \\ &\leq 2^{-\ell}\|\boldsymbol{\theta}^*\|_2 + \frac{1}{\kappa_\mu}\|Z_{t,\ell}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}}. \end{aligned}$$

To bound the $\|Z_{t,\ell}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}}$ term, we use Lemma 7.7.3. By the choice of w_s, for any $t \in \Psi_{T+1,\ell}$, we have

$$\|w_t(\mathbf{x}_t-\mathbf{y}_t)\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} = 2^{-\ell} \text{ and } w_t \leq 1.$$

We also have

$$\mathbb{E}[w_t^2\epsilon_t^2 \mid \mathcal{F}_t] \leq w_t^2\mathbb{E}[\epsilon_t^2 \mid \mathcal{F}_t] \leq w_t^2\sigma_t^2 \text{ and } |w_t\epsilon_t| \leq |\epsilon_t| \leq 1.$$

Therefore, Lemma 7.7.3 shows that with probability at least $1-\delta/L$, for all $t \in \Psi_{T+1,\ell}$, the following inequality holds

$$\|Z_{t,\ell}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \leq 16\cdot 2^{-\ell}\sqrt{\sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2\sigma_s^2\log(4t^2L/\delta)} + 6\cdot 2^{-\ell}\log(4t^2L/\delta).$$

Finally, we get

$$\left\|\widehat{\boldsymbol{\theta}}_{t,\ell}-\boldsymbol{\theta}^*\right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}} \leq \frac{2^{-\ell}}{\kappa_\mu}\left[16\sqrt{\sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2\sigma_s^2\log(4t^2L/\delta)} + 6\log(4t^2L/\delta)\right] + 2^{-\ell}.$$

Take a union bound on all $\ell \in [L]$, and then we finish the proof of Lemma 7.5.7. □

7.7.2 Proof of Lemma 7.5.8

Proof of Lemma 7.5.8. The proof of this lemma is similar to the proof of Lemma B.4 in Zhao et al. (2023a). For a fixed layer $\ell \in [L]$, using the definition of ϵ_s and σ_s, we have

$$\forall s \geq 1, \mathbb{E}[\epsilon_s^2 - \sigma_s^2 | \mathbf{x}_{1:s}, \mathbf{y}_{1:s}, o_{1:s-1}] = 0.$$

Therefore, we have

$$\sum_{s\in\Psi_{t,\ell}} \mathbb{E}[w_s^2(\epsilon_s^2-\sigma_s^2)^2|\mathbf{x}_{1:s},\mathbf{y}_{1:s},o_{1:s-1}] \le \sum_{s\in\Psi_{t,\ell}} \mathbb{E}[w_s^2\epsilon_s^4|\mathbf{x}_{1:s},\mathbf{y}_{1:s},o_{1:s-1}] \le \sum_{s\in\Psi_{t.\ell}} w_s^2\sigma_s^2,$$

where the last inequality holds due to the definition of σ_s and $\epsilon_s \le 1$. Then using Lemma 7.7.2 and taking a union bound on all $\ell \in [L]$, for all $t \ge 2$, we have

$$\begin{aligned}\left|\sum_{s\in\Psi_{t,\ell}} w_s^2(\epsilon_s^2-\sigma_s^2)\right| &\le \sqrt{2\sum_{s\in\Psi_{t.\ell}} w_s^2\sigma_s^2\log(4t^2L/\delta)} + \frac{2}{3}\cdot 2\log(4t^2L/\delta)\\ &\le \frac{1}{2}\sum_{s\in\Psi_{t.\ell}} w_s^2\sigma_s^2 + \frac{7}{3}\log(4t^2L/\delta),\end{aligned} \tag{7.7.2}$$

where we use the Young's inequality $ab \le \frac{1}{2}a^2 + \frac{1}{2}b^2$. Finally, we finish the proof of Lemma 7.5.8 by

$$\begin{aligned}\sum_{s\in\Psi_{t,\ell}} w_s^2\sigma_s^2 &= \left|\sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 - \sum_{s\in\Psi_{t,\ell}} w_s^2(\epsilon_s^2-\sigma_s^2)\right|\\ &\le \sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 + \left|\sum_{s\in\Psi_{t,\ell}} w_s^2(\epsilon_s^2-\sigma_s^2)\right|\\ &\le \sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 + \frac{1}{2}\sum_{s\in\Psi_{t.\ell}} w_s^2\sigma_s^2 + \frac{7}{3}\log(4t^2L/\delta),\end{aligned} \tag{7.7.3}$$

where the first inequality holds due to the triangle inequality. The second inequality holds due to (7.7.2). We also have

$$\begin{aligned}\sum_{s\in\Psi_{t,\ell}} w_s^2\sigma_s^2 &= \left|\sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 - \sum_{s\in\Psi_{t,\ell}} w_s^2(\epsilon_s^2-\sigma_s^2)\right|\\ &\ge \sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 - \left|\sum_{s\in\Psi_{t,\ell}} w_s^2(\epsilon_s^2-\sigma_s^2)\right|\\ &\ge \sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 - \frac{1}{2}\sum_{s\in\Psi_{t.\ell}} w_s^2\sigma_s^2 - \frac{7}{3}\log(4t^2L/\delta).\end{aligned}$$

The proof of this inequality is almost the same as (7.7.3). □

7.7.3 Proof of Lemma 7.5.9

Proof of Lemma 7.5.9. For a fixed $\ell \in [L]$, Lemma 7.5.8 indicates that

$$\begin{aligned}\sum_{s\in\Psi_{t,\ell}} w_s^2\sigma_s^2 &\le 2\sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 + \frac{14}{3}\log(4t^2L/\delta)\\ &\le \frac{14}{3}\log(4t^2L/\delta) + 4\sum_{s\in\Psi_{t,\ell}} w_s^2\left(o_s - \mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)^2\\ &\quad + \underbrace{4\sum_{s\in\Psi_{t,\ell}} w_s^2\left(\epsilon_s - \left(o_s - \mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)\right)^2}_{(I)},\end{aligned} \tag{7.7.4}$$

where the second inequality holds due to the basic inequality $(a+b)^2 \leq 2a^2+2b^2$ for all $a, b \in \mathbb{R}$. Using our definition of ϵ_s, $o_s = \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \boldsymbol{\theta}^*\right) + \epsilon_s$. Thus, we have

$$\begin{aligned}
(I) &= \sum_{s\in\Psi_{t,\ell}} w_s^2 \left(\epsilon_s - \left(o_s - \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)\right)^2 \\
&= \sum_{s\in\Psi_{t,\ell}} w_s^2 \left(\mu\big((\mathbf{x}_s - \mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell}\big) - \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \boldsymbol{\theta}^*\right)\right)^2 \\
&\leq L_\mu^2 \sum_{s\in\Psi_{t,\ell}} w_s^2 \left((\mathbf{x}_s - \mathbf{y}_s)^\top \big(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\big)\right)^2, \qquad (7.7.5)
\end{aligned}$$

where the last inequality holds because the first order derivative of function μ is upper bounded by L_μ (Assumption 7.3.2). Moreover, by expanding the square, we have

$$\begin{aligned}
(I) &\leq L_\mu^2 \sum_{s\in\Psi_{t,\ell}} w_s^2 \left((\mathbf{x}_s - \mathbf{y}_s)^\top \big(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\big)\right)^2 \\
&= L_\mu^2 \sum_{s\in\Psi_{t,\ell}} \big(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\big)^\top w_s^2 (\mathbf{x}_s - \mathbf{y}_s)(\mathbf{x}_s - \mathbf{y}_s)^\top \big(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\big) \\
&= L_\mu^2 \big(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\big)^\top \left(\sum_{s\in\Psi_{t,\ell}} w_s^2 (\mathbf{x}_s - \mathbf{y}_s)(\mathbf{x}_s - \mathbf{y}_s)^\top\right) \big(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\big) \\
&\leq L_\mu^2 \left\|\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\right\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}}^2, \qquad (7.7.6)
\end{aligned}$$

where the last inequality holds due to

$$\widehat{\boldsymbol{\Sigma}}_{t,\ell} = 2^{-2\ell}\kappa_\mu \mathbf{I} + \sum_{s\in\Psi_{t,\ell}} w_s^2 (\mathbf{x}_s - \mathbf{y}_s)(\mathbf{x}_s - \mathbf{y}_s)^\top \succeq \sum_{s\in\Psi_{t,\ell}} w_s^2 (\mathbf{x}_s - \mathbf{y}_s)(\mathbf{x}_s - \mathbf{y}_s)^\top.$$

Combining (7.7.5), (7.7.6) and the event $\mathcal{E}$ (Lemma 7.5.7), we have

$$\begin{aligned}
(I) &\leq \frac{2^{-2\ell} L_\mu^2}{\kappa_\mu^2} \left[16\sqrt{\sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \sigma_s^2 \log(4(t+1)^2 L/\delta)} + 6\log(4(t+1)^2 L/\delta) + \kappa_\mu\right]^2 \\
&\leq \frac{2^{-2\ell} L_\mu^2}{\kappa_\mu^2} \left[512\log(4(t+1)^2 L/\delta) \cdot \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \sigma_s^2 + 2\left(6\log(4(t+1)^2 L/\delta) + \kappa_\mu\right)^2\right],
\end{aligned}$$

where the last inequality holds due to the basic inequality $(a+b)^2 \leq 2a^2+2b^2$ for all $a, b \in \mathbb{R}$. When $2^\ell \geq 64(L_\mu/\kappa_\mu)\sqrt{\log(4(t+1)^2 L/\delta)}$, we can further bound the above inequality by

$$(I) \leq \frac{1}{8} \sum_{s\in\boldsymbol{\Psi}_{t+1,\ell}} w_s^2 \sigma_s^2 + \log(4(t+1)^2 L/\delta). \qquad (7.7.7)$$

Subitituting (7.7.7) into (7.7.4), we have

$$\begin{aligned}
\sum_{s\in\Psi_{t,\ell}} w_s^2 \sigma_s^2 \leq 4 \sum_{s\in\Psi_{t,\ell}} w_s^2 \left(o_s - \mu\left((\mathbf{x}_s - \mathbf{y}_s)^\top \widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)^2 \\
+ 9\log(4(t+1)^2 L/\delta) + \frac{1}{2} \sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2 \sigma_s^2.
\end{aligned}$$

Therefore, we prove the first inequality in Lemma 7.5.9 as follows

$$\sum_{s\in\Psi_{t,\ell}} w_s^2\sigma_s^2 \le 8\sum_{s\in\Psi_{t,\ell}} w_s^2\left(o_s-\mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)^2 + 18\log(4(t+1)^2L/\delta).$$

For the second inequality, we have

$$\sum_{s\in\Psi_{t,\ell}} w_s^2\left(o_s-\mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)^2 \le 2\sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 + 2\underbrace{\sum_{s\in\Psi_{t,\ell}} w_s^2\left(\epsilon_s-\left(o_s-\mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)\right)^2}_{(I)}.$$

We complete the proof of Lemma 7.5.9.

$$\begin{aligned}
&\sum_{s\in\Psi_{t,\ell}} w_s^2\left(o_s-\mu\left((\mathbf{x}_s-\mathbf{y}_s)^\top\widehat{\boldsymbol{\theta}}_{t,\ell}\right)\right)^2\\
&\le 2\sum_{s\in\Psi_{t,\ell}} w_s^2\epsilon_s^2 + \frac{1}{4}\sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2\sigma_s^2 + 2\log(4(t+1)^2L/\delta)\\
&\le 2\left(\frac{3}{2}\sum_{s\in\Psi_{t,\ell}} w_s^2\sigma_s^2 + \frac{7}{3}\log(4t^2L/\delta)\right) + \frac{1}{4}\sum_{s\in\boldsymbol{\Psi}_{t,\ell}} w_s^2\sigma_s^2 + 2\log(4(t+1)^2L/\delta)\\
&\le 4\sum_{s\in\Psi_{t,\ell}} w_s^2\sigma_s^2 + 8\log(4(t+1)^2L/\delta),
\end{aligned}$$

where the first inequality holds due to (7.7.7). The second inequality holds due to Lemma 7.5.8. □

7.7.4 Proof of Lemma 7.5.10

Proof of Lemma 7.5.10. We prove it by induction. For $\ell = 1$, we initialze the set $\mathcal{A}_{t,1}$ to be $\mathcal{A}_t$, thus trivially $\mathbf{x}_t^* \in \mathcal{A}_{t,1}$. Now we suppose $\mathcal{A}_{t,\ell}$ is defined and $\mathbf{x}_t^* \in \mathcal{A}_{t,\ell}$. By the way $\mathcal{A}_{t,\ell+1}$ is constructed, $\mathcal{A}_{t,\ell+1}$ is defined only when $\|\mathbf{x}-\mathbf{y}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \le 2^{-\ell}$ for all $\mathbf{x},\mathbf{y}\in\mathcal{A}_{t,\ell}$.

Let $\mathbf{x}_{\max} = \operatorname{argmax}_{\mathbf{x}\in\mathcal{A}_{t,\ell}} \mathbf{x}^\top\widehat{\boldsymbol{\theta}}_{t,\ell}$. Then we have

$$\begin{aligned}
\mathbf{x}_t^{*\top}\widehat{\boldsymbol{\theta}}_{t,\ell} - \mathbf{x}_{\max}^\top\widehat{\boldsymbol{\theta}}_{t,\ell} &= (\mathbf{x}_t^{*\top}\boldsymbol{\theta}^* - \mathbf{x}_{\max}^\top\boldsymbol{\theta}^*) + (\mathbf{x}_t^* - \mathbf{x}_{\max})^\top(\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*)\\
&\ge -\|\mathbf{x}_t^* - \mathbf{x}_{\max}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \cdot \|\widehat{\boldsymbol{\theta}}_{t,\ell} - \boldsymbol{\theta}^*\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}},
\end{aligned}$$

where the inequality holds due to the Cauchy-Schwarz inequality and the fact $\mathbf{x}_t^* = \operatorname{argmax}_{\mathbf{x}\in\mathcal{A}_t} \mathbf{x}^\top\boldsymbol{\theta}^*$. With the inductive hypothesis, we know $\mathbf{x}_t^* \in \mathcal{A}_{t,\ell}$. Thus we have $\|\mathbf{x}_t^* - \mathbf{x}_{\max}\|_{\widehat{\boldsymbol{\Sigma}}_{t,\ell}^{-1}} \le 2^{-\ell}$. Finally, with the inequality in Lemma 7.5.7, we have

$$\mathbf{x}_t^{*\top}\widehat{\boldsymbol{\theta}}_{t,\ell} \ge \max_{\mathbf{x}\in\mathcal{A}_{t,\ell}} \mathbf{x}^\top\widehat{\boldsymbol{\theta}}_{t,\ell} - 2^{-\ell}\widehat{\beta}_{t,\ell}.$$

Therefore, we have $\mathbf{x}_t^* \in \mathcal{A}_{t,\ell+1}$, and we complete the proof of Lemma 7.5.10 by induction. □

7.7.5 Proof of Lemma 7.5.11

Proof of Lemma 7.5.11. For any $s \in \Psi_{T+1,\ell}$, due to the definition of $\Psi_{T+1,\ell}$ and our choice of $\mathbf{x}_s, \mathbf{y}_s$ (Algorithm 7.1 Line 14-16), we have $\mathbf{x}_s, \mathbf{y}_s \in \mathcal{A}_{s,\ell}$. Additionally, because the set $\mathcal{A}_{s,\ell}$ is defined, $\|\mathbf{x}-\mathbf{y}\|_{\widehat{\boldsymbol{\Sigma}}_{s,\ell-1}^{-1}} \le$

$2^{-\ell+1}$ for all $\mathbf{x}, \mathbf{y} \in \mathcal{A}_{s,\ell-1}$. From Lemma 7.5.10, we can see that $\mathbf{x}_s^* \in \mathcal{A}_{s,\ell}$. Combining these results, we have

$$\|\mathbf{x}_s^* - \mathbf{x}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell-1}^{-1}} \leq 2^{-\ell+1}, \|\mathbf{x}_s^* - \mathbf{y}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell-1}^{-1}} \leq 2^{-\ell+1}, \tag{7.7.8}$$

where we use the inclusion property $\mathcal{A}_{s,\ell} \subseteq \mathcal{A}_{s,\ell-1}$. Moreover, $\mathbf{x}_s, \mathbf{x}_s^* \in \mathcal{A}_{s,\ell}$ shows that

$$\begin{aligned}\mathbf{x}_s^\top \widehat{\boldsymbol{\theta}}_{s,\ell-1} &\geq \max_{\mathbf{x} \in \mathcal{A}_{s,\ell-1}} \mathbf{x}^\top \widehat{\boldsymbol{\theta}}_{s,\ell-1} - 2^{-\ell+1}\widehat{\beta}_{s,\ell-1} \\ &\geq \mathbf{x}_s^{*\top} \widehat{\boldsymbol{\theta}}_{s,\ell-1} - 2^{-\ell+1}\widehat{\beta}_{s,\ell-1},\end{aligned} \tag{7.7.9}$$

where we use $\mathbf{x}_s \in \mathcal{A}_{s,\ell-1}$. Similarly, we have

$$\mathbf{y}_s^\top \widehat{\boldsymbol{\theta}}_{s,\ell-1} \geq \mathbf{x}_s^{*\top} \widehat{\boldsymbol{\theta}}_{s,\ell-1} - 2^{-\ell+1}\widehat{\beta}_{s,\ell-1}. \tag{7.7.10}$$

Now we compute the regret incurred in round s.

$$\begin{aligned}2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - \left(\mathbf{x}_s^\top \boldsymbol{\theta}^* + \mathbf{y}_s^\top \boldsymbol{\theta}^*\right) &= \left(\mathbf{x}_s^* - \mathbf{x}_s\right)^\top \boldsymbol{\theta}^* + \left(\mathbf{x}_s^* - \mathbf{y}_s\right)^\top \boldsymbol{\theta}^* \\ &\leq \left(\mathbf{x}_s^* - \mathbf{x}_s\right)^\top \widehat{\boldsymbol{\theta}}_{s,\ell-1} + \left|\left(\mathbf{x}_s^* - \mathbf{x}_s\right)^\top \left(\widehat{\boldsymbol{\theta}}_{s,\ell-1} - \boldsymbol{\theta}^*\right)\right| \\ &\quad + \left(\mathbf{x}_s^* - \mathbf{y}_s\right)^\top \widehat{\boldsymbol{\theta}}_{s,\ell-1} + \left|\left(\mathbf{x}_s^* - \mathbf{y}_s\right)^\top \left(\widehat{\boldsymbol{\theta}}_{s,\ell-1} - \boldsymbol{\theta}^*\right)\right| \\ &\leq 2^{-\ell+1}\widehat{\beta}_{s,\ell-1} + \|\mathbf{x}_s^* - \mathbf{x}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell-1}^{-1}} \left\|\widehat{\boldsymbol{\theta}}_{s,\ell-1} - \boldsymbol{\theta}^*\right\|_{\widehat{\mathbf{\Sigma}}_{s,\ell-1}} \\ &\quad + 2^{-\ell+1}\widehat{\beta}_{s,\ell-1} + \|\mathbf{x}_s^* - \mathbf{y}_s\|_{\widehat{\mathbf{\Sigma}}_{s,\ell-1}^{-1}} \left\|\widehat{\boldsymbol{\theta}}_{s,\ell-1} - \boldsymbol{\theta}^*\right\|_{\widehat{\mathbf{\Sigma}}_{s,\ell-1}} \\ &\leq 8 \cdot 2^{-\ell}\widehat{\beta}_{s,\ell-1},\end{aligned} \tag{7.7.11}$$

where the first inequality holds due to the basic inequality $x \leq |x|$ for all $x \in \mathbb{R}$. The second inequality holds due t (7.7.9), (7.7.10) and the Cauchy-Schwarz inequality. The last inequality holds due to (7.7.8) and Lemma 7.5.7. Now we can return to the summation of regret on the index set $\Psi_{T+1,\ell}$.

$$\begin{aligned}\sum_{s \in \Psi_{T+1,\ell}} \left(2\mathbf{x}_s^{*\top}\boldsymbol{\theta}^* - (\mathbf{x}_s^\top \boldsymbol{\theta}^* + \mathbf{y}_s^\top \boldsymbol{\theta}^*)\right) &\leq \sum_{s \in \Psi_{T+1,\ell}} 8 \cdot 2^{-\ell}\widehat{\beta}_{s,\ell-1} \\ &\leq 8 \cdot 2^{-\ell}\widehat{\beta}_{T,\ell-1}|\Psi_{T+1,\ell}| \\ &\leq 8 \cdot 2^{\ell}\widehat{\beta}_{T,\ell-1} \sum_{s \in \Psi_{T+1,\ell}} \|\omega_s \cdot (\mathbf{x}_s - \mathbf{y}_s)\|_{\widehat{\mathbf{\Sigma}}_{s,\ell}^{-1}}^2 \\ &\leq 8 \cdot 2^{\ell}\widehat{\beta}_{T,\ell-1} \cdot 2d \log\left(1 + 2^{2\ell+2}T/d\right),\end{aligned}$$

where the first inequality holds due to (7.7.11). The second inequality holds due to our choice of ω_s such that $\|\omega_s \cdot (\mathbf{x}_s - \mathbf{y}_s)\|_{\widehat{\mathbf{\Sigma}}_{s,\ell}^{-1}} = 2^{-\ell}$. The last inequality holds due to Lemma 7.7.1. Therefore, we complete the proof of Lemma 7.5.11. □

7.7.6 Auxiliary Lemmas

Lemma 7.7.1 (Lemma 11, Abbasi-Yadkori et al. 2011)**.** For any $\lambda > 0$ and sequence $\{\mathbf{x}_k\}_{k=1}^K \subseteq \mathbb{R}^d$ for $k \in [K]$, define $\mathbf{Z}_k = \lambda\mathbf{I} + \sum_{i=1}^{k-1} \mathbf{x}_i\mathbf{x}_i^\top$. Then, provided that $\|\mathbf{x}_k\|_2 \leq L$ holds for all $k \in [K]$, we have

$$\sum_{k=1}^K \min\{1, \|\mathbf{x}_k\|_{\mathbf{Z}_k^{-1}}^2\} \leq 2d\log(1 + KL^2/(d\lambda)).$$

Lemma 7.7.2 (Freedman 1975)**.** Let $M, v > 0$ be fixed constants. Let $\{x_i\}_{i=1}^n$ be a stochastic process, $\{\mathcal{G}_i\}_{i\in[n]}$ be a filtration so that for all $i \in [n]$, x_i is $\mathcal{G}_i$-measurable, while almost surely

$$\mathbb{E}[x_i|\mathcal{G}_{i-1}] = 0, |x_i| \leq M, \sum_{i=1}^n \mathbb{E}[x_i^2|\mathcal{G}_{i-1}] \leq v.$$

Then for any $\delta > 0$, with probability at least $1 - \delta$, we have

$$\sum_{i=1}^{n} x_i \leq \sqrt{2v \log(1/\delta)} + 2/3 \cdot M \log(1/\delta).$$

Lemma 7.7.3 (Zhao et al. 2023a)**.** Let $\{\mathcal{G}_k\}_{k=1}^{\infty}$ be a filtration, and $\{\mathbf{x}_k, \eta_k\}_{k \geq 1}$ be a stochastic process such that $\mathbf{x}_k \in \mathbb{R}^d$ is $\mathcal{G}_k$-measurable and $\eta_k \in \mathbb{R}$ is $\mathcal{G}_{k+1}$-measurable. Let $L, \sigma, \lambda, \epsilon > 0$, $\boldsymbol{\mu}^* \in \mathbb{R}^d$. For $k \geq 1$, let $y_k = \langle \boldsymbol{\mu}^*, \mathbf{x}_k \rangle + \eta_k$, where $\eta_k, \mathbf{x}_k$ satisfy

$$\mathbb{E}[\eta_k \mid \mathcal{G}_k] = 0, |\eta_k| \leq R, \sum_{i=1}^{k} \mathbb{E}[\eta_i^2 \mid \mathcal{G}_i] \leq v_k, \text{ for } \forall k \geq 1.$$

For $k \geq 1$, let $\mathbf{Z}_k = \lambda \mathbf{I} + \sum_{i=1}^{k} \mathbf{x}_i \mathbf{x}_i^\top$, $\mathbf{b}_k = \sum_{i=1}^{k} y_i \mathbf{x}_i$, $\boldsymbol{\mu}_k = \mathbf{Z}_k^{-1} \mathbf{b}_k$ and

$$\beta_k = 16\rho\sqrt{v_k \log(4k^2/\delta)} + 6\rho R \log(4k^2/\delta),$$

where $\rho \geq \sup_{k \geq 1} \|\mathbf{x}_k\|_{\mathbf{Z}_{k-1}^{-1}}$. Then, for any $0 < \delta < 1$, we have with probability at least $1 - \delta$,

$$\forall k \geq 1, \|\sum_{i=1}^{k} \mathbf{x}_i \eta_i\|_{\mathbf{Z}_k^{-1}} \leq \beta_k, \|\boldsymbol{\mu}_k - \boldsymbol{\mu}^*\|_{\mathbf{Z}_k} \leq \beta_k + \sqrt{\lambda}\|\boldsymbol{\mu}^*\|_2$$

Theorem 7.7.4 (Brouwer invariance of domain theorem,Brouwer 1911)**.** Let U be an open subset of $\mathbb{R}^d$, and let $f : U \to \mathbb{R}^d$ be a continuous injective map. Then $f(U)$ is also open.

Part IV

Conclusion

This dissertation advances the field of rank aggregation from pairwise comparisons through three major theoretical and algorithmic contributions:

1. Heterogeneous Random Utility Model: We introduced a novel extension of the Random Utility Model (RUM) that explicitly accounts for heterogeneous data sources with varying quality levels. This theoretical framework provides a principled approach to modeling and analyzing ranking data from multiple sources with different accuracy levels, laying the foundation for more sophisticated ranking algorithms.

2. Efficient Active Ranking Algorithms: We developed a family of active ranking algorithms that work under both Strong Stochastic Transitivity (SST) and Weak Stochastic Transitivity (WST) conditions:

    - For WST settings, we introduced the Probe-Rank algorithm that achieves near-optimal sample complexity
    - For SST settings, we introduced the Ada-IIR algorithm that achieves the same order of sample complexity as the oracle algorithm that has access to the optimal data source.
    - We proposed the Rank-with-Multiple-Oracles (RMO) framework that can handle WST settings with multiple data sources. And this can also be applied to the SST setting.

3. Contextual Dueling Bandits:

    - Introduced the first Borda score optimization framework for contextual dueling bandits
    - Developed variance-aware extensions to consider the noisiness of feedback

The practical impact of this work is particularly relevant in the era of large language models. Our algorithmic tools can be applied to address key challenges in:

- Reinforcement Learning with Human Feedback (RLHF), where efficient aggregation of human preferences is crucial
- Crowdsourcing platforms, where feedback quality varies significantly across contributors
- Recommendation systems, where contextual information and real-time learning are essential

Bibliography

Abbasi-Yadkori, Y., Pál, D. and Szepesvári, C. (2011). Improved algorithms for linear stochastic bandits. *Advances in neural information processing systems* **24**.

Aerts, S., Lambrechts, D., Maity, S., Van Loo, P., Coessens, B., De Smet, F., Tranchevent, L.-C., De Moor, B., Marynen, P., Hassan, B., Carmeliet, P. and Moreau, Y. (2006). Gene prioritization through genomic data fusion. *Nature Biotechnology* **24** 537–544.

Agarwal, A., Negahban, S., Wainwright, M. J. et al. (2012). Noisy matrix decomposition via convex relaxation: Optimal rates in high dimensions. *The Annals of Statistics* **40** 1171–1197.

Ailon, N. (2012). An Active Learning Algorithm for Ranking from Pairwise Preferences with an Almost Optimal Query Complexity. *J. Mach. Learn. Res.* **13**.

Ailon, N., Karnin, Z. and Joachims, T. (2014). Reducing dueling bandits to cardinal bandits. In *International Conference on Machine Learning*. PMLR.

Audibert, J.-Y., Munos, R. and Szepesvari, C. (2009). Exploration-exploitation tradeoff using variance estimates in multi-armed bandits. *Theor. Comput. Sci.* **410** 1876–1902.

Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research* **3** 397–422.

Auer, P., Cesa-Bianchi, N. and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning* **47** 235–256.

Balsubramani, A., Karnin, Z., Schapire, R. E. and Zoghi, M. (2016). Instance-dependent regret bounds for dueling bandits. In *Conference on Learning Theory*. PMLR.

Baltrunas, L., Makcinskas, T. and Ricci, F. (2010). Group Recommendations with Rank Aggregation and Collaborative Filtering. In *Proceedings of the Fourth ACM Conference on Recommender Systems*. RecSys '10, ACM, New York, NY, USA.

Bengs, V., Busa-Fekete, R., El Mesaoudi-Paul, A. and Hüllermeier, E. (2021). Preference-based online learning with dueling bandits: A survey. *Journal of Machine Learning Research* **22** 7–1.

Bengs, V., Saha, A. and Hüllermeier, E. (2022). Stochastic contextual dueling bandits under linear stochastic transitivity models. In *International Conference on Machine Learning*. PMLR.

Bradley, R. A. and Terry, M. E. (1952). Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons. *Biometrika* **39** 324–345.

Braverman, M. and Mossel, E. (2008). Noisy Sorting Without Resampling. In *ACM-SIAM Symp. Discrete Algorithms (SODA)*. Society for Industrial and Applied Mathematics, San Francisco, California.

Brouwer, L. E. (1911). Beweis der invarianz des n-dimensionalen gebiets. *Mathematische Annalen* **71** 305–313.

Busa-Fekete, R., Szorenyi, B., Cheng, W., Weng, P. and Hüllermeier, E. (2013). Top-k selection based on adaptive sampling of noisy preferences. In *International Conference on Machine Learning*. PMLR.

Caplin, A. and Nalebuff, B. (1991). Aggregation and social choice: A mean voter theorem. *Econometrica: Journal of the Econometric Society* 1–23.

Chen, J., Xu, P., Wang, L., Ma, J. and Gu, Q. (2018). Covariate adjusted precision matrix estimation via nonconvex optimization. In *International Conference on Machine Learning*.

Chen, X., Bennett, P. N., Collins-Thompson, K. and Horvitz, E. (2013). Pairwise ranking aggregation in a crowdsourced setting. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM.

Chen, Y. and Suh, C. (2015). Spectral MLE: Top-k rank aggregation from pairwise comparisons. In *International Conference on Machine Learning*.

Conitzer, V. and Sandholm, T. (2005). Communication complexity of common voting rules. In *Proceedings of the 6th ACM conference on Electronic commerce*.

Dani, V., Hayes, T. P. and Kakade, S. M. (2008). Stochastic linear optimization under bandit feedback. In *Annual Conference Computational Learning Theory*.

de Borda, J.-C. (1781). Mémoire sur les élections au scrutin. *Histoire de l'Académie royale des sciences* .

Di, Q., Jin, T., Wu, Y., Zhao, H., Farnoud, F. and Gu, Q. (2024). Variance-aware regret bounds for stochastic contextual dueling bandits. In *The Twelfth International Conference on Learning Representations*.
URL https://openreview.net/forum?id=rDH7dIFn20

Dudík, M., Hofmann, K., Schapire, R. E., Slivkins, A. and Zoghi, M. (2015). Contextual dueling bandits. In *Conference on Learning Theory*, vol. abs/1502.06362. PMLR.

Dwork, C., Kumar, R., Naor, M. and Sivakumar, D. (2001). Rank aggregation methods for the web. In *Proc. 10th Int. Conf. World Wide Web*. ACM.

Even-Dar, E., Mannor, S. and Mansour, Y. (2002). Pac bounds for multi-armed bandit and markov decision processes. In *International Conference on Computational Learning Theory*. Springer.

Falahatgar, M., Hao, Y., Orlitsky, A., Pichapati, V. and Ravindrakumar, V. (2017a). Maxing and ranking with few assumptions. *Advances in Neural Information Processing Systems* **30**.

Falahatgar, M., Jain, A., Orlitsky, A., Pichapati, V. and Ravindrakumar, V. (2018). The limits of maxing, ranking, and preference learning. In *International conference on machine learning*. PMLR.

Falahatgar, M., Orlitsky, A., Pichapati, V. and Suresh, A. T. (2017b). Maximum selection and ranking under noisy comparisons. In *International Conference on Machine Learning*. PMLR.

Farrell, R. H. (1964). Asymptotic behavior of expected sample size in certain one sided tests. *The Annals of Mathematical Statistics* 36–72.

Faury, L., Abeille, M., Calauzènes, C. and Fercoq, O. (2020). Improved optimistic algorithms for logistic bandits. In *International Conference on Machine Learning*. PMLR.

Feige, U., Raghavan, P., Peleg, D. and Upfal, E. (1994). Computing with noisy information. *SIAM Journal on Computing* **23** 1001–1018.

Filippi, S., Cappe, O., Garivier, A. and Szepesvári, C. (2010). Parametric bandits: The generalized linear case. *Advances in Neural Information Processing Systems* **23**.

Fishburn, P. C., Fishburn, P. C. et al. (1979). *Utility theory for decision making*. Krieger NY.

Freedman, D. A. (1975). On tail probabilities for martingales. *the Annals of Probability* 100–118.

Guiver, J. and Snelson, E. (2009). Bayesian Inference for Plackett-Luce Ranking Models. In *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09, ACM, New York, NY, USA.

Hajek, B., Oh, S. and Xu, J. (2014). Minimax-optimal Inference from Partial Rankings. In *Advances in Neural Information Processing Systems 27*.

Heckel, R., Shah, N. B., Ramchandran, K. and Wainwright, M. J. (2019). Active ranking from pairwise comparisons and when parametric assumptions do not help. *The Annals of Statistics* **47** 3099–3126.

Heckel, R., Simchowitz, M., Ramchandran, K. and Wainwright, M. (2018). Approximate ranking from pairwise comparisons. In *International Conference on Artificial Intelligence and Statistics*. PMLR.

Herbrich, R., Minka, T. and Graepel, T. (2006). Trueskill™: a bayesian skill rating system. *Advances in neural information processing systems* **19**.

Hunter, D. R. (2004). Mm algorithms for generalized Bradley-Terry models. *The Annals of Statistics* **32** 384–406.

Jain, P., Netrapalli, P. and Sanghavi, S. (2013). Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*. ACM.

Jamieson, K., Katariya, S., Deshpande, A. and Nowak, R. (2015). Sparse dueling bandits. In *Artificial Intelligence and Statistics*. PMLR.

Jin, T., Wu, Y., Gu, Q. and Farnoud, F. (2025). Ranking with multiple oracles. In *In submission*.

Jin, T., Xu, P., Gu, Q. and Farnoud, F. (2020). Rank aggregation via heterogeneous thurstone preference models. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. *Advances in neural information processing systems* **26**.

Jun, K.-S., Bhargava, A., Nowak, R. and Willett, R. (2017). Scalable generalized linear bandits: Online computation and hashing. *Advances in Neural Information Processing Systems* **30**.

Jun, K.-S., Willett, R., Wright, S. and Nowak, R. (2019). Bilinear bandits with low-rank structure. In *International Conference on Machine Learning*. PMLR.

Katariya, S., Jain, L., Sengupta, N., Evans, J. and Nowak, R. (2018). Adaptive sampling for coarse ranking. In *International Conference on Artificial Intelligence and Statistics*. PMLR.

Kendall, M. (1948). *Rank Correlation Methods*. London: Griffin.

Kim, M., Farnoud, F. and Milenkovic, O. (2015). HyDRA: Gene prioritization via hybrid distance-score rank aggregation. *Bioinformatics* **31** 1034–1043.

Kim, Y., Yang, I. and Jun, K.-S. (2022). Improved regret analysis for variance-adaptive linear bandits and horizon-free linear mixture mdps. *Advances in Neural Information Processing Systems* **35** 1060–1072.

Komiyama, J., Honda, J. and Nakagawa, H. (2016). Copeland dueling bandit problem: Regret lower bound, optimal algorithm, and computationally efficient algorithm. In *International Conference on Machine Learning*. PMLR.

Kuleshov, V. and Precup, D. (2014). Algorithms for multi-armed bandit problems. *arXiv preprint arXiv:1402.6028* .

Kumar, A. and Lease, M. (2011). Learning to Rank from a Noisy Crowd. In *Proceedings of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval*. SIGIR '11, ACM, New York, NY, USA.

Lai, T. L., Robbins, H. et al. (1985). Asymptotically efficient adaptive allocation rules. *Advances in applied mathematics* **6** 4–22.

Lattimore, T. and Szepesvári, C. (2020). *Bandit algorithms*. Cambridge University Press.

Li, L., Lu, Y. and Zhou, D. (2017). Provably optimal algorithms for generalized linear contextual bandits. In *International Conference on Machine Learning*. PMLR.

Liu, C., Jin, T., Hoi, S. C. H., Zhao, P. and Sun, J. (2017). Collaborative topic regression for online recommender systems: an online and bayesian approach. *Machine Learning* **106** 651–670.

Lou, H., Jin, T., Wu, Y., Xu, P., Gu, Q. and Farnoud, F. (2022). Active ranking without strong stochastic transitivity. *Advances in neural information processing systems* .

Luce, R. D. (1959). *Individual Choice Behavior: A Theoretical Analysis*. John Wiley & Sons, Inc., New York.

Maystre, L. and Grossglauser, M. (2017). Just sort it! a simple and effective approach to active preference learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

Merkle, M. (1998). Convolutions of logarithmically concave functions. *Publikacije Elektrotehničkog fakulteta. Serija Matematika* 113–117.

Minka, T. P., Cleven, R. and Zaykov, Y. (2018). Trueskill 2: An improved bayesian skill rating system. In *Microsoft Research*.

Mohajer, S., Suh, C. and Elmahdy, A. (2017). Active learning for top-k rank aggregation from noisy comparisons. In *International Conference on Machine Learning*. PMLR.

Mukherjee, S., Naveen, K. P., Sudarsanam, N. and Ravindran, B. (2017). Efficient-ucbv: An almost optimal algorithm using variance estimates. In *AAAI Conference on Artificial Intelligence*.

Negahban, S., Oh, S. and Shah, D. (2012). Iterative ranking from pair-wise comparisons. In *Advances in Neural Information Processing Systems*, vol. 25.

Negahban, S., Oh, S. and Shah, D. (2016). Rank Centrality: Ranking from Pairwise Comparisons. *Operations Research* **65**.

Negahban, S. and Wainwright, M. J. (2012). Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. *Journal of Machine Learning Research* **13** 1665–1697.

Nesterov, Y. (2003). *Introductory lectures on convex optimization: A basic course*, vol. 87. Springer Science & Business Media.

Niu, S., Lan, Y., Guo, J., Cheng, X., Yu, L. and Long, G. (2015). Listwise approach for rank aggregation in crowdsourcing. In *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*.

Piech, C., Huang, J., Chen, Z., Do, C., Ng, A. and Koller, D. (2013). Tuned models of peer assessment in moocs. *arXiv preprint arXiv:1307.2579* .

Ramamohan, S., Rajkumar, A. and Agarwal, S. (2016). Dueling bandits: Beyond condorcet winners to general tournament solutions. In *NIPS*.

Raman, K. and Joachims, T. (2014). Methods for ordinal peer grading. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

Raman, K. and Joachims, T. (2015). Bayesian ordinal peer grading. In *Proceedings of the Second (2015) ACM Conference on Learning*.

Ren, W., Liu, J. and Shroff, N. B. (2018). Pac ranking from pairwise and listwise queries: Lower bounds and upper bounds. *arXiv preprint arXiv:1806.02970* .

Ren, W., Liu, J. and Shroff, N. B. (2019). On sample complexity upper and lower bounds for exact ranking from noisy comparisons. In *Neural Information Processing Systems*, vol. 32.

Rusmevichientong, P. and Tsitsiklis, J. N. (2010). Linearly parameterized bandits. *Mathematics of Operations Research* **35** 395–411.

Saad, E. M., Verzelen, N. and Carpentier, A. (2023). Active ranking of experts based on their performances in many tasks. In *International Conference on Machine Learning*.

Saha, A. (2021). Optimal algorithms for stochastic contextual preference bandits. In *Neural Information Processing Systems*.

Saha, A. and Gopalan, A. (2019). Active ranking with subset-wise preferences. In *The 22nd International Conference on Artificial Intelligence and Statistics*. PMLR.

Saha, A., Koren, T. and Mansour, Y. (2021). Adversarial dueling bandits. In *International Conference on Machine Learning*, vol. abs/2010.14563. PMLR.

Shah, N. B. and Wainwright, M. J. (2017). Simple, robust and optimal ranking from pairwise comparisons. *The Journal of Machine Learning Research* **18** 7246–7283.

Slivkins, A. et al. (2019). Introduction to multi-armed bandits. *Foundations and Trends® in Machine Learning* **12** 1–286.

Sui, Y. and Burdick, J. (2014). Clinical online recommendation with subgroup rank feedback. In *Proceedings of the 8th ACM conference on recommender systems*.

Szörényi, B., Busa-Fekete, R., Paul, A. and Hüllermeier, E. (2015). Online rank elicitation for plackett-luce: A dueling bandits approach. *Advances in Neural Information Processing Systems* **28** 604–612.

Takanobu, R., Zhuang, T., Huang, M., Feng, J., Tang, H. and Zheng, B. (2019). Aggregating e-commerce search results from heterogeneous sources via hierarchical reinforcement learning. In *The World Wide Web Conference*.

Thurstone, L. L. (1927). A law of comparative judgment. *Psychological Review* **34** 273–286.

Tropp, J. A. (2012). User-friendly tail bounds for sums of random matrices. *Foundations of computational mathematics* **12** 389–434.

Tversky, A. (1969). Intransitivity of preferences. *Psychological review* **76** 31.

Tversky, A. and Kahneman, D. (1981). The framing of decisions and the psychology of choice. *Science* .

Urvoy, T., Clérot, F., Féraud, R. and Naamane, S. (2013). Generic exploration and k-armed voting bandits. In *ICML*.

Vasile, F., Smirnova, E. and Conneau, A. (2016). Meta-prod2vec: Product embeddings using side-information for recommendation. In *Proceedings of the 10th ACM conference on recommender systems*.

Vojnovic, M. and Yun, S. (2016). Parameter Estimation for Generalized Thurstone Choice Models. In *PMLR*. PMLR.

Wang, J., Huang, P., Zhao, H., Zhang, Z., Zhao, B. and Lee (2018). Billion-scale commodity embedding for e-commerce recommendation in alibaba. *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* .

Wang, S., Huang, S., Liu, T.-Y., Ma, J., Chen, Z. and Veijalainen, J. (2016). Ranking-oriented collaborative filtering: A listwise approach. *ACM Transactions on Information Systems (TOIS)* **35** 1–28.

Wang, Z., Gu, Q., Ning, Y. and Liu, H. (2015). High dimensional em algorithm: Statistical optimization and asymptotic normality. In *Advances in neural information processing systems*.

Wauthier, F., Jordan, M. and Jojic, N. (2013). Efficient Ranking from Pairwise Comparisons. In *PMLR*.

Weng, R. C. and Lin, C.-J. (2011). A Bayesian approximation method for online ranking. *Journal of Machine Learning Research* **12** 267–300.

Wu, H. and Liu, X. (2016). Double thompson sampling for dueling bandits. *Advances in neural information processing systems* **29**.

Wu, Y., Jin, T., Di, Q., Lou, H., Farnoud, F. and Gu, Q. (2024). Borda regret minimization for generalized linear dueling bandits. In *Proceedings of the 41st International Conference on Machine Learning*. ICML'24, JMLR.org.

Wu, Y., Jin, T., Lou, H., Xu, P., Farnoud, F. and Gu, Q. (2022). Adaptive sampling for heterogeneous rank aggregation from noisy pairwise comparisons. In *International Conference on Artificial Intelligence and Statistics*. PMLR.

Xu, P., Ma, J. and Gu, Q. (2017a). Speeding up latent variable Gaussian graphical model estimation via nonconvex optimization. In *Advances in Neural Information Processing Systems*.

Xu, P., Zhang, T. and Gu, Q. (2017b). Efficient algorithm for sparse tensor-variate Gaussian graphical models via gradient descent. In *Artificial Intelligence and Statistics*.

Yu, P. L. H. (2000). Bayesian analysis of order-statistics models for ranking data. *Psychometrika* **65** 281–299.

Yue, Y., Broder, J., Kleinberg, R. and Joachims, T. (2012). The k-armed dueling bandits problem. *Journal of Computer and System Sciences* **78** 1538–1556.

Yue, Y. and Joachims, T. (2009). Interactively optimizing information retrieval systems as a dueling bandits problem. In *Proceedings of the 26th Annual International Conference on Machine Learning*.

Yue, Y. and Joachims, T. (2011). Beat the mean bandit. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. Citeseer.

Zermelo, E. (1929). Die Berechnung der Turnier-Ergebnisse als ein Maximumproblem der Wahrscheinlichkeitsrechnung. *Mathematische Zeitschrift* **29** 436–460.

Zhang, X., Li, G. and Feng, J. (2016). Crowdsourced top-k algorithms: An experimental evaluation. *Proc. VLDB Endow.* **9** 612–623.

Zhang, X., Wang, L. and Gu, Q. (2018). A unified framework for nonconvex low-rank plus sparse matrix recovery. In *International Conference on Artificial Intelligence and Statistics*.

Zhang, Z., Yang, J., Ji, X. and Du, S. S. (2021). Improved variance-aware confidence sets for linear bandits and linear mixture mdp. *Advances in Neural Information Processing Systems* **34** 4342–4355.

Zhao, H., He, J., Zhou, D., Zhang, T. and Gu, Q. (2023a). Variance-dependent regret bounds for linear bandits and reinforcement learning: Adaptivity and computational efficiency. *arXiv preprint arXiv:2302.10371* .

Zhao, H., Zhou, D., He, J. and Gu, Q. (2023b). Optimal online generalized linear regression with stochastic noise and its application to heteroscedastic bandits. In *International Conference on Machine Learning*. PMLR.

Zhao, Z., Villamil, T. and Xia, L. (2018). Learning mixtures of random utility models. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Zhou, D. and Gu, Q. (2022). Computationally efficient horizon-free reinforcement learning for linear mixture mdps. *Advances in neural information processing systems* **35** 36337–36349.

Zhou, D., Gu, Q. and Szepesvari, C. (2021). Nearly minimax optimal reinforcement learning for linear mixture markov decision processes. In *Conference on Learning Theory*. PMLR.

Zhu, J., Ahmed, A. and Xing, E. P. (2012). Medlda: maximum margin supervised topic models. *J. Mach. Learn. Res.* **13** 2237–2278.

Zhu, R., Wang, L., Zhai, C. and Gu, Q. (2017). High-dimensional variance-reduced stochastic gradient expectation-maximization algorithm. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

Zoghi, M., Karnin, Z. S., Whiteson, S. and De Rijke, M. (2015). Copeland dueling bandits. *Advances in neural information processing systems* **28**.

Zoghi, M., Whiteson, S., Munos, R. and Rijke, M. (2014). Relative upper confidence bound for the k-armed dueling bandit problem. In *International conference on machine learning*.