TOWARDS COMPREHENSIVE CLUSTER-INDUCED METHODS FOR RECOMMENDER SYSTEMS

Thesis approved by
the Department of Computer Science
Technische Universität Kaiserslautern
for the award of the Doctoral Degree
Doctor of Natural Sciences (Dr. rer. nat.)

to

Rodrigo Augusto da Silva Alves

Date of defense: February 1st, 2022

Dean: Prof. Dr. Jens Schmitt

Ph.D. committee chair: Prof. Dr. Sebastian Michel (TU Kaiserslautern, Germany)

Reviewer: Prof. Dr. Marius Kloft (TU Kaiserslautern, Germany)

Reviewer: Prof. Dr. Pedro Olmo Stancioli Vaz de Melo (UFMG, Brazil)

DE-386

Rodrigo Augusto da Silva Alves

2021

Advisor: Prof. Dr. Marius Kloft

ABSTRACT

Recommender systems recommend items (e.g., movies, products, books) to users. In this thesis, we proposed two comprehensive and cluster-induced recommendation-based methods: *Orthogonal Inductive Matrix Completion (OMIC)* and *Burst-induced Multi-armed Bandit (BMAB)*. Given the presence of side information, the first method is categorized as context-aware. OMIC is the first matrix completion method to approach the problem of incorporating biases, side information terms and a pure low-rank term into a single flexible framework with a well-principled optimization procedure. The second method, BMAB, is context-free. That is, it does not require any side data about users or items. Unlike previous context-free multi-armed bandit approaches, our method considers the temporal dynamics of human communication on the web and treats the problem in a continuous time setting. We built our models' assumptions under solid theoretical foundations. For OMIC, we provided theoretical guarantees in the form of generalization bounds by considering the distribution-free case: no assumptions about the sampling distribution are made. Additionally, we conducted a theoretical analysis of community side information when the sampling distribution is known and an adjusted nuclear norm regularization is applied. We showed that our method requires just a few entries to accurately recover the ratings matrix if the structure of the ground truth closely matches the cluster side information. For BMAB, we provided regret guarantees under mild conditions that demonstrate how the system's stability affects the expected reward. Furthermore, we conducted extensive experiments to validate our proposed methodologies. In a controlled environment, we implemented synthetic data generation techniques capable of replicating the domains for which OMIC and BMAB were designed. As a result, we were able to analyze our algorithms' performance across a broad spectrum of ground truth regimes. Finally, we replicated a real-world scenario by utilizing well-established recommender datasets. After comparing our approaches to several baselines, we observe that they achieved state-of-the-art results in terms of accuracy. Apart from being highly accurate, these methods improve interpretability by describing and quantifying features of the datasets they characterize.

ZUSAMMENFASSUNG

Empfehlungssysteme empfehlen den Benutzern Artikel (z.B. Filme, Produkte, Bücher). In dieser Arbeit stellen wir zwei umfangreiche und cluster-induzierte empfehlungsbasierte Methoden vor: *Orthogonal Inductive Matrix Completion (OMIC)* und *(Burst-induced Multi-armed Bandit)*. Angesichts der vorhandenen Nebeninformationen wird die erste Methode als kontextbewusst eingestuft. OMIC ist die erste Methode zur Matrixvervollständigung, die das Problem der Einbeziehung von Verzerrungen, Nebeninformationen und einem rein niederrangigen Term in einem einzigem flexiblen Framework mit einem fundierten Optimierungsverfahren angeht. Die zweite Methodik, BMAB, ist kontextfrei. Das bedeutet, dass sie keine Nebendaten über Nutzer oder Artikel benötigt.Im Gegensatz zu vorherigen kontextfreien Ansätzen mit Multi-armed Bandits, berücksichtigt unsere Methode die temporale Dynamik menschlicher Kommunikation im Internet und behandelt das Problem in einem kontinuierlichen Zeitrahmen.Wir haben die Annahmen unserer Modelle auf eine solide theoretische Grundlage gestellt. Indem wir den verteilungsfreien Fall betrachtet haben, haben wir für OMIC theoretische Garantien in Form von Generalisierungsschranken aufgestellt. Für den Fall in dem die Datenverteilung bekannt ist und eine angepasste Kernnormregularisierung angewendet wird, haben wir darüber hinaus eine theoretische Analyse der Gemeinschaftsseiteninformationen durchgeführt. Wir haben gezeigt, dass unsere Methode nur wenige Matrixeinträge benötigt, um die Bewertungsmatrix genau wiederherzustellen, wenn die Struktur der Grundwahrheit (ground truth) nahezu mit der Seiteninformation des Clusters übereinstimmt. Für BMAB haben wir unter milden Bedingungen Rückgewinnungsgarantien entwickelt, die zeigen, wie die Stabilität des Systems die erwartete Belohnung beeinflusst. Des Weiteren haben wir umfangreiche Experimente durchgeführt, um unsere vorgestellten Methoden zu validieren. In einer kontrollierten Umgebung haben wir Techniken zur Erzeugung synthetischer Daten implementiert, die in der Lage sind, die Domänen zu replizieren, für die OMIC und BMAB entwickelt wurden. Nach dem Vergleich unserer Ansätze mit verschiedenen Basisverfahren stellen wir fest, dass sie in Bezug auf Genauigkeit die besten Ergebnisse erzielen und einen neuen Stand der Technik darstellen. Abgesehen von ihrer hohen Genauigkeit verbessern diese Methoden die Interpretierbarkeit durch eine Beschreibung und Quantifizierung von Merkmalen der charakterisierten Datensätze.

Dedico este trabalho à minha mãe.

ACKNOWLEDGMENTS

I really hope this is not the end. I have met so many wonderful people on this journey that I want to keep them in my life forever. Time to express gratitude. I cannot nominally thank who has helped me on this path, so I will often consider comprehensive clusters.

First of all, I would like to express my sincere gratitude to my advisor, Prof. Marius Kloft. Thank you very much for our scientific debates, for introducing fascinating problems, and for creating a perfect atmosphere for my research (with countless resources). I am also glad for our friendship and the many occasions when we were together outside of the academic environment: our shared passion for football, bowling, bike trips, and drinks, among other things. One day we will go to the Maracanã to watch Brazil vs. Germany together!

I would like to thank Prof. Sebastian Michel and Prof. Pedro Vaz de Melo for the relevant comments as members of the Ph.D. committee.

I also would like to show my appreciation to Antoine Ledent, my research collaborator and best friend. Thank you for being my buddy in both good and bad times. To share so many rejections until we reached a point where we could calibrate it. For the discussion of everything and nothing at the same time (often a bit drunk). And for being who you are.

I must mention the Machine Learning group, which is represented here by Naghmeh and Waleed (my officemates). Being colleagues at such a difficult period makes us friends for life. You may count on me indefinitely, all of you. Simply contact me; I will be available at all times.

Regarding the whole university personnel, I would like to thank Heike Neu, the secretary of the Machine Learning group. I will always remember you: you are an example of professionalism and kindness.

Additionally, I would like to convey my heartfelt appreciation to the Federal Republic of Germany, particularly to the taxpayers of the state of Rhineland-Palatinate, for the funding.

Eu gostaria de agradecer de coração à minha família. À minha mãe, Fátima, ao meu Pai, José (em memória) e aos meus irmãos, Thiago e Diogo, por compreenderem este momento que passamos longe e por estarem presentes em todas as etapas da minha vida. E à Veronika, pelo carinho, compreensão e pela dedicação.

Finally, to all those who are part of “the projects of the future”, since they are the reason I could be part of the present.

Table of Contents

Page
Abstract . . . ii
Zusammenfassung . . . iii
Dedication . . . iv
Acknowledgments . . . v
List of Figures . . . ix
List of Abbreviations . . . xii

Chapters

1 Introduction **1**
1.1 Author's Ph.D. Thesis . . . 3
1.2 Organization of this dissertation and the author's contributions . . . 5
1.2.1 Part One: Context-aware recommendation . . . 5
1.2.2 Part Two: Context-free and cold-start methods . . . 8

Part One: Context-aware recommendation

2 Methods, optimization and algorithms **13**
2.1 Description of the model and optimization procedure . . . 15
2.1.1 Modelling jointly trained user/item biases . . . 16
2.1.2 Modelling jointly trained cluster side information . . . 17
2.1.3 A comprehensive jointly trained user/item biases and cluster side information method . . . 18
2.2 Recovering inductive clusters via matrix completion . . . 19
2.3 The OMIC algorithm . . . 20
2.3.1 A scalable version for OMIC algorithm . . . 22
2.3.2 Complexity and runtime analysis . . . 27
2.3.3 Convergence guarantees . . . 29
2.4 Collaborative Clustering Algorithm . . . 31
2.4.1 Hyperparameter selection and scalability . . . 33
2.5 Conclusion . . . 34

3 Theoretical analysis **35**
3.1 Uniqueness of decomposition of the predictor . . . 36
3.2 Learning-theoretical guarantees under distribution-free sampling . . . 37

3.2.1 Generalization bounds . . . 37
3.2.2 Proof of the generalization bounds . . . 38
3.3 Learning-theoretical guarantees under a given distribution sampling . . . 44
3.3.1 Adjusted nuclear norm regulariser . . . 44
3.3.2 Generalization bounds . . . 45
3.3.3 Proof of the generalization bounds . . . 46
3.4 Remarks on proof techniques and comparison to IMC bounds . . . 48
3.5 Conclusion . . . 49

4 Experiments and applications 51
4.1 Metrics . . . 52
4.2 Baselines . . . 53
4.3 Synthetic experiments . . . 54
4.3.1 User and item biases generation procedure . . . 54
4.3.2 Cluster side information generation procedure . . . 55
4.3.3 Experimental setup and results . . . 56
4.4 OMIC as a recommender system . . . 59
4.4.1 Datasets . . . 59
4.4.2 Collaborative-filtering recommendation . . . 60
4.4.3 Illustration of OMIC's interpretability . . . 62
4.5 Predicting activity coeffients via matrix completion . . . 64
4.5.1 Dataset . . . 64
4.5.2 Baselines . . . 65
4.5.3 Experimental results . . . 66
4.6 Conclusion . . . 68

5 Related work and discussion 69
5.1 Related work on inductive matrix completion . . . 69
5.2 Related work on community discovery . . . 69
5.3 Related work on the theory of matrix completion . . . 70
5.4 Related work on prediction of activity coefficients . . . 71

Part Two: Context-free recommendation

6 Methods, optimization and algorithms 74
6.1 Description of the model and problem formulation . . . 76
6.1.1 Temporal dynamics assumptions . . . 76
6.1.2 Reward distribution assumptions . . . 78
6.2 Burst-induced multi-armed bandit . . . 79
6.2.1 Thompson sampling algorithm . . . 79
6.2.2 The BMAB algorithm . . . 80
6.2.3 BMAB regret guarantees . . . 84
6.2.4 A realistic state detector . . . 86
6.3 Disentangling loyal and curious audiences via an EM-approach . . . 88
6.3.1 Self-feeding process . . . 88
6.3.2 Notation and formal EM construction . . . 89
6.3.3 E-STEP . . . 91

6.3.4 M-STEP 92
6.3.5 Algorithmic details 93
6.3.6 Warm-starts for θ and parameter initialization 94
6.3.7 Complexity analysis 94
6.4 Conclusion 95

7 Experiments and applications **96**
7.1 Context-free recommendation 97
7.1.1 Baselines and parameter selection 97
7.1.2 Generation of synthetic audience dynamics 98
7.1.3 Synthetic data experiments 99
7.1.4 Real-world data experiments 100
7.2 Characterizing online timeseries 102
7.2.1 Synthetic data generation and evaluation 102
7.2.2 Baseline models and fitting metrics for real datasets 103
7.2.3 Data Sets 104
7.2.4 Fitting and characterization of real-world datasets 106
7.2.5 Absolute and relative loyalty 107
7.3 Conclusion 108

8 Related Work and Discussion **109**
8.1 Related work on multi-armed bandits 109
8.2 Related work on the dynamics of human communication 110

9 Conclusion **112**
9.1 Summary of Results 113
9.2 Future work 114

Appendix A The OMIC algorithm convergence guarantees **116**
A.1 The fully-known case 116
A.2 Convergence guarantees 117

Bibliography **126**
Curriculum Vitae 139

List of Figures

Figure **Page**

1.1 Two examples of time series factorization that motivate our method. **Left:** loyal and curious systems' audiences. **Right:** activity related to Psy and David Bowie. The data is taken from Google Trend (Jan/2008 to Dec/2020; country: USA; search engine: Youtube). We give a detailed description of the results in the main text. 8

2.1 Visualization of orthogonal inductive matrix completion (specifically, the comprehensive version explained in Section 2.1.3). The model is a sum of matrix terms, each of the form $XMY^\top$. It means each combination of X and Y gives rise to a term in the sum. We interpret the magnitude of this term as its relevance to the prediction. 16

2.2 Runtime comparison of main computations required for one iteration of Soft-impute and OMIC. The red line is the identity. 27

4.1 Summary of synthetic data simulations results. The first graph shows the relationship between all combinations of the parameters $(\alpha, \gamma, p_\Omega)$ and the RMSE. The second one shows how $(\alpha, \gamma, p_\Omega)$ influences the correct recovery of user and item biases ((UBD+IBD)/2). Each box plot in the first two graphs is obtained from 50 simulations. The third graph displays the distribution of the MBD over all of the simulations. 56

4.2 Rand index distribution grouped by the parameters $alpha$, β and $\mathbb{P}_\Omega$ . . . 57

4.3 The distribution of the number of our model detected clusters according to the parameters α, β and $\mathbb{P}_\Omega$ 58

4.4 ϕ distribution grouped by the parameters α, β and $\mathbb{P}_\Omega$ 58

4.5 The first two graphs show the relative influence of the components on the predictions of the whole matrix and one individual entry, respectively. The last two graphs show the distribution of the users and item biases obtained by the Douban dataset. 62

4.6 Affinity between user communities (grouped by gender and age) and movie genres for MovieLens. These biases can be directly read from the component $X^{(2)}M^{(2,2)}(Y^{(2)})^\top$ in the instance of Section 2.1.3. 63

4.7 Probabilistic graph of our model. Each activity coefficient in R_{ij} is distributed as a Cauchy distribution with scale λ and centered around the inner product of u_i and v_j. Each solute (solvent) feature vector v_i (u_i) is distributed as a multivariate normal distribution with zero mean vector $\mathbf{0}_k$ and co-variance matrix Σ_u (Σ_v). 65
4.8 Visualization of orthogonal inductive matrix completion for the prediction of activity coefficients. The model is a sum of explainable matrix terms. . . . 67

6.1 Difference between audiences, system's states and user-activity category. **First graph**: jointly activity of the loyal and curious audiences (gray curve, mixture of a HPP and a PW-HPP). The actual user-activity categories are shown by the dots on the first graph: blue for loyal *category*, and orange for the curious *category*. The jointly activity is disentangled in the following two graphs. **Second graph**: HPP (green curve), models the loyal *audience*. **Third graph**: PW-HPP (yellow curve), models the curious *audience*. In all graphs, the vertical lines indicate the actual state transitions (i.e., the set M) and the colored stripes at the background represents the system's states: blue for loyal *state*, and orange for the curious *state*. 77
6.2 A mixture of point processes in timelines. The *curious* (PW-HPP) and *loal audience* (HPP) labels, as well as the transitions (events of M), are not observed. 78
6.3 Realization of the BMAB algorithm in the mixture of point processes presented in Figure 6.1 ($K = 3$; Loyal state: $[\theta_1, \theta_2, \theta_3] = [0.3, 0.4, 0.5]$; and Curious state: $[\theta_1, \theta_2, \theta_3] = [0.3, 0.9, 0.5]$ (in $(m_1, m_2]$) and $[\theta_1, \theta_2, \theta_3] = [0.9, 0.4, 0.5]$ (in $(m_3, m_4]$)). **First row of graphs**: prior distribution of θ_1, θ_2, and θ_3 at the times $\{0, m_1, m_2, m_3, m_4, t_N = 100\}$ concerning the loyal state. **Second row of graphs**: prior distribution of θ_1, θ_2, and θ_3 at the times $\{0, m_1, m_2, m_3, m_4, t_N = 100\}$ concerning the curious state. 81
6.4 Forgetting procedure of the BMAB algorithm: in the interval $(m_2, m_3]$ (loyal state active) of the the mixture of point processes presented in Figure 6.1 ($K = 3$; Curious state: $[\theta_1, \theta_2, \theta_3] = [0.3, 0.9, 0.5]$ (in $(m_1, m_2]$)). Prior distribution of θ_1, θ_2, and θ_3 at the times $\{m_2, m_2^5, m_2^{10}, m_2^{15}, m_3\}$ concerning the curious state. 83
6.5 **First two graphs:** comparison between theoretical guarantees and experimental results of the regret ($\mathcal{R}(T)$). The red line is the function $f(x) = x$. The second line shows the function $f(x) = ax + b$, where a and b are the coefficients of the linear regression fitting of $\mathcal{R}(T)$ versus $O(\mathcal{R}(T))$. **Last graph:** boxplot of the dispersion of the state detector accuracy. 86
6.6 State detector and activity categorization. **First graph**: optimal detector scenario. **Three last graphs:** point-wise user-activity categorization (loyal and curious) by using the detector proposed in Section 6.2.4 (we fixed $\delta = 0.95$ and varied $\Delta \in \{5, 10, 15\}$). Red dots means wrong categorization. In all graphs, the vertical lines indicate the actual state transitions (i.e., the set M), the colored stripes at the background represents the system's states detection and the user-activity categories are shown by the dots. 87

6.7 Approaching an SFP by a PW-HPP. **First row of graphs:** three different SFP genarations with $mu = 0.1$ and $t \in (0, 100]$; **Second row of graphs:** approximation of the SFP above by a PW-HPP. The vertical lines on the second row of graphs represents the Poisson transitions 89

6.8 Mixture of point process in timelines for our EM approach. The *curious* (PW-HPP) and *loal audience* (HPP) labels, as well as the transitions (events of M), are not observed. 90

7.1 Summary of the results of the synthetic data experiments. The graphs are organized according to the tuple $\{\mathbb{E}[N], P_H, n_b, b\}$. The green line ("Optimal") is the theoretical reward of the best possible algorithm. Each data point corresponds to the average of 20 simulations. 99

7.2 Summary of the results of our experiments with synthetic data. $\delta(\mu, \hat{\mu})$ (top, yellow) and $\delta(\lambda_L, \hat{\lambda}_L)$ (bottom, green) distributions grouped by $(\mathbb{E}(P_H))$. The x-axis shows the expected number of observed events. 103

7.3 **Top:** fitting performance (determination coefficient) as a function of the proportion $P_H = |PP|/n$ of Poisson events per dataset. **Bottom:** distribution of the proportion $P_H = |PP|/n$ per dataset. 106

7.4 Left: Scatterplot of the average $\tilde{\kappa}$ versus the average κ for each of the real datasets. Right: Dispersion of the indicators κ and $\tilde{\kappa}$ per datasets. 107

List of Abbreviations

B-SI SoftImpute with bias. 53, 56, 57

BMAB Burst-induced Multi-Armed Bandit. 3, 9–11, 74, 75, 79, 80, 82, 83, 85, 90, 95–97, 108, 112, 114

DDB Dortmund Data Bank. 64, 66, 71

EM Expectation–maximization. 10, 75, 88, 90, 91, 95, 104, 114

HPP Homogeneous Poisson process. 10, 76–78, 85, 88–92, 95, 102–104, 107, 108

IBD Item Bias Deviation. 52, 53, 57

IMC Inductive Matrix Completion. 5, 6, 48, 53, 69, 71, 114

IMCNF Inductive Matrix Completion with Noisy Features. 53, 60

MAB multi-armed bandit. 2, 8, 9, 74, 75, 79, 82, 95, 97, 109–111, 115

MAD Mean Absolute Deviation. 52, 53, 66

MBD Matrix Bias Deviation. 52, 53, 57

OMIC Orthogonal Inductive Matrix Completion. 3, 5–7, 16, 19, 22, 27, 29, 31, 33, 35, 49, 51, 56, 57, 61, 63, 66, 68, 112, 113

PW-HPP Piece-Wise Homogeneous Poisson process. 76–78, 88–90, 115

RI Random Index. 53, 61

RMSE Root-mean-square Error. 52, 53, 57, 61, 66, 113

RS Recommender System. 1, 2, 7–9, 74, 100

RSE Random Series of Events. 10, 75

RSs Recommender Systems. 1, 2, 9, 22

SFP Self-feeding process. 10, 88–92, 94, 95, 102–104, 107

SI SoftImpute. 53, 56, 57, 60

SOTA state-of-the-art. 3, 7, 11, 68, 97, 112, 113

SPC Spearman Correlation. 52, 53, 61, 113

SVD Singular Value Decomposition. 23, 27, 28, 69

TS Thompson sampling. 79, 82, 84, 102, 109, 114

UBD User Bias Deviation. 52, 53, 57

UCB Upper Confidence Bounds. 109

Chapter 1

INTRODUCTION

Recommender Systems (RSs) recommend items (e.g., movies, products, books) to users. The first Recommender System (RS)'s research was published in 1998 [1]. The authors investigated an *automatic agent* to find relevant papers and recommend them to researchers. Since then, hundreds of works have been published and learning-based recommendation has become a fundamental machine-learning task. Academia's interest in RSs is driven by success in recommenders' applications. Specialists project that, by 2025, the recommendation engine market size will reach over $12 billion, more than ten times its market value in the year of 2018 [2].

In practical applications, the information about users (or items) is frequently available in the form of clusters (categorical attributes) such as gender, nationality, occupation (or genres, brands, authors, respectively) [3–6]. For instance, this type of side information, also entitled as communities in the literature, is extensively used to improve RSs' performance in terms of accuracy enhancement [6], interpretability [7], and scalability [8]. When community information is not directly observed, it can be recovered, as has been attempted in a series of works [9, 10].

Despite the significant progress made by cluster-induced approaches in recommendation systems, there are still a variety of unanswered research concerns. For instance, consider the learning algorithms for matrix completion, which is the classical problem of recovering the missing entries of a partially observed matrix. Since the *NetFlix Prize* [11], matrix completion became a standard method for performing the recommendation task. In RS, the rows correspond to users and the columns correspond to items, with each entry $\{i, j\}$ corresponding to the rating of the user i to item j. In practical applications, the following refinements have proven helpful for enhancing the performance of classic matrix completion methods:

(a) *Incorporating biases* [12, 13]: some users may generally be more critical than others. This means that, generally, they tend to rate items lower than other users. Furthermore, certain items are fundamentally superior to others. Thus they receive higher ratings;

(b) *Incorporating side information*: there is plenty of items' side information on the web, and one may have access to user attributes, from which we can derive clusters (community side information) [14].

Previous approaches to incorporate side information do not explore communities' orthogonality properties. Orthogonal constraints can benefit matrix completion techniques in a myriad of aspects, including optimization and interpretability. Moreover, no comprehensive method combines refinements (a) and (b) with a pure low-rank term in a single joint-trained approach.

One can find another lack of comprehensiveness in the RS's research by analyzing the methodologies used to solve the especially challenging case where there is not only a lack of information about user preferences and behavior, but of *any* usable side information. Without the ability to profile users and items densely, a RS can rely only on recent user-item interaction [15, 16]. A popular option is to model the problem as a context-free multi-armed bandit (MAB) problem [17]: at each trial (*new* user requisition), the gambler (RS) selects an arm (an item) to pull (to recommend) and observes a reward (a click or lack thereof). Throughout the event history, an algorithm improves the policy to maximize the reward (e.g., the number of clicks). The standard MAB setting is inadequate in practice because it assumes that the (unknown) item popularity distribution is stationary [18–20]. Assuming that the items' popularity is *not* changing over time is highly unrealistic [16]. Therefore, prior research [21–25] has dealt with the shifts in the items' popularity by modeling context-free and non-stationary MABs. They constructed shift detection methods to categorize the users' activities into stationary segments based *solely* on the observation of the reward distribution. However, rather than actively exploiting the continuous temporal activity of RSs, they model the problem as a discrete-time. This restricts previous approaches, as they are unable to fully explore the complex dynamics of human communication (e.g., the presence of bursts in recommendation requisitions) commonly observed in RSs' activities [16, 26]. Modeling the time in a continuous configuration provides recommendation algorithms the ability to recognize activity patterns that are connected to item popularity [27].

My research is guided to mitigate the incompleteness of the RSs' methods in the above scenarios. As a consequence, I have established and developed methodologies that help advance the area of recommender systems. The following thesis, which is detailed in the next section, describes this dissertation's main hypothesis.

1.1 Author's Ph.D. Thesis

This dissertation concerns the validation of the following thesis:

> "Orthogonal Inductive Matrix Completion (OMIC) and Burst-induced Multi-Armed Bandit (BMAB), two comprehensive and cluster-induced recommendation methods that I developed and which enjoy favorable theoretical guarantees, significantly contribute to the field of learning-based recommender systems *respectively* in the (1) context-aware and (2) context-free scenarios. The methods exhibit better performance than the state-of-the-art (SOTA), as demonstrated experimentally in the thesis through extensive synthetic and real-world data experiments. Besides being highly accurate, these methods present the added benefit of improved interpretability by describing and quantifying features of the datasets that they characterize."

Here, *context-aware* are those methods that perform recommendation in the presence of an additional user and/or item side information, whilst *context-free* methods do not have any side information as an input. In addition, we define a cluster as a collection of related components (e.g., user, items, user-activity, etc.) that were not necessarily obtained using unsupervised learning methods.

Previously Published Work

This dissertation is based on the following *selected* publications.

The core framework was published in:

[1] Antoine Ledent*, **Rodrigo Alves***, and Marius Kloft. Orthogonal inductive matrix completion. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

[2] **Rodrigo Alves**, Antoine Ledent, and Marius Kloft. Burst Induced Multi-armed Bandit for Learning Recommendation. *Proceedings of the 15th ACM RecSys Recommender System Conference*, 2021.

The applications of matrix completion were discussed in:

[3] **Rodrigo Alves***, Antoine Ledent*, Renato Assunção, and Marius Kloft. An empirical study of the discreteness prior in low-rank matrix completion. In *Proceedings of Machine Learning Research*(1), volume 148: Pre-registration in ML, pages 111–125, 2021.

[4] Fabian Jirasek*, **Rodrigo Alves***, Julie Damay*, Robert Vandermeulen, Robert Bamler, Michael Bortz, Stephan Mandt, Marius Kloft, and Hans Hasse. Machine learning in thermodynamics: Prediction of activity coefficients by matrix completion. *The journal of physical chemistry letters*, 11(3):981–985, 2020.

Bounds for matrix completion with adjusted nuclear norm regulariser were discussed in:

[5] Antoine Ledent, **Rodrigo Alves**,Yunwen Lei, and Marius Kloft. Fine-grained Generalisation Analysis of Inductive Matrix Completion. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021 (**To appear**).

Methods for disentangling loyal and ephemeral audiences in online time series were proposed in:

[6] **Rodrigo Alves***, Antoine Ledent*, Renato Assunção, Pedro Vaz de Melo, and Marius Kloft. Are you here to stay? Disentangling the loyal audience from the curious on social media. **Submitted to** *ACM Transactions on Knowledge Discovery from Data (TKDD)*, Aug 2021.

[7] **Rodrigo Alves**, Renato Assunção, and Pedro Vaz de Melo. Burstiness scale: A parsimonious model for characterizing random series of events. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1405–1414, 2016.

[6] builds on [7]. The model proposed in [6] is entirely novel, with far fewer approximations and is both more efficient and effective. [7] is based on my Master's thesis, with significant refinements made after I received my degree.

Organization of the dissertation: The content of this dissertation is related to the above publications in the following way:

Part One: Context-aware Recommendation – **Chapter 2** is based on [1] and [3]; **Chapter 3** contains material from [1] and [5]; **Chapter 4** is based on [1], [3] and [4];

Part Two: Context-free Recommendation – **Chapter 6** is based on [2], [6] and [7]; **Chapter 7** is based on [2] and [6].

Chapter 5 (resp. **Chapter 8**) contains discussions regarding related work of the first (resp. second) part of the dissertation.

* The authors contributed equally to this research.

(1) Formerly JMLR WCP proceedings.

1.2 Organization of this dissertation and the author's contributions

The dissertation starts with an introduction in the first chapter, followed by an outline of its main contributions. Chapter 9 draws a conclusion and discusses future directions. The remaining chapters are organized into two parts, each reflecting one of the two main scenarios of recommendation systems: (1) context-aware (chapters 2 to 5) and (2) context-free (chapters 6 to 8). The closing chapter of each part discusses the related work.

1.2.1 Part One: Context-aware recommendation

Methods, optimization and algorithms: In Chapter 2, I introduce OMIC, a matrix completion method. To recover the missing entries, one must make an assumption about the structure of the ground truth matrix, and the most frequent assumption is that it is of low rank. However, optimally approximating the observed entries while minimizing the rank is NP-hard [28]. The SoftImpute algorithm [29] bypasses this difficulty by using the nuclear norm as a regularizer. Not only does SoftImpute work well in practice, but it also enjoys favorable theoretical guarantees: it requires only a small number of known entries to recover the underlying low-rank matrix exactly [30, 31] or approximately [32, 33] from noisy entries. Aligned to SoftImpute, OMIC also imposes nuclear-norm regularization as an effective convex relaxation of the rank constraint.

In practical matrix-completion applications, the following refinements have proven useful for improving the performance of traditional matrix completion methods: (1) *incorporating biases*; and (2) *Incorporating side information.* Previous work incorporated user and item biases in a pre-processing step and then trained matrix completion on the residuals [12, 13]. Inductive Matrix Completion (IMC) [14] uses side information to guide the prediction of the user-item ratings. IMC, which is backed up by well-developed learning theory [14, 34–37], can be applied also to new users with no ratings, but for which side information is given.

The aim here is to create a comprehensive generic model that can incorporate all the improvements mentioned above into a single flexible framework with a well-principled optimization procedure. The literature review in the first part of the thesis focuses on matrix completion methods with a theoretical foundation. A thorough search of the relevant literature yielded only one work that attempted to incorporate some of the aforementioned improvements into a single jointly trained model [38]. They consider outputs of the form

$$f_{i,j} = \mathbf{x}_i^\top M \mathbf{y}_j + z_{i,j}, \tag{1.1}$$

with nuclear-norm regularization imposed on both Z and M. The model is trained with gradient descent. The incorporation of both the standard low-rank term Z and the IMC term $XMY^\top$ allows the model to capture both generic low-rank phenomena, as well as any behavior related to the side information. The hyperparameters involved in the nuclear-norm constraints can be optimized through cross-validation and allow the model to decide how relevant the side information is. However, a single given matrix $f_{\cdot,\cdot}$ can correspond to several possible choices of M and Z, thereby limiting the interpretability of the model and the individual terms of the sum (1.1). Furthermore, the model does not capture user and item biases. OMIC remedies these failings. Firstly, the corresponding predictors can take the following form as a particular case:

$$f_{i,j} = c + u_i + m_j + \mathbf{x}_i^\top M \mathbf{y}_j + z_{i,j}, \tag{1.2}$$

where c is an unknown constant that corresponds to a global bias of the model, u_i and m_j are the biases of the i-th user and j-th movie, $\mathbf{x}_i$ and $\mathbf{y}_j$ are the *known* side information vectors of the i-th user and j-th movie, whilst M and $Z = (z_{i,j})$ are parameter matrices to which nuclear norm regularization is applied. The nuclear norm of a matrix is defined as the sum of its singular values. Thus, our regularizer encourages M (and consequently the side information term $\mathbf{x}_i^\top M \mathbf{y}_j$) and Z to have low-rank. In summary, OMIC is the first matrix completion method able to model biases, side information terms, as well as residual generic low-rank effects in a single, jointly trained model.

Furthermore, OMIC imposes orthogonality constraints for the construction of the inductive matrices, which effectively require each term in the sum in (1.2) to live in separate, mutually orthogonal subspaces. Such constraints have three advantages. First, training can be performed for all components simultaneously. Second, the variables in (1.2) admit interpretation. This is because any ground truth matrix can be represented *uniquely* (thanks to the orthogonality conditions). The magnitude of the terms of the sum can be interpreted as their relevance to the model. And third, it provides the ideal framework for modeling IMC using cluster side information. Observe that clustering information is intrinsically orthogonal: m users distributed in d disjointed clusters can be represented by an orthogonal matrix $X \in \mathbb{R}^{m\times d}$. Each row i of X is a one-hot encoding vector with the single high in the position corresponding to the cluster that user i belongs to.

The second chapter also covers a proposal of an efficient optimization algorithm for solving the OMIC's optimization problem, as well as it shows upper bounds on its convergence rate (proofs available in the appendix). Further, I proposed a *cluster detection* method: as in a typical use of OMIC, the method assumes that the rows (users) and columns (items) of the rating matrix can be split into groups (communities), with the user and item communities contributing to the rating in an additive way. However, the community memberships are *unknown* and they must be learned by the method. Previous attempts at detecting user and item clusters based purely on a low-rank partially observed matrix assume noisily observed pure community behaviour [9, 10]. On the other hand, by performing community discovery and low-rank matrix completion *jointly*, my method showed that community behavior and continuous low-rank structure can *coexist* in the same matrix.

Theoretical analysis: In Chapter 3 I theoretically analyse OMIC. First, I prove the uniqueness of decomposition of OMIC's predictor. Such property allows our method to give rise to interpretable solutions. Then, I proved learning-theoretical guarantees in the form of generalization bounds. These apply to the case of a sampling distribution in the distribution-free case (no assumption on the ground truth distribution). The better the model matches the ground truth, the tighter the bounds.

I also conducted a theoretical analysis of community side information when the sampling distribution is known. For that, I consider the case of adjusted nuclear norm regulariser: instead of the nuclear norm of matrix M (with the predictor $XMY^{\top}$), the method regularizes the nuclear norm of $\widetilde{M} = [D^{\frac{1}{2}}ME^{\frac{1}{2}}]$. Here, the diagonal matrix D has the entry $D_{u,u} = \sum_{j\leqslant n; f(i)=u} p_{i,j}$, where $f(i)$ denotes the cluster to which user-i belongs and $p_{i,j}$ is the probability of sampling entry i, j. In other words, $D_{u,u}$ is the marginal probability of hitting any entry whose user component belongs to cluster u. E can be defined analogously.

Experiments and applications: Chapter 4 presents experiments and applications of OMIC in recommender systems and natural sciences. By generating synthetic data, I first experimentally demonstrate that the proposed model is capable of retrieving biases and cluster information if such a structure is present in the ground-truth matrix. For the RS application case, I demonstrate that OMIC outperforms the SOTA in terms of accuracy, with the added benefit of interpretability, on a large set of real-world data.

Lastly, I examined the application of OMIC in the natural sciences for the prediction of activity coefficients in thermodynamics. In chemical engineering, activity coefficients are a physicochemical property of binary liquid mixtures and they have relevance to modeling chemical and phase equilibria as well as transport processes. I was the first to propose a matrix completion method to predict activity coefficients through a probabilistic matrix factorization model. In this case, each row corresponds to a solute while each column corresponds to a solvent. The entry $\{i, j\}$ is the activity coefficient of a mixture of solute i and solvent j. My model outperformed the modified-UNIFAC [39], the SOTA method refined over three decades. Although the probabilistic method requires much less training effort than the traditional baseline, it is still an expensive model in terms of algorithm complexity. This fact makes it difficult to validate the model and to select the hyperparameters properly. Furthermore, the scalability of large matrices is unfeasible. By using OMIC with cluster side information (e.g., chemical family of the solute/solvent) to solve this matrix completion problem, I achieved SOTA results with a (much) more efficient algorithm and interpretable model.

Related work: Chapter 5 contains related work of the first part of the dissertation.

1.2.2 Part Two: Context-free and cold-start methods

Methods, optimization and algorithms: In Chapter 6 begins the second part of the dissertation. I shift our focus to address the particularly challenging recommendation scenario, where there is not only a lack of information about user preferences and behavior, but of *any* usable side information (context-free). Providing effective recommendations here requires identifying the 'trending' items most popular among the audience. A popular option is to model the problem as a multi-armed bandit [17]. The classic MAB approach, on the other hand, is unrealistic because it assumes an environment with stationary reward distribution. In fact, the popularity of the items may fluctuate over time [16, 40, 41].

I therefore model this challenge recommendation scenario as a *non-stationary* MAB problem. Interestingly, the recommender is continuously faced with the classic exploration/exploitation dilemma known from reinforcement learning: the RS must maintain a balance between recommending a classic popular item and recommending the object of the current viral fad. To illustrate this dilemma, consider the following example. Suppose that a RS must select among videos of two artists: the South Korean singer Psy and the British singer David Bowie. The gray lines in both graphs of Figure 1.1 show the cumulative[1] level of system activity (only USA audience) associated with both artists. Most of the time, the rate of growth of the system activity is approximately constant. However, this linearity is sometimes broken by sudden bursts of events highlighted by the two vertical lines. The first spike (vertical red line) matches with the "*Gangnam Style*" release. The hit had an unprecedented explosion of popularity[2] and its music video "broke" the YouTube view counter's limit. The second burst of events (vertical blue line) coincides with David Bowie's unexpected death. This unfortunate exogenous event triggered the audience's curiosity.

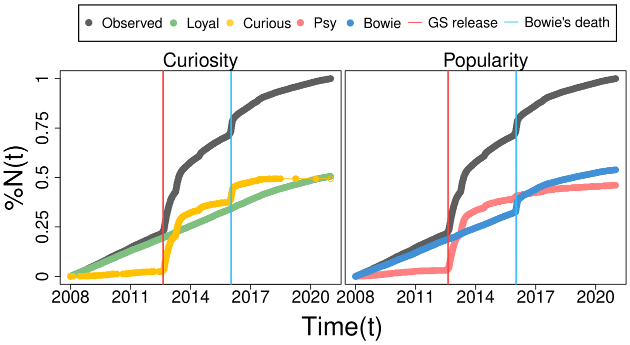


Figure 1.1: Two examples of time series factorization that motivate our method. **Left:** loyal and curious systems' audiences. **Right:** activity related to Psy and David Bowie. The data is taken from Google Trend (Jan/2008 to Dec/2020; country: USA; search engine: Youtube). We give a detailed description of the results in the main text.

---

[1] For a time series $T = \{t_1, t_2, \cdots, t_n\}$, we denote the numbers of events that happened before t by $N(t) := \sum_i^n 1_{\{t_i < t\}}$.

[2] The instant popularity $p(t)$ can be expressed as $p(t) = \partial \mathbb{E}(N(t))/\partial t$

The existence of two types of audiences (disentangled in Figure 1.1, left graph) explains the variations in the users' activity: the *loyal* audience and the *curious* audience. The loyal audience (green curve) is constituted by fans who assiduously follow the topic. In contrast, the curious audience (yellow curve) only turned their attention to the topic due to an extraordinary event. Thus, the environmental context in which the RS must make decisions alternates between calm periods (where the loyal audience drives the users' behavior), and disruptive or 'bursty' periods (where the curious audience is driving sudden bursts of interest in certain topics). In the real-life example presented here, during stable periods, the ratio between the singers' popularity is stable and Bowie is consistently more popular than Psy (Figure 1.1, right graph). On the other hand, during the disruptive period dominated by *curious* behavior, the relative popularity between Psy and Bowie changed drastically.

Motivated by this phenomenon, entitled *audience curiosity* [26, 42, 43], I developed the BMAB, which is a novel context-free, non-stationary and cluster-induced MAB algorithm. The algorithm's core consists of two consecutive stages: (1) categorizing the user-activity based on the system's state; and (2) inducing the exploration and exploitation procedures based on user-activity categories. BMAB is the main contribution of the second part of the dissertation.

Prior work on non-stationary MAB problems [24, 44] has dealt with the shifts in the items' popularity in both context-free [21–25] and context-aware [16, 45, 46] situations. While the first group solely uses the observed rewards, the latter group requires user or item features to build its arm-selection strategy. This dissertation focus on context-free MABs. Context-free MAB's algorithms are divided into sliding-window methods [21, 23–25, 47, 48], discounted factor methods [22, 25], and mixed approaches [49]. However, previous work did not take into account the effect of audience curiosity in the reward distribution. They also treat the problem as a discrete-time one, rather than actively exploiting the continuous temporal activity of RSs. Comprehensively, BMAB incorporates all the cited refinements.

To implement these improvements, firstly, BMAB clusters the user-activity according to the *system*'s state by carefully modeling the audience's temporal dynamics. Depending on which type of audience currently dominates the system requisitions, the method can attribute two possible user-activity categories: loyal (stable) and curious (unstable). In a continuous-time setting, I modeled the events as a combination of two Poisson processes, one for each conceivable audience. By assuming that the *loyal* audience has a constant incoming rate λ_L, I proposed a state detector based on a hypothesis test: it checks if a homogeneous Poisson process, which has intensity function $\lambda(t) = \lambda_L$, generates the last Δ events. If the hypothesis is rejected, a burst is detected and the user-activity is classified as *curious*. Otherwise, it is classified as *loyal*.

In the second stage, by taking into account the user-activity category, BMAB induces the slot-machine procedures that are responsible for exploring the environment and exploiting prior knowledge of rewards distribution. As a result, I proposed a method to detect such dramatic changes in the environment: it intensifies its *exploratory behavior* during the turbulent period to keep up with changes in the optimal strategy. In the example above, Psy's sudden burst of popularity after the release of 'Gangnam Style' deeply altered the environment and reward distribution: Psy momentarily became more popular than David Bowie.

A crucial parameter of BMAB is the incoming rate λ_L of the *loyal* audience. As a contribution of this dissertation, I developed a model which is capable of disentangling the two sorts of audiences illustrated in Figure 1.1 (left graph). The model does not rely on difficult-to-obtain external data but just on Random Series of Events (RSE). RSEs are time series in which the timestamps correspond to interactions between users and a specific item.

Stochastic point processes form a statistical framework to learn and infer about RSEs [50, 51]. In theory, they could be used to the problem of estimating the *loyal audience* of online items, but existing models are not appropriate for this particular setting. While Poisson process [52, 53] can easily estimate the *loyal audience* when all incoming events arrive at a fixed and predictable rate, they fail to mimic the bursts of events seen in real data. On the other hand, self-exciting processes, such as Hawkes and Wold processes, are able to capture the correlations between consecutive events that generate bursts of activity [42, 54, 55]. In line with my previous work [26], we model the audience dynamic as a mixture of two stochastic point processes. The first component is a Self-feeding process (SFP) [55] and the second is a Homogeneous Poisson process (HPP). The SFP generates a bursty behavior, corresponding to viral threads caused by sudden external events. In contrast, the HPP models normal background behavior that is influenced only by the overall popularity of the topic (the *loyal audience*). Therefore, the model is able to flexibly incorporate dependencies between the two hidden and underlying point processes involving the *loyal* and the *curious* audience.

The methodology here, however, is entirely novel, more efficient and more effective. In order to disentangle the above-mentioned mixture of point Processes, I developed a new Expectation–maximization (EM) methodology. To optimize the likelihood, I used the EM algorithm, relying on Gibbs sampling in the E-step. Note that the EM algorithm in the case of point processes requires great care since the events are not independent and the usual derivations are not appropriate. In [26], a complicated EM strategy was derived based on a series of approximations. Here, I introduce a different approach that requires (far) fewer approximations.

Theoretically-wise, I proved regret guarantees for the BMAB algorithm when the states are recoverable and bursts are separable. Then, I conducted an experimental analysis of the theorized regret bounds using a large spectrum of the synthetic experimental setup.

Experiments and applications: In Chapter 7, I conducted extensive experiments to evaluate BMAB in a context-free recommendation scenario. First, I proposed a generation technique that establishes a link between user activity and item popularity. Then I generated several time series in order to analyze BMAB in different ground-truth regimes. My method was also fitted in four real-world datasets. The comparison considered six SOTA baselines and it achieved competitive results in both synthetic and real-world strands.

Furthermore, I used the audience disentangling model to monitor the changes in the faithful audience behavior. Then, I illustrate the advantages of the disentangling method through the analysis of real-world data: I describe the *absolute loyalty* and the *relative loyalty*, indicators drawn from my model to characterize the absolute and relative influence of the loyal audience on the observed events. At last, I performed extensive synthetic analysis to show that my methodology is able to recover the ground truth model and experimentally demonstrate that it fits a large number of real datasets with better results than alternative models.

Related work: Chapter 8 contains related work of the second part of the dissertation.

Part One:
Context-aware recommendation

Chapter 2

METHODS, OPTIMIZATION AND ALGORITHMS

Matrix completion, the problem of recovering the missing entries of a partially observed matrix, has found application in a wide range of domains. As examples, consider the following. (1) A streaming provider recommends movies to its users based on an incomplete database of user-movie ratings. (2) A social network wants to find missing links in their friendship network. (3) A chemical producer wants to predict interactions of chemical compounds from a subset of known pairwise interactions. These examples—from the domains of recommender systems [56], social network analysis [57], and chemical engineering [58]—highlight the wide range of applications of matrix competition. For simplicity, we frequently use movie recommendation as a running example, so the data consists of user-movie ratings. It should be clear that, more generally, we can work with type1-type2 pairs, depending on the application, e.g., user-book, user-user, compound-compound, etc.

This chapter introduces cluster-induced-based matrix completion methods. Our approach is context-aware due to the inclusion of side information. In contrast to previous matrix completion methods, our strategy can comprehensively incorporate bias, side information, and pure low-rank terms into a single framework with a well-principled optimization procedure. To accomplish this, we carefully design *inductive matrices* that enable our method to capture the aforementioned refinements.

Based on these premises, our primary task is to recommend non-rated items to users. We, therefore, propose a method (see Section 2.1.3) able to accurately recover the missing entries from the rating matrix and then predict which items are most convenient for each user. Notably, when side information is present, inductive matrix completion methods demand the observation of only a tiny proportion [14, 34, 35, 37] of ratings from a large set of users and items.

However, user and item communities (clusters) are often not explicitly available. In this case, our parallel task is to recover hidden communities that can assist the recommendation task. Previous attempts at detecting user and item clusters based purely on a low-rank partially observed matrix assume noisily observed pure community behaviour [9, 59]. On the other hand, we model community behaviour and continuous low-rank structure *coexisting* in the same matrix. By constructing a model which efficiently exploits the "discreteness prior" on the existence of underlying user and item communities that play a role in the generation of the ratings, we propose to perform community discovery and low-rank matrix completion *jointly*.

The **main** contributions in this chapter are the following:

- we propose a framework of inductive matrix completion learning methods, which imposes orthogonal constraints on the columns of the inductive matrices. Then, we construct an instance of our framework that comprehensively jointly models biases, cluster side information, and a pure low-rank term;
- we propose an efficient optimization algorithm for solving our matrix completion problem. Furthermore, we show its convergence and give upper bounds on its convergence rate (proofs in the appendix);
- we also provide a scalable implementation of our algorithm that allows us to work on large datasets;
- we also provide a method to recover hidden clusters based on the hypothesis that community behavior and continuous low-rank structure can *coexist* in the same matrix.

Parts of this chapter are based on:

> Antoine Ledent*, **Rodrigo Alves***, and Marius Kloft. Orthogonal inductive matrix completion. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
>
> **Rodrigo Alves***, Antoine Ledent*, Renato Assunção, and Marius Kloft. An empirical study of the discreteness prior in low-rank matrix completion. In *Proceedings of Machine Learning Research*(1), volume 148: Pre-registration in ML, pages 111–125, 2021.

* The authors contributed equally to this research.
(1) Formerly JMLR WCP proceedings.

2.1 Description of the model and optimization procedure

Basic notation: We assume that we have an $m \times n$ matrix R whose entries are partially revealed to us. The set of revealed entries is denoted by $\Omega \subset \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\}$, and $\Omega^\perp$ denotes the complement $\{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \backslash \Omega$ (i.e., the set of unobserved entries). The projection on the set of matrices with all entries on $\Omega^\perp$ being zero is denoted by P_Ω. $P_{\Omega^\perp}$ is defined similarly. We denote the matrix of observed entries by R_Ω.

The general form of the optimization problem we consider is as follows:

$$\min_M \quad \mathcal{L}(R_\Omega, M, \Lambda) \qquad \text{with} \tag{2.1}$$

$$\mathcal{L}(R_\Omega, M, \Lambda) = \sum_{k=1}^{K} \sum_{l=1}^{L} \lambda_{k,l} \|M^{(k,l)}\|_* + \frac{1}{2} \sum_{(i,j)\in\Omega} \ell \left[R_{i,j}, \left(\sum_{k=1,l=1}^{K,L} X^{(k)} M^{(k,l)} (Y^{(l)})^\top \right)_{i,j} \right].$$

Here K (resp L) is the number of auxiliary matrices related to users (resp. items) and $\lambda_{k,l}$ a regularization parameter. Therefore, our model requires choosing sets of inductive matrices $X^{(k)} \in \mathbb{R}^{m \times d_k^{(1)}}$, $Y^{(l)} \in \mathbb{R}^{n \times d_l^{(2)}}$ representing prior knowledge about the problem. The inductive matrices define the specific model choice with respect to the incorporation (or lack thereof) of biases, side information etc. We now observe that the terms of equation (1.2) can all be written in the form $X^{(k)} M^{(k,l)} (Y^{(l)})^\top$ for some suitably defined $X^{(k)}, Y^{(l)}$. For instance, if we set X as the identity matrix and Y as a column matrix of all 1's, then the matrix $XMY^\top$ has the user biases as entries: $\mathbf{x}_i M \mathbf{y}_j^\top = u_i$ for all i, j. If we set both X and Y as identity matrices, we obtain the specific user-movie match $\mathbf{x}_i M \mathbf{y}_j^\top = z_{i,j}$ for all i, j.

We note that several specific variations of our model are possible depending on whether or not side information is present, how many distinct types of side information are present, whether or not we want to include user/item biases, etc. Therefore, we rely on a *unified formal framework* to describe all possible variations of these ideas with the general model's predictors have the following form:

$$F = (f_{i,j}) = \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} M^{(k,l)} (Y^{(l)})^\top. \tag{2.2}$$

In Figure 2.1, we illustrate an comprehensive example with $K = L = 3$ which takes into account user and item biases, as well as side information in the form of partitions of users and items into communities (with the movie communities being genres). This representative example is described in more technical detail in Section 2.1.3, where we also further explain how equation (1.2) can be fully incorporated as a particular case of (2.2). To ensure the uniqueness of the decomposition, we require that the *columns* of the sets of

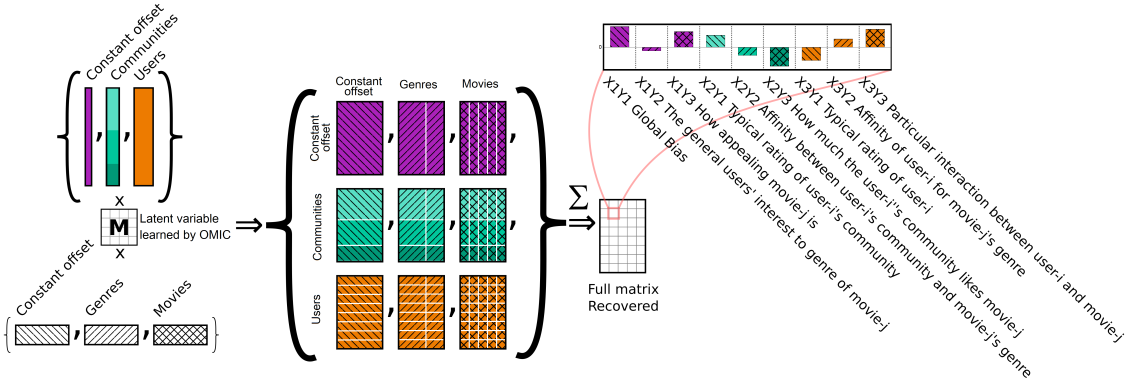

Constant offset
Communities
Users
x
M
Latent variable learned by OMIC
x
Constant offset
Genres
Movies
Constant offset
Genres
Movies
Constant offset
Communities
Users
Σ
Full matrix Recovered
0
X1Y1 Global Bias
X1Y2 The general users' interest to genre of movie-j
X1Y3 How appealing movie-j is
X2Y1 Typical rating of user-i's community
X2Y2 Affinity between user-i's community and movie-j's genre
X2Y3 How much the user-i's community likes movie-j
X3Y1 Typical rating of user-i
X3Y2 Affinity of user-i for movie-j's genre
X3Y3 Particular interaction between user-i and movie-j

Figure 2.1: Visualization of orthogonal inductive matrix completion (specifically, the comprehensive version explained in Section 2.1.3). The model is a sum of matrix terms, each of the form $XMY^\top$. It means each combination of X and Y gives rise to a term in the sum. We interpret the magnitude of this term as its relevance to the prediction.

inductive matrices form orthonormal bases of their respective spaces, i.e.

$$\begin{aligned} &\sum_{i=1}^{m} X^{k_1}_{i,j_1} X^{k_2}_{i,j_2} = \delta_{k_1,k_2}\delta_{j_1,j_2}; \qquad && \sum_{i=1}^{n} Y^{l_1}_{i,j_1} Y^{l_2}_{i,j_2} = \delta_{l_1,l_2}\delta_{j_1,j_2}; \\ &\mathrm{span}_{k,j}(X^{k}_{\bullet,j}) = \mathbb{R}^m \qquad \text{and} && \mathrm{span}_{l,j}(Y^{l}_{\bullet,j}) = \mathbb{R}^n. \end{aligned} \tag{2.3}$$

Thus we call our method by Orthogonal Inductive Matrix Completion (OMIC). Observe that the orthogonality assumption is **not** an assumption on the ground truth matrix, but a restriction of model choice. Orthogonal constraints benefit matrix completion methods in a variety of aspects, including providing an ideal environment for modeling biases and cluster side information, efficient optimization and interpretability enhancement.

2.1.1 Modelling jointly trained user/item biases

One notable example of this setting provides a way to train a low-rank matrix completion model together with user biases [12, 13]. For that, we generate the inductive matrices in the following way:

STEP 1: Set $X^{(1)} = (\frac{1}{\sqrt{m}}, \dots, \frac{1}{\sqrt{m}})^\top$ and $Y^{(1)} = (\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \dots, \frac{1}{\sqrt{n}})^\top$

STEP 2: Set $X^{(2)}$ (resp. $Y^{(2)}$) to be the orthogonal complement of $X^{(1)}$ (resp. $Y^{(1)}$) in $\mathbb{R}^m$ (resp. $\mathbb{R}^n$)

Then our model (2.1) is equivalent to the prediction function

$$f_{i,j} = c + u_i + m_j + Z_{i,j}, \tag{2.4}$$

where Z is constrained to have columns and rows summing to zero.

Here, the terms $X^{(1)}M^{(1,1)}(Y^{(1)})^\top$ and $X^{(1)}M^{(1,2)}(Y^{(2)})^\top + X^{(2)}M^{(2,1)}(Y^{(1)})^\top$, correspond to a general, matrix-wise bias, and a combination of user/item specific biases respectively. Meanwhile, the term $X^{(2)}M^{(2,2)}(Y^{(2)})^\top$ represents purely bias-free low-rank effects: an entry of $X^{(2)}M^{(2,2)}(Y^{(2)})^\top$ will be large if the item and user are particularly well-fitted to each other, independently of the general behavior of either user or item. This can be especially interesting in terms of interpretability, or if each user must be paired with a single item.

2.1.2 Modelling jointly trained cluster side information

A particularly representative case is when we are given side information about users and items in the form of clusters, with each user (resp. item) belonging to exactly one user (resp. item) cluster [38]. In this situation, we construct the columns of $X^{(1)}$ (resp. $Y^{(1)}$) as normalized indicator functions of the user (resp. item) communities, as following explained for $X^{(1)}$:

STEP 1: First, consider an orthogonal matrix $X \in \mathbb{R}^{m \times d}$, where m users will be split into d disjointed clusters. For that, make each row i of X a one-hot encoding vector with the single high in the position that corresponds to the cluster that user i belongs to.

STEP 2: For all $j \in \{1, 2, \cdots, n\}$, normalize $||X_{\cdot,j}|| = 1$

STEP 3: Finally, make $X^{(1)} = X$.

STEP 4: The columns of $X^{(2)}$ can then be chosen as an orthonormal basis of the orthogonal complement of $X^{(1)}$ in $\mathbb{R}^m$.

Analogously, we define Y and $Y^{(1)}$. Note that we have $Y^{(2)}$ as any orthonormal basis of the orthogonal complement of $Y^{(1)}$ in $\mathbb{R}^n$.

In this case, our model (2.1) is equivalent to optimizing a prediction function of the form

$$f_{i,j} = C_{f(i),g(j)} + M_{i,g(j)} + U_{f(i),j} + Z_{i,j}, \tag{2.5}$$

Here, the function $f : \{1, 2, \ldots, m\} \to \{1, 2, \ldots, d_1^1\}$ (resp. $g : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, d_2^1\}$) assigns to each user (resp. item) its cluster. In terms of interpretability, note that in the predictors $C_{f(i),g(j)} + M_{i,g(j)} + U_{f(i),j} + Z_{i,j}$ above, the contribution from C corresponds to effects that only depend on the cluster of the user and the cluster of the item, the contribution from M (resp. U) corresponds to 'specific user-item cluster' (resp. 'specific item-user cluster') effects, whilst the contribution from Z corresponds to effects that are purely specific to the user and the item (independently of their respective clusters).

2.1.3 A comprehensive jointly trained user/item biases and cluster side information method

We note that several variations of the ideas for the construction of the auxiliary matrices $X^{(k)}$ and $Y^{(l)}$ as above are useful in practice. Now we will present a more comprehensive instance, which combines the ideas above by incorporating *both* user and item biases *and* cluster side information. This is the specific model explained in Figure 2.1, and it is further empirically investigated in the sections of the experiments (See Section 4.4). Given the presence of cluster side information for users and items, we define our inductive matrices $X^{(k)}$ and $Y^{(l)}$ as follows:

STEP 1: $X^{(1)}$ and $Y^{(1)}$ are constructed as in Section 2.1.1, i.e. $X^{(1)} = (\frac{1}{\sqrt{m}}, \ldots, \frac{1}{\sqrt{m}})^\top$, $Y^{(1)} = (\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}})^\top$.

STEP 2: As defined in Section 2.1.2, let X (resp. Y) denote a normalized matrix whose columns are indicator functions of the user (resp. item) communities. The columns of $X^{(2)}$ (resp. $Y^{(2)}$) form an orthonormal basis of the space $\{v \in \mathrm{span}(X) : \langle v, X^{(1)} \rangle = 0\}$ (resp. $\{v \in \mathrm{span}(Y) : \langle v, Y^{(1)} \rangle = 0\}$), where $\mathrm{span}(X)$ (resp. $\mathrm{span}(Y)$) denotes the span of the columns of X (resp. Y).

STEP 3: Finally, the columns of $X^{(3)}$ (resp. $Y^{(3)}$) form an orthonormal basis of the orthogonal complement of the columns of $(X^{(1)}, X^{(2)})$ (resp. $(Y^{(1)}, Y^{(2)})$) in $\mathbb{R}^m$ (resp. $\mathbb{R}^n$).

Therefore, this set-up corresponds to equation (1.2), together with some orthogonality constraints. Indeed, $X^{(1)}M^{(1,1)}(Y^{(1)})^\top$ is a constant and corresponds to c. Furthermore, all the rows of $X^{(1)}M^{(1,3)}(Y^{(3)})^\top$ (resp. columns of $X^{(3)}M^{(3,1)}(Y^{(1)})^\top$) are equal, so that the term $X^{(1)}M^{(1,3)}(Y^{(3)})^\top$ (resp. $X^{(3)}M^{(3,1)}(Y^{(1)})^\top$) corresponds to $\mathbf{u}$ (resp. $\mathbf{m}$). $X^{(2)}M^{(2,2)}(Y^{(2)})^\top$ is the side information term corresponding to $\mathbf{x}_i M \mathbf{y}_j^\top$. Meanwhile, the remaining terms $X^{(1)}M^{(1,2)}(Y^{(2)})^\top + X^{(2)}M^{(2,1)}(Y^{(1)})^\top + X^{(3)}M^{(3,2)}(Y^{(2)})^\top + X^{(2)}M^{(2,3)}(Y^{(3)})^\top + X^{(3)}M^{(3,3)}(Y^{(3)})^\top$ correspond to the term $Z_{i,j}$ from equation (1.2), further refined into specific components distinguishing effects involving (1) only the side information of the users but not that of the items (or vice versa), (2) interactions between user bias and item side information (or vice versa) or (3) no side information or biases whatsoever.

2.2 Recovering inductive clusters via matrix completion

In Section 2.1 we propose a class of methods that uses communities (clusters) to induce the learning procedure. However, user and item communities are often not explicitly available. Here, we assume the rows (users) and columns (items) of the matrix split into groups (communities) with the property that each entry of the matrix is *a sum of components* corresponding to community behaviour and a purely low-rank component corresponding to individual behaviour. We introduced such a decomposition in Section 2.1.2, *assuming complete knowledge of the communities* of users and items. In contrast, we formulate an optimization problem *over* all (completed matrix, set of communities) *pairs* based on a nuclear-norm regularizer which jointly encourages *both* low-rank solutions *and* the recovery of ‘relevant’ communities.

Based on (2.1), we propose the following optimization problem:

$$\min_{f,g}\ \min_{C,M,U,Z} \mathcal{L} \quad \text{with} \qquad \mathcal{L} = \sum_{(i,j)\in\Omega} |C_{f(i),g(j)} + M_{i,g(j)} + U_{f(i),j} + Z_{i,j} - R_{i,j}|^2 \\ + \lambda_C \|C\|_* + \lambda_{MU}\left[\|M\|_* + \|U\|_*\right] + \lambda_Z \|Z\|_*, \tag{2.6}$$

subject to

$$\begin{aligned} &\sum_{i\in f^{-1}(u)} M_{i,v} = 0 \quad \forall u \leqslant d_1, v \leqslant d_2, && \sum_{j\in g^{-1}(v)} U_{u,j} = 0 \quad \forall u \leqslant d_1, v \leqslant d_2, \\ &\sum_{i\in f^{-1}(u)} Z_{i,j} = 0 \quad \forall j \leqslant n, \quad \text{and} && \sum_{j\in g^{-1}(v)} Z_{i,j} = 0 \quad \forall i \leqslant m, \end{aligned} \tag{2.7}$$

where λ_C, λ_{MU} and λ_Z are regularization parameters. The conditions of (2.7) are the same orthogonal conditions of our framework OMIC. They imply that the matrix Z is free of any community-wide behaviour component (for either users and items) and the matrices

$M \in \mathbb{R}^{m \times d_2}$ and $U \in \mathbb{R}^{d_1 \times n}$ are free of any community-wide behaviour components for the users and items respectively.

Remark 2.2.1. *The optimization in* (2.6) *is over the matrices C, M, U and Z, and the choice of communities f, g.*

2.3 The OMIC algorithm

In this section, we propose an iterative imputation algorithm to solve the problem (2.1) with the square loss. The first step is to develop a method to solve (2.1) in the case where $\Omega = \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\}$, the so-called "fully known case". The final solution can then be obtained by iteratively using this method.

The fully known case: Recall the singular value thresholding operator S_λ from [29], which is defined by $S_\lambda(Z) = \sum_{i=1}^{r} (\lambda_i - \lambda)_+ v_i w_i^\top$, where $Z = \sum_{i=1}^{r} \lambda_i v_i w_i^\top$ is the singular value decomposition (SVD) of Z.

In our case, we now introduce the *Generalized singular value thresholding operator* S_Λ, which, for any set of parameters $\Lambda = (\lambda_{k,l})_{\substack{k \leqslant K \\ l \leqslant L}}$, and given a set of auxiliary matrices $X^{(k)}, Y^{(l)}$ (satisfying the orthogonality conditions (2.3)), is defined by

$$S_\Lambda(Z) = \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} S_{\lambda_{k,l}}(M^{(k,l)})(Y^{(l)})^\top, \tag{2.8}$$

where $M^{(k,l)} = (X^{(k)})^\top Z (Y^{(l)})$, which ensures $Z = \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} M^{(k,l)} (Y^{(l)})^\top$.

Proposition 2.3.1. *The definition in equation* (2.8) *is equivalent to the following:*

$$S_\Lambda(Z) = \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} S_{\lambda_{k,l}} \left((X^{(k)})^\top Z Y^{(l)} \right) (Y^{l})^\top. \tag{2.9}$$

Furthermore, $\tilde{Z} = S_\Lambda(Z)$ is the solution to the following optimization problem:

$$\min \frac{1}{2} \|\tilde{Z} - Z\|_{\mathrm{Fr}}^2 + \sum_{k=1}^{K} \sum_{l=1}^{L} \lambda_{k,l} \left\| (X^{(k)})^\top Z Y^{(l)} \right\|_*, \tag{2.10}$$

or equivalently

$$\min \frac{1}{2} \|\tilde{Z} - Z\|_{\mathrm{Fr}}^2 + \sum_{k=1}^{K} \sum_{l=1}^{L} \lambda_{k,l} \left\| M^{(k,l)} \right\|_*, \tag{2.11}$$

$$\textit{subject to} \quad \tilde{Z} = \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} M^{(k,l)} (Y^{(l)})^\top. \tag{2.12}$$

Proposition 2.3.1 is proved in Appendix A.

For any fixed set of hyperparameters Λ and auxiliary matrices $X^{(k)}, Y^{(l)}$ (for $k \leqslant K, l \leqslant L$), the final solution to the optimization problem (2.1) is obtained iteratively by the Algorithm 1: at each step, a target matrix is constructed by setting the entries of Ω to the observed values and imputing the values of the previous iteration to the entries of $\Omega^{\perp}$ (line 3). We then apply the fully known case (Proposition 2.3.1) to this target matrix to reach the next iterate (line 4). This algorithm converges for any initial (0th iteration) matrix, for example setting all the unobserved entries to 0 (line 1).

Algorithm 1 OMIC

INPUT: Partially observed matrix $R_\Omega \in \mathbb{R}^{m \times n}$, regularizing parameters $\Lambda \in \mathbb{R}^{K \times L}$ and start matrix $S \in \mathbb{R}^{m \times n}$

MODEL CHOICE: $X^{(1)}, \dots, X^{(K)}$ and $Y^{(1)}, \dots, Y^{(L)}$

OUTPUT: A recovered matrix $Z \in \mathbb{R}^{m \times n}$

```
1: Initialize $Z^{\text{new}} \leftarrow S$
2: while not converged do
3:     $Z^{\text{old}} \leftarrow R_\Omega + P_{\Omega^\perp}(Z^{\text{new}})$
4:     $Z^{\text{new}} \leftarrow S_\Lambda\left(Z^{\text{old}}\right)$
5: end while
6: return $Z = Z^{\text{new}}$
```

If we must calculate several values of Λ, we can use warm starts to improve efficiency. Algorithm 2 does this in the case where the range of values for Λ is a product $\mathcal{V} = \prod_{k,l} \mathcal{V}_{k,l}$ where the $\mathcal{V}_{k,l}$ are finite sets of candidate values for $\lambda_{k,l}$ (ordered in increasing or decreasing order): initial estimates of $M_{k,l}$ for each value of $\lambda_{k,l}$ are calculated by setting each $\lambda_{k',l'}$ to infinity for $k' \neq k, l' \neq l$ and solving the full problem (2.1) in this case. Furthermore, each of those sets of $M_{k,l}$ are calculated using warm starts along the sequence of $\lambda_{k,l} \in \mathcal{V}_{k,l}$. For any real number λ, $p_{k,l}(\lambda)$ denotes the set of hyperparameters Λ with $\Lambda_{k,l} = \lambda$ and $\Lambda_{k',l'} = \infty$ otherwise.

Remark 2.3.2. *"Setting $\lambda_{k',l'}$ to infinity" amounts excludes the (k', l') term in the sum (2.2) which defines our predictors. Thus our warm starts are computed by training each term in that sum independently.*

Algorithm 2 OMIC with warm-starts

INPUT: Partially observed matrix $R_\Omega \in \mathbb{R}^{m\times n}$ and a set of regularizing parameters $\mathcal{V} = \prod_{k,l} \mathcal{V}_{k,l}$

MODEL CHOICE: $X^{(1)}, \ldots, X^{(K)}$ and $Y^{(1)}, \ldots, Y^{(L)}$

OUTPUT: Set of recovered matrices Z^Λ for all $\Lambda \in \mathcal{V}$

```
for $k \in \{1, 2, \ldots, K\}$ do
    for $l \in \{1, 2, \ldots, L\}$ do
        for $\lambda \in \mathcal{V}_{k,l}$ do
            $\Lambda = p_{k,l}(\lambda)$, $S = 0$
            $Z^{k,l,\lambda} \leftarrow$OMIC$(R_\Omega, \Lambda, S)$
        end for
    end for
end for
for $\Lambda \in \mathcal{V}$ do
    $S \leftarrow \sum_{k,l=1}^{K,L} Z^{k,l,\lambda_{k,l}}$
    $Z^\Lambda \leftarrow$OMIC$(R_\Omega, \Lambda, S)$
end for
return  $Z^\Lambda$ for $\Lambda \in \mathcal{V}$
```

2.3.1 A scalable version for OMIC algorithm

The main computational step at each iteration of the Algorithm 1 is the calculation of the solution to an instance of the fully known case (line 4), which can be obtained via our *generalized singular value thresholding operator* S_Λ. Observe that the sum of the numbers of columns of all $X^{(k)}$ (resp. $Y^{(l)}$) is m (resp. n). Thus, if m and n are large (which is often the case in RSs contexts), it is infeasible to even store all the auxiliary matrices in memory, let alone perform operations on them directly. The same problem can occur with some of the latent matrices $M^{(k,l)}$. It is easy to see that the largest $M^{(k,l)}$ has at least $m/K \times n/L$ entries which is also very large.

In this section, we show how to circumvent this difficulty in the specific case of for the comprehensive case of OMIC (with biases and cluster side information, Section 2.1.3) where $K = L = 3$ and the auxiliary matrices are defined assuming the side information X, Y consists of indicator functions of clusters. For instance, the columns of Y could represent sets of movies belonging to a specific genre, whilst the columns of X could represent countries, genders or professions for users. Our strategy heavily relies on both the "*sparse-plus-low-rank*" structure present in the target matrices of the "fully known problem" solved at each iteration, as well as the specific structure of the community side information.

First, we define the matrices $\widetilde{X}^{(0)}, \cdots, \widetilde{X}^{(3)}$ and $\widetilde{Y}^{(0)}, \cdots, \widetilde{Y}^{(3)}$ as follows:

STEP 1: Let $\widetilde{X}^{(0)} = (0, \ldots, 0)^\top$ and $\widetilde{Y}^{(0)} = (0, \ldots, 0)^\top$.

STEP 2: $\widetilde{X}^{(1)} = X^{(1)}$ and $\widetilde{Y}^{(1)} = Y^{(1)}$ are constructed as in Section 2.1.1, i.e. $X^{(1)} = (\frac{1}{\sqrt{m}}, \ldots, \frac{1}{\sqrt{m}})^\top$, $Y^{(1)} = (\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \ldots, \frac{1}{\sqrt{n}})^\top$;

STEP 3: Similarly to what we did in Section 2.1.2, let X (resp. Y) denote a matrix whose columns are indicator functions of the user (resp. item) communities. Make $\widetilde{X}^{(2)} = \text{normalize}(X)$ and $\widetilde{Y}^{(2)} = \text{normalize}(Y)$. Here, normalize denotes the operation of normalizing each column.

STEP 4: Finally, $\widetilde{X}^{(3)} = \text{Id}$ and $\widetilde{Y}^{(3)} = \text{Id}$.

Remark 2.3.3. *Note that X (resp. Y) is a sparse matrix composed of the indicator functions of user (resp. item) communities. Therefore, the matrices $\widetilde{X}^{(1)}$, $\widetilde{X}^{(2)}$,$\widetilde{X}^{(3)}$,$\widetilde{Y}^{(1)}$,$\widetilde{Y}^{(2)}$and $\widetilde{Y}^{(3)}$ can easily be stored in memory, and it is easy to multiply them by arbitrary vectors on either side.*

Now observe that due to the rotational invariance of the Singular Value Decomposition (SVD), the operator S_Λ can be rewritten as

$$S_\Lambda(Z) = \sum_{k=1}^{3} \sum_{l=1}^{3} S_{\lambda_{k,l}} \left(X^{(k)} (X^{(k)})^\top Z Y^{(l)} (Y^{(l)})^\top \right). \tag{2.13}$$

Thus, for a given Z, calculating $S_\Lambda(Z)$ boils down to computing the SVD of the matrices $H^{(k,l)} = X^{(k)}(X^{(k)})^\top Z Y^{(l)} (Y^{(l)})^\top$, which are the projections of Z on the spaces corresponding to each pair of auxiliary matrices. We perform the SVD computation through a rank-restricted alternating least squares algorithm (see Algorithm 4). This requires an efficient strategy to compute $H^{(k,l)} W_1$ and $W_2 H^{(k,l)}$, where W_1 and W_2 are any real conformable matrices.

We now observe that for any conformable orthogonal matrix U, if $W_1 \in \mathbb{R}^{n \times r}$, $UU^\top W_1$ is the projection of W_1 on the span of the columns of U. Crucially, if V is an orthogonal matrix with $\text{span}(V) \subset \text{span}(U)$, then for any $W_1 \in \mathbb{R}^{n \times r}$, $UU^\top W_1 - VV^\top W_1$ is the projection of

W_1 on the orthogonal complement of V in U. Applying this to the matrices $Y^{(l)}$ and $\widetilde{Y}^{(l)}$, we obtain, for all $l \leqslant 3$:

$$Y^{(l)}(Y^{(l)})^\top W_1 = \widetilde{Y}^{(l)}(\widetilde{Y}^{(l)})^\top W_1 - \widetilde{Y}^{(l-1)}(\widetilde{Y}^{(l-1)})^\top W_1. \tag{2.14}$$

Similarly, for any $W_2 \in \mathbb{R}^{m \times r}$ and $k \leqslant 3$,

$$X^{(k)}(X^{(k)})^\top W_2 = \widetilde{X}^{(k)}(\widetilde{X}^{(k)})^\top W_2 - \widetilde{X}^{(k-1)}(\widetilde{X}^{(k-1)})^\top W_2. \tag{2.15}$$

With (2.14) and (2.15), we are now in a position to present our algorithm for calculating $H^{(k,l)}W_1$. We will divide the task into three steps.

STEP 1: First, we evaluate $\widetilde{W_1} = Y^{(l)}[(Y^{(l)})^\top W_1]$ using (2.14).

Observe that as a byproduct of the iterative imputation procedure, Z can be decomposed as the sum of a sparse matrix Z_{Sp} and a low-rank matrix Z_{LR} as follows:

$$Z = Z_{Sp} + Z_{LR} = Z_{Sp} + U_{LR}\left[D_{LR}{V_{LR}}^\top\right]. \tag{2.16}$$

STEP 2: Then, using this decomposition, it is easy to calculate the quantity $\widetilde{\widetilde{W_1}} = Z\widetilde{W_1}$ as follows:

$$\widetilde{\widetilde{W_1}} = Z\widetilde{W_1} = Z_{Sp}\widetilde{W_1} + U_{LR}D_{LR}({V_{LR}}^\top \widetilde{W_1}). \tag{2.17}$$

STEP 3: Finally, we calculate $H^{(k,l)}W_1 = \big(X^{(k)}[(X^{(k)})^\top \widetilde{\widetilde{W_1}}]\big)^\top$ using (2.15). Symmetrically and with the same arguments we can compute $W_2 H^{(k,l)}$.

Algorithm 4 (based on [60]) describes how to compute $S_\Lambda(Z)$ for a fixed hyperparameter set Λ and Algorithm 3 is our fully scalable implementation of Algorithm 1 for the case with biases and cluster side information. In practice, we can further use warm start strategies such as the one presented in Algorithm 2 to speed up convergence.

Remark 2.3.4. *To avoid manipulating or storing large matrices, it is necessary to perform the above operations in the correct order, which is defined by the brackets. This remark applies, in particular, to Algorithms 3 and 4.*

Remark 2.3.5. *The convergence of Algorithm 4 follows from that of Algorithm 2.1 in [60]: for each combination* (k, l)*, of which there are finitely many, the while loop from lines 10 to 21 essentially runs Algorithm 2.1 from [60] on the component matrix* $R^{(k,l)} = (X^{(k)})^\top R Y^{(l)}$*, thus the full algorithm converges. The convergence of Algorithm 3 then follows from (1) the convergence of Algorithm 4 (which is used inside the while loop) together with (2) the convergence of Algorithm 1, which is established in Theorem 2.3.1.*

Algorithm 3 Scalable OMIC with biases and cluster side information

INPUT: Partialy observed matrix $R_\Omega \in \mathbb{R}^{m \times n}$ (stored as sparse matrix), regularizing parameters $\Lambda \in \mathbb{R}^{3\times 3}$, maximum rank $r \geqslant 1$ and a singular value decomposition of a start matrix $S_{\text{SVD}}\{U_{LR}, D_{LR}, V_{LR}\}$

MODEL CHOICE: Make $\widetilde{X}^{(0)}, \cdots, \widetilde{X}^{(3)}$ from $X^{(1)}, \ldots, X^{(3)}$ and $\widetilde{Y}^{(0)}, \cdots, \widetilde{Y}^{(3)}$ from $Y^{(1)}, \ldots, Y^{(3)}$

OUTPUT: Singular value decomposition of the recovered matrix $Z \in \mathbb{R}^{m \times n}$

```
$Z_s \leftarrow R_\Omega$
$Z_{LR} \leftarrow S_{\text{SVD}}$
while Not converged do
    $Z_s \leftarrow R_\Omega - P_\Omega(Z_{LR})$
    $Z_{LR} \leftarrow (S_\lambda)\text{-ALS}(Z = \{Z_s, Z_{LR}\})$
end while
return $Z \leftarrow Z_{LR}$
```

Algorithm 4 (S_Λ)-ALS: Generalized restricted truncated singular value thresholding via alternating least squares

INPUT: Matrix $Z \in \mathbb{R}^{m\times n}$ decomposed as in (2.16); thresholding parameters $\Lambda \in \mathbb{R}^{3\times 3}$ and maximum rank $r \geqslant 1$

MODEL CHOICE: Make $\widetilde{X}^{(0)}, \cdots, \widetilde{X}^{(3)}$ from $X^{(1)}, \ldots, X^{(3)}$ and $\widetilde{Y}^{(0)}, \cdots, \widetilde{Y}^{(3)}$ from $Y^{(1)}, \ldots, Y^{(3)}$

OUTPUT: $S_\Lambda(Z)$ represented in storable low-rank format as $(\mathfrak{U}, \mathrm{diag}(\Sigma), \mathfrak{V})$ such that $S_\Lambda(Z) = \mathfrak{U}\,\mathrm{diag}(\Sigma)\mathfrak{V}^\top$

```
Procedure Projection($E^{(1)}$,$E^{(0)}$,$\Theta$)
return $E^{(1)}[(E^{(1)})^\top\Theta] - E^{(0)}[(E^{(0)})^\top\Theta]$
end Procedure
$\mathfrak{U} \leftarrow \Sigma \leftarrow \mathfrak{V} \leftarrow NULL$
for k in (1..3) do
  for l in (1..3) do
    $U \leftarrow$ random orthogonal $m \times r$ matrix
    $D \leftarrow \mathrm{Id}_{r\times r}$
    $A \leftarrow UD$
    while $AB^\top$ not converged do
      $\Theta \leftarrow UD(D^2 + \lambda_{k,l}I)^{-1}$
      $\widetilde{\Theta} \leftarrow \mathrm{Projection}(\widetilde{X}^{(k)}, \widetilde{X}^{(k-1)}, \Theta)$
      $\widetilde{\widetilde{\Theta}} = \widetilde{\Theta}^\top Z_{Sp} + [(\widetilde{\Theta}^\top U_{LR})D_{LR}]{V_{LR}}^\top$
      $\tilde{B} \leftarrow \mathrm{Projection}(\widetilde{Y}^{(l)}, \widetilde{Y}^{(l-1)}, \widetilde{\widetilde{\Theta}^\top})$
      Compute the SVD of $\tilde{B}D = \tilde{V}\tilde{D}^2R^\top$ and attribute $V \leftarrow \tilde{V}$, $D \leftarrow \tilde{D}$ and $B \leftarrow VD$
      $\Theta \leftarrow VD(D^2 + \lambda_{k,l}I)^{-1}$
      $\widetilde{\Theta} \leftarrow \mathrm{Projection}(\widetilde{Y}^{(l)}, \widetilde{Y}^{(l-1)}, \Theta)$
      $\widetilde{\widetilde{\Theta}} = Z_{Sp}\widetilde{\Theta} + U_{LR}[D_{LR}({V_{LR}}^\top\widetilde{\Theta})]$
      $\tilde{A} \leftarrow \mathrm{Projection}(\widetilde{X}^{(k)}, \widetilde{X}^{(k-1)}, \widetilde{\widetilde{\Theta}})$
      Compute the SVD of $\tilde{A}D = \tilde{U}\tilde{D}^2\tilde{R}^\top$ and attribute $U \leftarrow \tilde{U}$, $D \leftarrow \tilde{D}$ and $A \leftarrow UD$
    end while
    $\widetilde{\Theta} \leftarrow \mathrm{Projection}(\widetilde{Y}^{(l)}, \widetilde{Y}^{(l-1)}, V)$
    $\widetilde{\widetilde{\Theta}} = Z_{Sp}\widetilde{\Theta} + U_{LR}[D_{LR}({V_{LR}}^\top\widetilde{\Theta})]$
    $M \leftarrow \mathrm{Projection}(\widetilde{X}^{(k)}, \widetilde{X}^{(k-1)}, \widetilde{\widetilde{\Theta}})$
    Compute the SVD of $M$: $M = \bar{U}D_\sigma\bar{R}^\top$.
    $\mathfrak{U} \leftarrow \mathrm{CONCAT}(\mathfrak{U}, \bar{U})$
    $\Sigma \leftarrow \mathrm{CONCAT}(\Sigma, \big((\sigma_1 - \lambda_{k,l})_+, (\sigma_2 - \lambda_{k,l})_+, \ldots, (\sigma_r - \lambda_{k,l})_+\big))$
    $\mathfrak{V} \leftarrow \mathrm{CONCAT}(\mathfrak{V}, V\bar{R})$
  end for
end for
return $\mathfrak{U}; \mathrm{diag}(\Sigma); \mathfrak{V}$
```

2.3.2 Complexity and runtime analysis

Note that at each iteration of the Algorithms 1 and 3 theoretically require KL SVD operations. However, for the instances of OMIC presented in this dissertation, most of the SVD calculations are actually *trivial*. For instance, in the specific case of Section 2.1.1, matrices $M^{(k,l)} = (X^{(k)})^\top Z Y^{(l)}$ are vectors whenever $k = 1$ or $l = 1$, thus calculating the SVD simply amounts to normalizing a vector. In fact, for any combination (k, l) such that $d_1^{(k)} + d_2^{(l)}$ is small, it is easy to compute the small matrix $M^{(k,l)} \in \mathbb{R}^{d_1^{(k)} \times d_2^{(l)}}$ and perform its SVD through standard methods.

We performed experiments with synthetic data to compare the runtimes of SVD calculations in SoftImpute and our algorithm. For each datapoint, we randomly selected a matrix with the following parameters: rank $\in \{5, 6, \cdots, 10\}$, $m \in \{100, 101, \cdots, 1000\}$ and $d_1^{(1)} = d_2^{(1)} \in \{2, 3, \cdots, \lceil 0.1m \rceil\}$. More specifically, the users and items were each divided into $d_1^{(1)} = d_2^{(1)}$ communities and the matrices $X^{(1)}, X^{(2)}, X^{(3)}, Y^{(1)}, Y^{(2)}, Y^{(3)}$ were constructed according to the standard procedure described in Section 2.1.3. The matrices $M^{(k,l)}$ were chosen with *iid* Gaussian entries.

We then compared, on the one hand (left part of Figure 2.2) the following two operations:

1. Performing the SVD of the full matrix $R = \sum_{k,l} X^{(k)} M^{(k,l)} (Y^{(k)})^\top$,

2. Performing the SVDs of *all nine matrices* $M^{(k,l)}$ for $k, l \leqslant 3$;

and on the other hand (right part of the figure), the following two operations:

1. One *full iteration* of the Softimpute algorithm, including imputation and singular value thresholding operator.

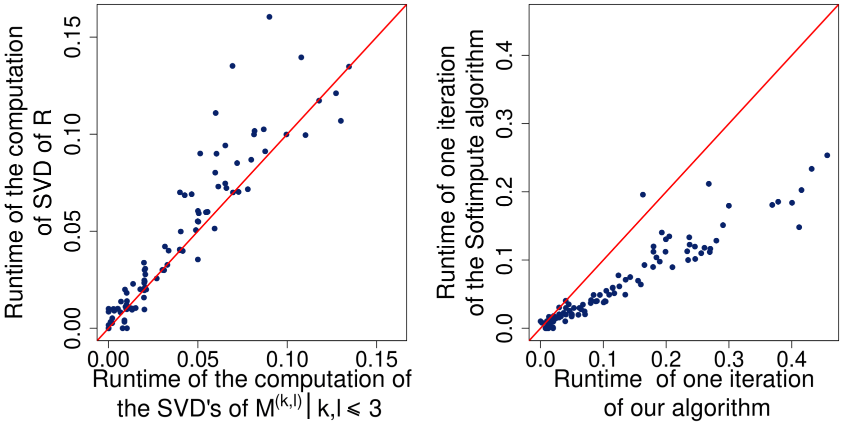


Figure 2.2: Runtime comparison of main computations required for one iteration of Softimpute and OMIC. The red line is the identity.

2. One *full iteration* of our algorithm, including the imputation and the application of the generalized singular value thresholding operator.

Remark 2.3.6. *As we can see from Figure 2.2, the computational burden of all nine SVDs required in our algorithm is not more significant than that of the single SVD required in the SoftImpute implementation. Furthermore, although one full iteration of our algorithm is slower than one full iteration of the SoftImpute algorithm due to extra multiplication steps, this appears to be the case only by a small constant factor.*

Formal complexity analysis: We will provide a formal complexity analysis of our efficient implementation proposed in Section 2.3.1. In this case, the number of iterations required at each step of both Algorithm 3 as well as the SVD calculation (Algorithm 4) depend on the many practicalities related to various warm starts applied in both cases. However, it is possible to write down the complexity of performing one iteration.

At each iteration of Algorithm 3, the key step is the singular value thresholding operation using Algorithm 4. For each iteration inside Algorithm 4, the complexity can be computed from the following operations, which are each required a fixed number of times (here, as usual, r is the fixed maximum rank set as a hyperparameter):

- Multiplying each column of a matrix in $\mathbb{R}^{m\times r}$ or $\mathbb{R}^{n\times r}$ by a different constant (e.g. lines 11 and 16). Cost: $O((m+n)r)$.
- Computing the SVD of a $\mathbb{R}^{m\times r}$ or $\mathbb{R}^{n\times r}$ matrix (e.g., lines 15, 20 and 25). Cost: $O(r^3+(m+n)r^2)$.
- Performing projections onto the spaces corresponding to $X^{(k)}$ or $Y^{(l)}$ via the procedure from line 1. Cost: $O(m+n)$.
- Multiplying r vectors by the current target (see lines 13, 18 and 23). Cost: $O(|\Omega|r + (m+n)r^2)$.

Since $r \leqslant m+n$, this yields an overall complexity of $O(KL[|\Omega|r+(m+n)r^2])$. Note that $KL = 9$, so that the complexity is $O(9[|\Omega|r+(m+n)r^2]) = O(|\Omega|r+(m+n)r^2)$, the same as the SoftImpute algorithm [29].

2.3.3 Convergence guarantees

The OMIC algorithm enjoys convergence guarantees, which we present here. Here, we fix a Λ and assume that we perform the iterative imputation procedure in the algorithm above starting from $Z^0 = 0$, with (for each $i \geqslant 0$)

$$Z^{i+1} = S_\Lambda \left(P_\Omega(R) + P_{\Omega^\perp}(Z^i) \right). \tag{2.18}$$

We have the following two results.

Theorem 2.3.1. *Consider our general setting with auxiliary matrices satisfying the conditions (2.3) and the operator S_Λ defined accordingly. The sequence Z^i defined in (2.18) converges to a solution Z^∞ of the optimization problem (2.1) with the squared loss. In particular, the loss $\mathcal{L}$ converges to the minimum $\mathcal{L}^*$ of optimization problem (2.1).*

Theorem 2.3.2. *Let $\mathcal{L}^*$ be the minimum value of the loss $\mathcal{L}$ from problem (2.1). For every fixed Λ, the sequence Z^i has the following worst-case asymptotic convergence:*

$$\mathcal{L}(Z^i) - \mathcal{L}(Z^\infty) = \mathcal{L}(Z^i) - \mathcal{L}^* \leqslant \frac{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2}{i+1}, \tag{2.19}$$

where Z^∞ is the limit of the sequence Z^i.

We prove Theorems 2.3.1 and 2.3.2 in the Appendix A.

Remark 2.3.7. *We use the notation Z^∞ to refer to the limit of the sequence of iterates, instead of referring to 'the solution Z^* of the optimization problem (2.1)' because the solution is not necessarily unique and Z^∞ may depend on the initialization. Indeed, the optimization problem is convex but not strongly convex. Thus, there can be several solutions, but each corresponds to the same value of the objective function.*

To check that the specific problem (2.1) can indeed have several equivalent solutions, consider the following example. $X^{(1)} = Y^{(1)} = \mathrm{Id}$ (so that our algorithm coincides with Softimpute), $m = n = 2$, and let the observed entries be $R_{2,1} = R_{1,2} = 1$ (thus, $\Omega = \{(2,1);(1,2)\}$.

Here are three equivalent solutions to the optimization problem with regularising parameter λ:

$$
\begin{aligned}
A_- &= \begin{pmatrix} \lambda - 1 & 1 - \lambda \\ 1 - \lambda & \lambda - 1 \end{pmatrix}; \\
A_+ &= \begin{pmatrix} 1 - \lambda & 1 - \lambda \\ 1 - \lambda & 1 - \lambda \end{pmatrix}; \qquad \text{and} \\
A_0 &= \begin{pmatrix} 0 & 1 - \lambda \\ 1 - \lambda & 0 \end{pmatrix}.
\end{aligned}
$$

In all three cases, the nuclear norm is $2 - 2\lambda$, and the value of the objective function is $\lambda(2 - 2\lambda) + \frac{1}{2}(\lambda^2 + \lambda^2)$. It is also easy to check that those are actually solutions of (2.1) because performing any extra iteration of the algorithm yields the same matrix: for A_0, after the imputation step we get the target

$$
\begin{aligned}
&\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \\
&= 1 \times \begin{pmatrix} 0 \\ 1 \end{pmatrix} \begin{pmatrix} 1 & 0 \end{pmatrix} + 1 \times \begin{pmatrix} 1 \\ 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \end{pmatrix},
\end{aligned}
$$

where the second line is the SVD. It is clear from the SVD that applying the singular value thresholding operator will return the matrix A_0. Thus, the algorithm converges exactly to A_0 in one (zero) iteration(s).

For A_+, note that after the imputation step, we get the target

$$
\begin{aligned}
&\begin{pmatrix} 1 - \lambda & 1 \\ 1 & 1 - \lambda \end{pmatrix} \\
&= (2 - \lambda) \times \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \\
&+ \lambda \times \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & -1/\sqrt{2} \end{pmatrix},
\end{aligned}
$$

and it is clear that after applying the SVT operator we obtain

$$
\begin{aligned}
&(2 - 2\lambda) \times \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix} \\
&= \begin{pmatrix} 1 - \lambda & 1 - \lambda \\ 1 - \lambda & 1 - \lambda \end{pmatrix} = A_+,
\end{aligned}
$$

as expected. A similar calculation shows that A_- is also a solution.

2.4 Collaborative Clustering Algorithm

Since (2.6) involves optimization over a combinatorial number of possible functions f, g we propose a heuristic algorithm to reach a solution. Like (2.6), our algorithm takes as input the partially observed matrix R and the hyperparameters d_1, d_2 and $\Lambda = \{\lambda_Z, \lambda_C, \lambda_{MU}\}$. Our strategy, further represented in Algorithm 5, is as follows.

STEP 1: First, we solve the optimization problem (2.6) for $f = g = \text{null}$ (which is equivalent to $d_1 = d_2 = 0$, see [29]).

STEP 2: Secondly, we cluster (using the k-means algorithm) both the rows and the columns of the recovered matrix, with the numbers of clusters set to d_1 and d_2, yielding the partitions f_0 and g_0 respectively.

STEP 3: By using OMIC algorithm we solve problem (2.6) with $f = f_0, g = g_0$ fixed, obtaining the matrices $\hat{R}_0 = \{C_0, M_0, U_0, Z_0\}$.

Our next aim is now to iteratively refine the partitions f and g.

STEP 4: To this end, for each set of non-negative parameters $\theta = (\theta_1, \theta_2, \theta_3, \theta_4)$ in some predetermined set Θ, the following cluster profile:

$$S^\theta = \theta_1 \tilde{C_0} + \theta_2 \tilde{M_0} + \theta_3 \tilde{U_0} + \theta_4 Z_0. \tag{2.20}$$

Here $\tilde{C}$, $\tilde{M}$ and $\tilde{U}$ are $m \times n$ matrices such that $\tilde{C}_{i,j} = C_{f(i),g(j)} \forall i \leqslant m, j \leqslant n$, $\tilde{M}_{i,j} = M_{i,g(j)} \forall i \leqslant m, j \leqslant n$, and $\tilde{U}_{i,j} = U_{f(i),j} \forall i \leqslant m, j \leqslant n$[3].

STEP 5: For each $\theta \in \Theta$ we now obtain partitions f_θ (resp. g_θ) of the users (resp. items) by clustering the rows (resp. columns) of S^θ.

---

[3] Note that since the matrices $\tilde{C}, \tilde{M}, \tilde{U}$ and Z live in mutually orthogonal subspaces with respect to the Frobenius inner product, the matrices C, M, U, Z (and in particular the loss $\mathcal{L}$) are well-defined for any full matrix $R = \tilde{C} + \tilde{M} + \tilde{U} + Z$ for any given set of hyperparameters and partitions f, g.

STEP 6: Next we solve (2.6) fixing $f = f_\theta, g = g_\theta$, obtaining the matrices $\hat{R}_\theta = \{C_\theta, M_\theta, U_\theta, Z_\theta\}$ and calculate $\ell_\theta = \mathcal{L}(R_\Omega, \hat{R}_\theta, \Lambda, f_\theta, g_\theta)$.

STEP 7: Then, we compute the minimum $\ell_{\theta_{\min}}$ of ℓ_θ over all values of θ and retain the partitions $f_{\theta_{\min}}, g_{\theta_{\min}}$ and the associated matrices $\hat{R}_{\theta_{\min}} = \{C_{\theta_{\min}}, M_{\theta_{\min}}, U_{\theta_{\min}}, Z_{\theta_{\min}}\}$.

STEP 8: Finally, if the algorithm is not converged, we can feed this data to the next iteration of the algorithm: we attribute $\hat{R}_0 = \hat{R}_{\theta_{\min}}$, $f_0 = f_{\theta_{\min}}, g_0 = g_{\theta_{\min}}$ and go back to STEP 3.

Regarding the choice of the searched set Θ, since we use the k-means algorithm as the clustering procedure, we can restrict ourselves to θs such that $\theta_1 + \theta_2 + \theta_3 + \theta_4 = 1$, and for computational reasons, we set Θ to be the intersection of that set with a given discrete grid. Note that the value $\theta = (1, 0, 0, 0) \in \Theta$ will always return the same clustering and the same loss as the previous iteration. Thus, the loss is guaranteed to decrease monotonically at each iteration and the algorithm converges.

The intuition behind the introduction of the heuristic search parameter θ and the construction (2.20) of S^θ is as follows. If λ_Z and λ_{MU} are both very large, and the item partition g is correct, it is best to cluster the rows of $\tilde{M} + \tilde{C}$. Indeed, the items only exhibit community behaviour in those components. On the other hand, if the ground truth contains a large $\tilde{U}$ component (i.e., if there is significant interaction between user communities and specific items), or if the current item partition g is significantly wrong, then the component $Z + \tilde{U}$ will be more relevant to the clustering problem. We further split all components so we can look for solutions across a spectrum of confidence in the current partition (a very large θ_4 will reset the optimization procedure to a distant solution, whilst a large θ_1 will keep the current solution unchanged). Thus our algorithm includes a mix of incremental steps and explorative search.

Algorithm 5 Collaborative Clustering

INPUT: Partially observed matrix R_Ω and hyperparameters d_1,d_2, $\Lambda = \{\lambda_Z, \lambda_C, \lambda_{MU}\}$

OUTPUT: Cluster functions f and g

```
1:  $f = \text{null}, g = \text{null}$
2:  $Z = \arg\min_Z \mathcal{L}(R_\Omega, \Lambda, f, g)$
3:  $f_0 = \text{clusterRows}(Z, d_1),\ g_0 = \text{clusterColumns}(Z, d_2)$
4:  $\hat{R}_0 = \{C_0, M_0, U_0, Z_0\} = \arg\min_{C,M,U,Z} \mathcal{L}(R_\Omega, \Lambda, f_0, g_0)$
5:  repeat
6:      MAKE $\tilde{C_0}, \tilde{M_0}, \tilde{U_0}$ FROM $\hat{R}_0, f_0, g_0$
7:      $f = f_0, g = g_0, \ell_0 = \mathcal{L}(R_\Omega, \hat{R}_0, \Lambda, f_0, g_0)$
8:      for $\theta \in \Theta$ do
9:          $S^\theta = \theta_1 \tilde{C} + \theta_2 \tilde{M} + \theta_3 \tilde{U} + \theta_4 Z$
10:         $f_\theta = \text{clusterRows}(S^\theta, d_1),\ g_\theta = \text{clusterColumns}(S^\theta, d_2)$
11:         $\hat{R}_\theta = \{C_\theta, M_\theta, U_\theta, Z_\theta\} = \arg\min_{C,M,U,Z} \mathcal{L}(R_\Omega, \Lambda, f_\theta, g_\theta)$
12:         $\ell_\theta = \mathcal{L}(R_\Omega, \hat{R}_\theta, \Lambda, f_\theta, g_\theta)$
13:     end for
14:     $\theta_{\min} = \arg\min_\theta(\ell_\theta)$
15:     $\hat{R}_0 = \hat{R}_{\theta_{\min}}, f_0 = f_{\theta_{\min}}, g_0 = g_{\theta_{\min}}$
16: until $f_0 == f$ and $g_0 == g$
17: return $f$,$g$
```

2.4.1 Hyperparameter selection and scalability

The relevant hyperparameters in our model are $d_1, d_2, \Lambda = \{\lambda_Z, \lambda_C, \lambda_{MU}\}$. In practice, they can later be determined through *cross-validation*. Note that the cross-validation procedure can be executed in parallel: different sets of $(d_1, d_2, \Lambda,)$ can be fitted separately. In the case of d_1 and d_2, it is not necessary to run the full algorithm for each combination. Indeed, note that the choice of d_1 and d_2 is likely to have a large effect on the optimal loss for typical values of Λ. Thus, a promising strategy is to run a rudimentary version of our algorithm (e.g., with a single clustering step) for several d_1's and d_2's, and select the best performing values.

Regarding the for loop in Algorithm 5 (lines 8-13) observe that the iteration $i+1$ does not depend on iteration i. In this case, small adjustments also allow these steps to be executed in parallel, significantly reducing the computing burden of the search for the parameters Θ.

Note that line 11, which requires performing OMIC algorithm to solve the version of problem (2.6) for known f, g, can be greatly accelerated with warm starts (see Algorithm 2): the full recovered matrix from the previous iteration (of the repeat loop) is used as a warm start for each value of Θ, so that only a small number of imputations is required. Similar to other involved optimization algorithms[4], further improvements can be performed

---

[4] such as architecture search for neural networks

if necessary. For instance, one could initially select the optimal value of Θ based on an even smaller number of imputations, and perform a more thorough imputation procedure on the chosen Θ before moving to the next iteration of the repeat loop.

2.5 Conclusion

We presented a framework for cluster-induced matrix completion. Due to the presence of side information, our approach is context-aware. Comprehensively, we are the first to integrate bias, side information, and pure low-rank terms into a single framework using a well-principled optimization approach.

Additionally, we provided an algorithm to deal with our complex optimization problem and show that it is convergent regardless of initialization. To handle large matrices, we extend our technique to a scalable variant that computes matrices in sparse plus low-rank structure.

Finally, we developed a strategy for recovering latent clusters in the absence of side information. Contrasting previous works, by exploring cluster detection and matrix completion together, we proposed the coexistence of a cluster side information term and a pure low-rank term in the same matrix.

Chapter 3

THEORETICAL ANALYSIS

This chapter presents a theoretical analysis of OMIC, a method described in Chapter 2. We began by proving the uniqueness of the decomposition of the predictor. Then, we provide theoretical guarantees in the form of generalizations bounds.

The **main** contributions of this chapter are the following:

- we prove the uniqueness of decomposition of OMIC's predictor. Such property allows our method to give rise to interpretable solutions.
- we provide generalization bounds in the distribution-free case (where we make no assumption on the ground truth distribution). The better our model matches the ground truth, the tighter the bounds.
- we present a theoretical analysis for the cluster side information case, assuming that the sampling distribution of the entries is known and employing an adjusted nuclear norm regulariser.

Parts of this chapter are based on:

> Antoine Ledent*, **Rodrigo Alves***, and Marius Kloft. Orthogonal inductive matrix completion. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
>
> Antoine Ledent, **Rodrigo Alves**,Yunwen Lei, and Marius Kloft. Fine-grained Generalisation Analysis of Inductive Matrix Completion. *Advances in Neural Information Processing Systems (NeurIPS)*, 34, 2021 (**To appear**).

* The authors contributed equally to this research.

3.1 Uniqueness of decomposition of the predictor

We note that our orthogonality constraints offer the additional advantage of interpretability: the spaces $\left\{X^{(k)}M(Y^{(l)})^\top \middle| M \in \mathbb{R}^{d_1^{(k)} \times d_2^{(l)}}\right\}$ are orthogonal with respect to the Frobenius inner product. Thus, each ground truth matrix R has a unique representation $R = \sum_{k=1}^K \sum_{l=1}^L X^{(k)}R^{(k,l)}(Y^{(l)})^\top$. Bellow, we provide the detailed proof of these results:

Proposition 3.1.1. *Let* $\mathcal{F}_{k,l} = \{R : \exists M \in \mathbb{R}^{d_k^1 \times d_l^2} : R = X^{(k)}M(Y^{(l)})^\top\}$ *denote the KL subspaces corresponding to each pair of auxiliary matrices* $(X^{(k)}, Y^{(l)})$*. Those vector spaces* $\mathcal{F}_{k,l}$ *are orthogonal (w.r.t. the Frobenius inner product) and their direct sum is the whole of* $\mathbb{R}^{m\times n}$*:*

$$\bigoplus \mathcal{F}_{k,l} = \mathbb{R}^{m\times n}. \tag{3.1}$$

In particular, for any $R \in \mathbb{R}^{m\times n}$*, there exist a unique collection of matrices* $R^{(k,l)} \in \mathcal{F}_{k,l}$ *such that* $R = \sum_{k,l} R^{(k,l)}$*. In fact,*

$$R^{(k,l)} = (X^{(k)})^\top R Y^{(l)}. \tag{3.2}$$

Proof. We divide the proof into two parts: the proof that the subspaces are mutually orthogonal and the proof that their direct sum is the whole of the matrix space.
The subspaces $\mathcal{F}^{k,l}$ are mutually orthogonal. Let $A \in \mathcal{F}^{k,l}$ and $B \in \mathcal{F}^{k',l'}$ where either $k \neq k'$ or $l \neq l$. By definition of the subspaces in question, there exist $M \in \mathbb{R}^{d_k^1 \times d_l^2}$ and $N \in \mathbb{R}^{d_{k'}^1 \times d_{l'}^2}$ such that $A = X^{(k)}M(Y^{(l)})^\top$ and $B = X^{(k')}N(Y^{(l')})^\top$.

We now compute the (Frobenius) inner product between A and B in both cases.
Case 1: $l \neq l'$, in which case by the assumption on $Y^{(1)}, \ldots Y^{(l)}$, we have $(Y^{(l)})^\top Y^{(l')} = \mathbf{0} \in \mathbb{R}^{d_l^2 \times d_{l'}^2}$. Then,

$$\begin{aligned}
\langle A, B\rangle &= \operatorname{Tr}\left(X^{(k)}M(Y^{(l)})^\top (X^{(k')}N(Y^{(l')})^\top)^\top\right) \\
&= \operatorname{Tr}\left(X^{(k)}M(Y^{(l)})^\top Y^{(l')}N^\top (X^{(k')})^\top\right) \\
&= \operatorname{Tr}\left(X^{(k)}M\mathbf{0}N^\top (X^{(k')})^\top\right) \\
&= 0
\end{aligned}$$

Case 2: $k \neq k'$

$$\begin{aligned}\langle A, B\rangle &= \operatorname{Tr}\left((X^{(k')}M(Y^{(l)})^\top)^\top X^{(k)}N(Y^{(l)})^\top\right)\\ &= \operatorname{Tr}\left(Y^{(l)}M^\top(X^{(k')})^\top X^{(k)}M(Y^{(l)})^\top\right)\\ &= \operatorname{Tr}\left(Y^{(l)}M^\top\mathbf{0}M(Y^{(l)})^\top\right) = 0,\end{aligned}$$

as expected.

The direct sum is the whole of $\mathbb{R}^{m\times n}$: $\bigoplus_{k,l}\mathcal{F}^{k,l} = \mathbb{R}^{m\times n}$. Let $R \in \mathbb{R}^{m\times n}$. For each column vector $v \in \mathbb{R}^m$ we have immediately $v = \sum_k X^{(k)}(X^{(k)})^\top v$ by our assumption on the X's. Applying this to each column of R, we have $R = \sum_k X^{(k)}(X^{(k)})^\top R$. Similarly, $R = \sum_l RY^{(l)}(Y^{(l)})^\top$. Plugging the second equation into the first one, we obtain

$$\begin{aligned}R &= \sum_k X^{(k)}(X^{(k)})^\top R\\ R &= \sum_k X^{(k)}(X^{(k)})^\top \sum_l R(Y^{(l)}(Y^{(l)})^\top)\\ &= \sum_{k,l} X^{(k)}\left[(X^{(k)})^\top RY^{(l)}\right](Y^{(l)})^\top\\ &\in \bigoplus_{k,l}\mathcal{F}_{k,l},\end{aligned}$$

as expected (this also proves equality (3.2)). ☐

3.2 Learning-theoretical guarantees under distribution-free sampling

3.2.1 Generalization bounds

In this section, we focus on generalization bounds which apply to the comprehensive method with biases plus cluster side information (Section 2.1.3). We can easily extract similar bounds that hold for the instances presented in Sections 2.1.1 and 2.1.2 (see Section 3.2.2).

We assume a distribution-free approach, meaning that we do not have any assumption about the sampling distribution. We will write $C_{k,l}$ for an upper bound on the entries of the ground truth component $X^{(k)}R^{(k,l)}(Y^{(l)})^\top$ where $R^{(k,l)} = (X^{(k)})^\top RY^{(l)}$. Similarly, we will write $r_{k,l}$ for the rank of $R^{(k,l)}$. Thus, e.g., if $C_{2,3}$ is large, one concludes that in the ground truth matrix, the specific affinities of items in $\{1, 2, \ldots n\}$ to a whole cluster of users is a significant factor in determining the value of each entry. If $C_{3,3}$ is large, the individual affinities between users and items, independently of their respective clusters, is a strong factor.

The following follows from the Theorem 3.2.1 in the Section 3.2.2.

Corollary 3.2.1. *Consider the comprehensive setting (Section 2.1.3)and assume the user (resp. item) communities are of sizes within a ratio of* $O(1)$*, as well as that (without loss of generality)* $b \geqslant a$*, and* $m \geqslant n$*. For any* $\epsilon > 0$*, the required number of entries to recover the ground truth matrix within* ϵ *expected loss (with probability* $\geqslant 1 - \delta$*) is*

$$O\Bigg((1/\epsilon^2)\Bigg(C_{1,1}^2 + C_{1,2}^2 b + C_{1,3}^2 n + C_{2,1}^2 a + C_{2,2}^2 b\sqrt{a r_{2,2}} + C_{2,3}^2 n\sqrt{a r_{2,3}} \\ + C_{3,1}^2 m + C_{3,2}^2 m\sqrt{b r_{3,2}} + C_{3,3}^2 m\sqrt{r_{3,3} n} + \log(1/\delta)\Bigg)\Bigg).$$

Alternatively, using the nuclear norms of the component matrices, we have the following sample complexity bound:

$$O\Bigg((1/\epsilon^2)\Bigg(C_{1,1}\mathcal{M}_{1,1} + \sqrt{b}C_{1,2}\mathcal{M}_{1,2} + \sqrt{n}C_{1,3}\mathcal{M}_{1,3} + \sqrt{a}C_{2,1}\mathcal{M}_{2,1} + \sqrt{b}C_{2,2}\mathcal{M}_{2,2} + \sqrt{n}C_{2,3}\mathcal{M}_{2,3} \\ + \sqrt{m}C_{3,1}\mathcal{M}_{3,1} + \sqrt{m}C_{3,2}\mathcal{M}_{3,2} + \sqrt{m}C_{3,3}\mathcal{M}_{3,3} + \log(1/\delta)\Bigg)\Bigg),$$

where $\mathcal{M}_{k,l}$ *is the nuclear norm of the matrix* $R^{(k,l)}$*.*

3.2.2 Proof of the generalization bounds

Notation: In this section, we assume the entries are sampled with i.i.d. noise, so that observations of entry $R_{i,j}$ are of the form $R_{i,j} + \delta_{i,j}$ for $\delta_{i,j} \sim \Delta_{i,j}$ for some noise distribution $\Delta_{i,j}$[5]. Thus, N i.i.d. observations indexed by $\alpha \in \{1, 2, \ldots, N\}$ are denoted by $R_{i_\alpha, j_\alpha} + \delta_\alpha$ where (i_α, j_α) is the α'th i.i.d. choice of entry, and each δ_α is drawn from $\Delta_{i_\alpha, j_\alpha}$ independently. The loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$ is bounded by a constant B, with Lipschitz constant bounded by L_ℓ. For all $k \leqslant K, l \leqslant L, i \leqslant m, j \leqslant n$, we will write $\mathbf{x}_i^k$ (resp. $\mathbf{y}_j^l$) for the ith row (resp. jth column) of the matrix $X^{(k)}$ (resp. $Y^{(l)}$), $\mathcal{X}^{(k)}$ for $\max_{i=1}^m \|\mathbf{x}_i^k\|_2 = \max_{i=1}^m \|\mathbf{x}_i^k\|_2$ and $\mathcal{Y}^{(l)}$ for $\max_{i=1}^n \|\mathbf{y}_j^l\|_2$. For a predictor $f : \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to \mathbb{R}$, we will write $\mathcal{R}(f)$ for the expected risk $\mathbb{E}_{(i,j)\sim\mathcal{D}}(\ell(f(i,j), R_{i,j} + \delta_{i,j})$ and $\hat{\mathcal{R}}(f)$ for the empirical risk $(1/N)\sum_{\alpha=1}^N \ell(f(i,j), R_{i_\alpha, j_\alpha} + \delta_\alpha)$.

---

[5] Furthermore, $R_{i,j}$ and $\Delta_{i,j}$ are defined so that $R_{i,j} = \arg\min_y \mathbb{E}_{\delta_{i,j}}(\ell(R_{i,j} + \delta_{i,j}, y))$.

First, let us recall that for any $x_1, \ldots, x_N$ and any function class $\mathcal{F}$ we can define the (data dependent) Rademacher complexity $\mathfrak{R}_{(x_1,\ldots,x_n)}(\mathcal{F})$ as

$$\mathfrak{R}_{(x_1,\ldots,x_n)}(\mathcal{F}) := \mathbb{E}_\sigma \sup_{f \in \mathcal{F}} \frac{1}{N} \sum_{i=1}^{N} \sigma_i f(x_i), \tag{3.3}$$

where the σ_i's are i.i.d. Rademacher random variables (i.e. $\mathbb{P}(\sigma_i = 1) = \mathbb{P}(\sigma_i = -1) = 0.5$). Then, recall the following lemma from [38, 61].

Lemma 3.2.2. *Let a matrix $R \in \mathbb{R}^{m \times n}$, which is observed with i.i.d. noise, so that observing entry (i, j) results in an output of $R_{i,j} + \delta$ where $\delta \sim \Delta_{i,j}$ where the $\Delta_{i,j}$ are distributions. Let $F_{\mathcal{M}}$ be the set of matrices $\tilde{R} \in \mathbb{R}^{m \times n}$ with $\|\tilde{R}\|_* \leqslant \mathcal{M}$ and $|\tilde{R}_{i,j}| \leqslant C$. Let us write the data-dependent Rademacher complexity for N samples indexed by $\alpha \in \{1, 2, \ldots, N\}$ by $\mathfrak{R}_N(F_{\mathcal{M}}) = \mathbb{E}\left(\sup_{\tilde{R} \in F_{\mathcal{M}}} \frac{1}{N} \sum_{\alpha=1}^{N} \sigma_\alpha \tilde{R}_{i_\alpha, j_\alpha}\right)$, where the σ_α are independent Rademacher random variables, and the (i_α, j_α) are entries sampled independently.*

We have the following bound on the expected complexity $\mathfrak{R} = \mathbb{E}_\Omega(\mathfrak{R}_N(F_{\mathcal{M}}))$:

$$\mathfrak{R} = \mathbb{E}_\Omega(\mathfrak{R}_N(F_{\mathcal{M}})) \leqslant \sqrt{\frac{9\mathcal{M}C\mathcal{C}(\sqrt{m} + \sqrt{n})}{N}}.$$

Here, $\mathcal{C}$ is the universal constant from [62].

We now can show the following lemma regarding cluster side information. The central concept is to absorb the variation between entries that correspond to the same cluster into the "noise" of a corresponding problem that we can apply Lemma 3.2.2.

Lemma 3.2.3. *Let $X \in \mathbb{R}^{m \times a}$ and $Y \in \mathbb{R}^{n \times b}$ be auxiliary matrices whose columns are indicator functions of distinct sets forming partitions $\{c_1, c_2, \ldots, c_a\}$ and $\{s_1, \ldots, s_b\}$ of $\{1, 2, \ldots, n\}$ and $\{1, 2, \ldots, m\}$ respectively. Set $\mathcal{M} > 0$ and consider the function class $\mathcal{F}_{\mathcal{M}} := \left\{XMY^\top \middle| \|M\|_* \leqslant \mathcal{M}\right\}$. The Rademacher complexity $\mathfrak{R}_N(\mathcal{F}_{\mathcal{M}})$ satisfies*

$$\mathfrak{R} = \mathbb{E}_\Omega(\mathfrak{R}_N(F_{\mathcal{M}})) \leqslant \sqrt{\frac{9C\mathcal{M}\mathcal{C}\left(\sqrt{a} + \sqrt{b}\right)}{N}}. \tag{3.4}$$

where C is a bound on the predicted entries.

Proof. This follows from Lemma 3.2.2 applied to the following modified problem: let $f(i)$ (resp. $g(j)$) be the cluster to which user i (resp. item j) belongs for all $i \leqslant m$ (resp. $j \leqslant n$). For $u \leqslant a$ and $v \leqslant b$, an observation $\overline{R_{u,v}}$ will have the distribution $\left\{R_{i,j} + \Delta_{i,j} \middle| (i, j) \sim \mathcal{D}_{u,v}\right\}$ where $\mathcal{D}_{u,v}$ is the distribution $\mathcal{D}$ conditioned on $u = f(i)$ and $v = g(j)$. We note that Lemma 3.2.2 allows for the observations of $\bar{R}_{u,v}$ to be perturbed by random variables with distributions $\overline{\Delta_{u,v}}$ conditioned on (u, v). The differences between the values of the ground truth matrix at different pairs (i, j) where the communities of i and j are fixed can be absorbed into this perturbation. ☐

Proposition 3.2.4. *Let $A \in \mathbb{R}^{m\times n}$ be a matrix and let $v \in \mathbb{R}^m$ and $w \in \mathbb{R}^n$ be two vectors. We have*

$$\|vw^\top \odot A\|_* \leqslant \max_{i,j} |v_i||w_j|\|A\|_* \tag{3.5}$$

where $\odot$ denotes the Hadamard (entry wise) product.

For the proof of Proposition 3.2.4, we will need the following lemma (Lemma 6 from [29], see also [28]):

Lemma 3.2.5. *For any matrix Z, the following holds:*

$$\begin{aligned}\|Z\|_* &= \min_{\substack{U,V;\\ UV^\top = Z}} \|U\|_{\mathrm{Fr}}\|V\|_{\mathrm{Fr}} \\ &= \min_{\substack{U,V;\\ UV^\top = Z}} \frac{1}{2}\left(\|U\|_{\mathrm{Fr}}^2 + \|V\|_{\mathrm{Fr}}^2\right) \end{aligned} \tag{3.6}$$

Now we will prove Proposition 3.2.4.

Proof. By an equivalent formulation of the nuclear norm 3.2.5 we have:

$$\begin{aligned}\|vw^\top \odot A\|_* &= \min\left(\|B\|_{\mathrm{Fr}}\|C\|_{\mathrm{Fr}}; BC^\top = vw^\top \odot A\right) \\ &\leqslant \min\left(\|\operatorname{diag}(v)B\|_{\mathrm{Fr}}\|\operatorname{diag}(w)C\|_{\mathrm{Fr}}; \operatorname{diag}(v)B(\operatorname{diag}(w)C)^\top = vw^\top \odot A\right) \\ &= \min\left(\|\operatorname{diag}(v)B\|_{\mathrm{Fr}}\|\operatorname{diag}(w)C\|_{\mathrm{Fr}}; (vw^\top) \odot (BC^\top) = vw^\top \odot A\right) \\ &\leqslant \min\left(\|\operatorname{diag}(v)B\|_{\mathrm{Fr}}\|\operatorname{diag}(w)C\|_{\mathrm{Fr}}; BC^\top = A\right) \\ &\leqslant \min\left(\max_i |v_i|\|B\|_{\mathrm{Fr}} \max_j |w_j|; \|\operatorname{diag}(w)C\|_{\mathrm{Fr}} \middle| BC^\top = A\right) \\ &= \max_{i,j} |v_i||w_j|\|A\|_*,\end{aligned}$$

as expected.

☐

We are now in a position to present the main results.

Theorem 3.2.1. *Consider the setting from Section 2.1.3 (in particular, $K = L = 3$). Let $\mathcal{K}$ denote the maximum ratio between the sizes of any two user or item clusters. Without loss of generality, assume $n \leqslant m$. Choose some $\mathcal{M}_{k,l}$ and $C_{k,l}$ such that $\|R^{(k,l)}\|_* \leqslant \mathcal{M}_{k,l}$ and $\max_{i,j} |R^{(k,l)}_{i,j}| \leqslant C_{k,l}$ for all (k,l). Let $\hat{f}$ be the solution to the optimization problem*

$$\begin{aligned} &\min \quad \hat{\mathcal{R}}(f) \text{ s.t. } \forall k,l, \\ &f = \sum_{k,l} (X^{(k)} M^{(k,l)} (Y^{(l)})^\top); \ \|M^{(k,l)}\|_* \leqslant \mathcal{M}_{k,l}; \\ &\text{and } \|X^{(k)} M^{(k,l)} (Y^{(l)})^\top\|_\infty \leqslant C_{k,l} \quad \forall k,l,i \text{ and } j. \end{aligned} \tag{3.7}$$

With probability $\geqslant 1-\delta$ over the draw of the training set, the solution to the optimization problem (3.7) satisfies

$$\begin{aligned} \mathcal{R}(\hat{f}) \leqslant 2L_\ell \sqrt{\frac{9\mathcal{C}}{N}} \times \Bigg[& \frac{\sqrt{2}}{\sqrt[4]{mn}} \sqrt{C_{1,1}\mathcal{M}_{1,1}} + \frac{1}{\sqrt[4]{cm}} \sqrt{C_{1,2}\mathcal{M}_{1,2}(1+\sqrt{b})} + \frac{1}{\sqrt[4]{m}} \sqrt{C_{1,3}\mathcal{M}_{1,3}(1+\sqrt{n})} \\ & + \frac{1}{\sqrt[4]{cn}} \sqrt{C_{2,1}\mathcal{M}_{2,1}(\sqrt{a}+1)} + \frac{1}{\sqrt{c}} \sqrt{C_{2,2}\mathcal{M}_{2,2}(\sqrt{a}+\sqrt{b})} + \frac{1}{\sqrt[4]{c}} \sqrt{C_{2,3}\mathcal{M}_{2,3}(\sqrt{a}+\sqrt{n})} \\ & + \frac{1}{\sqrt[4]{n}} \sqrt{C_{3,1}\mathcal{M}_{3,1}(\sqrt{m}+1)} + \frac{1}{\sqrt[4]{c}} \sqrt{C_{3,2}\mathcal{M}_{3,2}(\sqrt{m}+\sqrt{b})} + \sqrt{C_{3,3}\mathcal{M}_{3,3}(\sqrt{m}+\sqrt{n})} \Bigg] \\ & + 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E}. \end{aligned} \tag{3.8}$$

Expressed in terms of matrix ranks instead, we obtain:

$$\begin{aligned} \mathcal{R}(\hat{f}) \leqslant 2L_\ell \sqrt{\frac{9\mathcal{C}}{N}} \times \Bigg[& \sqrt{2}C_{1,1} + C_{1,2}\sqrt[4]{\mathcal{K}b}\sqrt{1+\sqrt{b}} + C_{1,3}\sqrt[4]{n}\sqrt{1+\sqrt{n}} \\ & + C_{2,1}\sqrt[4]{\mathcal{K}a}\sqrt{\sqrt{a}+1} + C_{2,2}\sqrt[4]{abr_{2,2}}\sqrt{\mathcal{K}}\sqrt{\sqrt{a}+\sqrt{b}} + C_{2,3}\sqrt[4]{\mathcal{K}anr_{2,3}}\sqrt{\sqrt{a}+\sqrt{n}} \\ & + C_{3,1}\sqrt[4]{m}\sqrt{\sqrt{m}+1} + C_{3,2}\sqrt[4]{\mathcal{K}mbr_{3,2}}\sqrt{\sqrt{m}+\sqrt{b}} + C_{3,3}\sqrt[4]{mnr_{3,3}}\sqrt{\sqrt{m}+\sqrt{n}} \Bigg] \\ & + 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E}. \end{aligned} \tag{3.9}$$

Here $\mathcal{C}$ is an absolute constant and c is the number of elements in the smallest cluster among all users and items' clusters together.

Proof. For convenience we start the proof by considering the setting from the Section 2.1.2 (in particular, $K = L = 2$, without the presence of biases). Let c_1 (resp. c_2) be the size of the smallest community of users (resp. items). Let $\tilde{X}^{(1)} \in \mathbb{R}^{m\times a}$ and $\tilde{Y}^{(1)} \in \mathbb{R}^{n\times b}$ denote matrices whose columns are the (non-normalised) indicator functions of the communities.

By Lemma 3.2.2, the Rademacher complexity of the function class $\{\tilde{X}^{(1)}M(\tilde{Y}^{(1)})^\top; \|M\|_* \leqslant \mathcal{M}_{1,1} \wedge \|\tilde{X}^{(1)}M(\tilde{Y}^{(1)})^\top\|_\infty \leqslant C_{1,1}\}$ is bounded by

$$\sqrt{\frac{9C_{1,1}\mathcal{M}_{1,1}\mathcal{C}(\sqrt{a}+\sqrt{b})}{N}}.$$

Now observe that by Lemma 3.2.4, the function class

$$\mathcal{F}_{1,1} := \{X^{(1)}M(Y^{(1)})^\top | \|M\|_* \leqslant \mathcal{M}_{1,1} \wedge \|X^{(1)}M(Y^{(1)})^\top\|_\infty \leqslant C_{1,1}\}$$

satisfies

$$\mathcal{F}_{1,1} \subset \widetilde{\mathcal{F}_{1,1}} := \{\tilde{X}^{(1)}M(\tilde{Y}^{(1)})^\top; \|M\|_* \leqslant \mathcal{M}_{1,1}c_1^{-1/2}c_2^{-1/2} \wedge \|\tilde{X}^{(1)}M(\tilde{Y}^{(1)})^\top\|_\infty \leqslant C_{1,1}\},$$

where c_1 (resp. c_2) is the size of the smallest community of users (resp. items). Recall that $c = \min(c_1, c2)$. Then, it follows that

$$\mathfrak{R}(\mathcal{F}_{1,1}) \leqslant \sqrt{\frac{9C_{1,1}(\mathcal{M}_{1,1}/\sqrt[2]{c_1c_2})\mathcal{C}(\sqrt{a}+\sqrt{b})}{N}} \leqslant \frac{1}{\sqrt{c}}\sqrt{\frac{9C_{1,1}\mathcal{M}_{1,1}\mathcal{C}(\sqrt{a}+\sqrt{b})}{N}}$$

By the same argument applied to the two situations where each user or item is a single community, we obtain the following results for $\mathcal{F}_{1,2} := \{X^{(1)}M(Y^{(2)})^\top | \|M\|_* \leqslant \mathcal{M}_{1,2} \wedge \|X^{(1)}M(Y^{(2)})^\top\|_\infty \leqslant C_{1,2}\}$, $\mathcal{F}_{2,1} := \{X^{(2)}M(Y^{(1)})^\top | \|M\|_* \leqslant \mathcal{M}_{2,1} \wedge \|X^{(2)}M(Y^{(1)})^\top\|_\infty \leqslant C_{2,1}\}$ and $\mathcal{F}_{2,2} := \{X^{(2)}M(Y^{(2)})^\top | \|M\|_* \leqslant \mathcal{M}_{2,2} \wedge \|X^{(2)}M(Y^{(2)})^\top\|_\infty \leqslant C_{2,2}\}$:

$$\mathfrak{R}(\mathcal{F}_{1,2}) \leqslant \mathfrak{R}(\widetilde{\mathcal{F}_{1,2}}) \leqslant \frac{1}{\sqrt[4]{c}}\sqrt{\frac{9C_{1,2}\mathcal{M}_{1,2}\mathcal{C}(\sqrt{a}+\sqrt{n})}{N}};$$

$$\mathfrak{R}(\mathcal{F}_{2,1}) \leqslant \mathfrak{R}(\widetilde{\mathcal{F}_{2,1}}) \leqslant \frac{1}{\sqrt[4]{c}}\sqrt{\frac{9C_{2,1}\mathcal{M}_{2,1}\mathcal{C}(\sqrt{m}+\sqrt{b})}{N}};$$

and

$$\mathfrak{R}(\mathcal{F}_{2,2}) \leqslant \mathfrak{R}(\widetilde{\mathcal{F}_{2,2}}) \leqslant \sqrt{\frac{9C_{2,2}\mathcal{M}_{2,2}\mathcal{C}(\sqrt{m}+\sqrt{n})}{N}},$$

after noting that $\mathcal{F}_{1,2} \subset \widetilde{\mathcal{F}_{1,2}} := \{\tilde{X}^{(1)}M(\tilde{Y}^{(2)})^\top; \|M\|_* \leqslant \mathcal{M}_{1,2}c_1^{-1/2} \wedge \|\tilde{X}^{(1)}M(\tilde{Y}^{(2)})^\top\|_\infty \leqslant C_{1,2}\}$; $\mathcal{F}_{2,1} \subset \widetilde{\mathcal{F}_{2,1}} := \{\tilde{X}^{(2)}M(\tilde{Y}^{(1)})^\top; \|M\|_* \leqslant \mathcal{M}_{2,1}c_2^{-1/2} \wedge \|\tilde{X}^{(2)}M(\tilde{Y}^{(1)})^\top\|_\infty \leqslant C_{2,1}\}$; and $\mathcal{F}_{2,2} \subset \widetilde{\mathcal{F}_{2,2}} := \{\tilde{X}^{(2)}M(\tilde{Y}^{(2)})^\top; \|M\|_* \leqslant \mathcal{M}_{2,2} \wedge \|\tilde{X}^{(2)}M(\tilde{Y}^{(2)})^\top\|_\infty \leqslant C_{2,2}\}$. Here $\tilde{X}^{(2)}$ and $\tilde{Y}^{(2)}$ are identity matrices.

By considering the subadditivity of Rademacher complexity, Talagrand's lemma and the classic Rademacher theorem, we can calculate the empirical risk $\mathcal{R}(\hat{f})$ (setting from Section 2.1.2) directly as:

$$\begin{aligned}\mathcal{R}(\hat{f}) &\leqslant 2L_\ell\Big[\mathfrak{R}(\mathcal{F}_{1,1}) + \mathfrak{R}(\mathcal{F}_{1,2}) + \mathfrak{R}(\mathcal{F}_{2,1}) + \mathfrak{R}(\mathcal{F}_{2,2})\Big] + 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E} \\ &\leqslant 2L_\ell\sqrt{\frac{9\mathcal{C}}{N}}\Bigg[\frac{1}{\sqrt{c}}\sqrt{C_{1,1}\mathcal{M}_{1,1}(\sqrt{a}+\sqrt{b})} \\ &+ \frac{1}{\sqrt[4]{c}}\sqrt{C_{1,2}\mathcal{M}_{1,2}(\sqrt{a}+\sqrt{n})} \\ &+ \frac{1}{\sqrt[4]{c}}\sqrt{C_{2,1}\mathcal{M}_{2,1}(\sqrt{m}+\sqrt{b})} \\ &+ \sqrt{C_{2,2}\mathcal{M}_{2,2}(\sqrt{m}+\sqrt{n})}\Bigg] \\ &+ 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E}. \end{aligned} \tag{3.10}$$

Now observe that if we set the number of user and item communities to one, then we obtain exactly the model setting of Section 2.1.1 (only with biases). Then we can directly extract our second partial result:

$$\begin{aligned}\mathcal{R}(\hat{f}) \leqslant 2L_\ell\sqrt{\frac{9\mathcal{C}}{N}}\Bigg[&\frac{\sqrt{2}}{\sqrt[4]{mn}}\sqrt{C_{1,1}\mathcal{M}_{1,1}} \\ &+ \frac{1}{\sqrt[4]{m}}\sqrt{C_{1,2}\mathcal{M}_{1,2}(1+\sqrt{n})} \\ &+ \frac{1}{\sqrt[4]{n}}\sqrt{C_{2,1}\mathcal{M}_{2,1}(\sqrt{m}+1)} \\ &+ \sqrt{C_{2,2}\mathcal{M}_{2,2}(\sqrt{m}+\sqrt{n})}\Bigg] \\ &+ 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E}. \end{aligned} \tag{3.11}$$

By using 3.10 and 3.12, we reach our first result of the theorem (distribution free bounds for the comprehensive case described in Section 2.1.3):

$$
\begin{aligned}
\mathcal{R}(\hat{f}) &\leqslant 2L_\ell\Big[\sum_{k=1}^{3}\sum_{l=1}^{3}\ \mathfrak{R}(\mathcal{F}_{k,l})\Big] + 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E} \\
&\leqslant 2L_\ell\sqrt{\frac{9\mathcal{C}}{N}} \times \Bigg[\frac{\sqrt{2}}{\sqrt[4]{mn}}\sqrt{C_{1,1}\mathcal{M}_{1,1}} + \frac{1}{\sqrt[4]{cm}}\sqrt{C_{1,2}\mathcal{M}_{1,2}(1+\sqrt{b})} + \frac{1}{\sqrt[4]{m}}\sqrt{C_{1,3}\mathcal{M}_{1,3}(1+\sqrt{n})} \\
&+ \frac{1}{\sqrt[4]{cn}}\sqrt{C_{2,1}\mathcal{M}_{2,1}(\sqrt{a}+1)} + \frac{1}{\sqrt{c}}\sqrt{C_{2,2}\mathcal{M}_{2,2}(\sqrt{a}+\sqrt{b})} + \frac{1}{\sqrt[4]{c}}\sqrt{C_{2,3}\mathcal{M}_{2,3}(\sqrt{a}+\sqrt{n})} \\
&+ \frac{1}{\sqrt[4]{n}}\sqrt{C_{3,1}\mathcal{M}_{3,1}(\sqrt{m}+1)} + \frac{1}{\sqrt[4]{c}}\sqrt{C_{3,2}\mathcal{M}_{3,2}(\sqrt{m}+\sqrt{b})} + \sqrt{C_{3,3}\mathcal{M}_{3,3}(\sqrt{m}+\sqrt{n})}\Bigg] \\
&+ 2B\sqrt{\frac{\log(1/\delta)}{2N}} + \mathcal{E}. \qquad (3.12)
\end{aligned}
$$

Regarding the second result, we have $\mathcal{M}_{k,l} \simeq \sqrt{c_{1,k}c_{2,l}}C_{k,l}\sqrt{d_{1,k}d_{2,l}r_{k,l}}$, where $r_{k,l}$ is the rank of $M_{k,l}$ and $c_{1,k}$ ($c_{2,l}$) is the size of the largest community among $X^{(k)}$ ($Y^{(l)}$). If $X^{(k)}$ is a identity matrix, then $c_{1,k} = 1$. In the case of $X^{(k)}$ be a unity vectors $c_{1,k} = m$. Plugging this back into the first result yields the second result, as expected.

□

3.3 Learning-theoretical guarantees under a given distribution sampling

3.3.1 Adjusted nuclear norm regulariser

In this section, we introduce our adjusted nuclear norm regulariser. We first recall that in standard (non-inductive) matrix completion, the weighted nuclear norm [63, 64] of a matrix Z is defined as $\|\sqrt{D}Z\sqrt{E}\|_*$ where $D \in \mathbb{R}^{m\times m}$ (resp. $E \in \mathbb{R}^{n\times n}$) are diagonal matrices whose diagonal entries contain the marginal row (resp. column) sampling probabilities. Regularising the weighted nuclear norm instead of the standard nuclear norm leads to better theoretical guarantees.

The method is based on a careful distribution-dependent rescaling of the matrix M. The idea is that we must look at the principal directions (singular vectors) of the side information matrices, but computed with respect to a distribution-sensitive inner product: when computing inner products of vectors in the column space of x, components corresponding to highly users which are more likely to be sampled must be weighted more. In practical applications, one can observe that some movies have more ratings than others, and some users are more active than their counterparts.

Here we want to analyze the use of the adjusted nuclear norm regulariser in the presence of cluster side information. Let $p_{i,j}$ be the probability of sampling entry i, j. We define a diagonal matrix $D \in \mathbb{R}^{d_1 \times d_1}$ as follows: for each user cluster u we set $D_{u,u} = \sum_{j \leqslant n; f(i)=u} p_{i,j}$, where $f(i)$ denotes the cluster to which user-i belongs. In other words, $D_{u,u}$ is the marginal probability of hitting any entry whose user component belongs to cluster u. We define $E \in \mathbb{R}^{d_2 \times d_2}$ analogously. We can now write our predictors as

$$XMY^\top = XD^{-\frac{1}{2}}[D^{\frac{1}{2}}ME^{\frac{1}{2}}]E^{-\frac{1}{2}}Y^\top, \tag{3.13}$$

where X and Y are indicator matrices, respectively, of users and items community information. For instance, if user i belongs to the community c_i out of d_1 communities, the vector $x_i \in \mathbb{R}^{d_1}$ has 0s in all positions except position c_i, which has the value of 1.

Remark 3.3.1. *Although we consider the knowledge of the distribution, we could also define empirical versions of those quantities:* $\widehat{D}_{u,u} := \sum_{j \leqslant n; f(i)=u} \frac{h_{i,j}}{N}$*, where* $h_{i,j}$ *is the number of times that entry* (i,j) *was sampled, i.e.* $h_{i,j} = \sum_{o=1}^{N} 1_{\xi_o=(i,j)} = \#(\Omega \cap \{(i,j)\})$.

Therefore, the aim would be to regularise $[D^{\frac{1}{2}}ME^{\frac{1}{2}}]$ instead of M. However, some extra technical modifications may be necessary: if some users or items have extremely small sampling probability, the corresponding entries of $D^{\frac{1}{2}}$ and $E^{\frac{1}{2}}$ will be very large. We tackle this issue by forcing the entries of D and E to be bounded, which we achieve via smoothing. Fixing a parameter $\alpha \in [0,1]$, we define $\widetilde{D} = \alpha D + (1-\alpha)I/d_1$ and $\widetilde{E} = \alpha E + (1-\alpha)I/d_2$ where I is the identity matrix. We also define accordingly $\widetilde{M} = \widetilde{D}^{\frac{1}{2}}M\widetilde{E}^{\frac{1}{2}}$.

3.3.2 Generalization bounds

This section focuses on generalization bounds that apply to the adjusted nuclear norm regulariser proposed in the previous section. We assume the sampling distribution is known. The following follows from the Theorem 3.3.1 in Section 3.3.3.

Corollary 3.3.2. *Consider the setting described in Section 3.3.1. Assume that* $d = max(d_1, d_2)$*, the ground truth rank is* r *and the its entries are bounded by* C*. Assuming* $\widetilde{\mathcal{M}}$ *is cross validated within a constant factor of the optimal value, for any* $\epsilon > 0$*, the required number of entries to recover the ground truth matrix within* ϵ *expected loss (with probability* $\geqslant 1 - \delta$*) is*

$$O\Bigg((1/\epsilon^2)\Bigg(C^2 dr \log d\Bigg)\Bigg).$$

Alternatively, expressed in terms $\widetilde{\mathcal{M}}$*, we have the following sample complexity bound:*

$$O\Bigg((1/\epsilon^2)\Bigg(d\widetilde{\mathcal{M}}^2 \log d\Bigg)\Bigg).$$

3.3.3 Proof of the generalization bounds

Notation: In this section, we also assume the entries are sampled with i.i.d. noise, so that observations of entry $R_{i,j}$ are of the form $R_{i,j} + \delta_{i,j}$ for $\delta_{i,j} \sim \Delta_{i,j}$ for some noise distribution $\Delta_{i,j}$. The loss function $\ell : \mathbb{R} \times \mathbb{R} \to \mathbb{R}^+$ is bounded by a constant B, with Lipschitz constant bounded by L_ℓ. For a predictor $f : \{1, 2, \ldots, m\} \times \{1, 2, \ldots, n\} \to \mathbb{R}$, we will write $\mathcal{R}(f)$ for the expected risk $\mathbb{E}_{(i,j)\sim\mathcal{D}}(\ell(f(i,j), R_{i,j} + \delta_{i,j})$ and $\hat{\mathcal{R}}(f)$ for the empirical risk $(1/N)\sum_{\alpha=1}^{N} \ell(f(i,j), R_{i_\alpha,j_\alpha} + \delta_\alpha)$. Finally, the predictors f are restricted to the class $\widetilde{\mathcal{F}}_r = \Big\{XMY^\top : \|\widetilde{M}\|_* \leqslant \widetilde{\mathcal{M}}\Big\}$ for some real number $\mathcal{M} \in \mathbb{R}^+$.

Theorem 3.3.1. *Let* $X \in \mathbb{R}^{m\times d_1}$ *and* $Y \in \mathbb{R}^{n\times d_2}$ *be cluster* indicator *matrices, respectively, of users and items. Without loss of generality, assume* $n \leqslant m$ *and let* d =max(d_1, d_2)*. Let* $\hat{f}$ *be the solution to the optimization problem:*

$$\begin{aligned} \min \quad & \hat{\mathcal{R}}(f) \\ f = \widetilde{\mathcal{F}}_r = & \Big\{XMY^T : \|\widetilde{M}\|_* \leqslant \widetilde{\mathcal{M}}\Big\} \ . \end{aligned} \tag{3.14}$$

Assume also that the ground truth matrix R *is realisable, i.e. there exists an* $\bar{M}$ *such that* $R = X\bar{M}Y^\top$ *and* $\|\bar{M}\|_* \leqslant \widetilde{\mathcal{M}}$*. (In particular,* $\widetilde{\mathcal{M}}$ *must be chosen large enough via cross validation).*

With probability $\geqslant 1-\delta$ *over the draw of the training set, the solution to the optimization problem* (3.14) *satisfies*

$$\mathcal{R}(\hat{f}) \leqslant O\left(\frac{L_\ell \widetilde{\mathcal{M}}\sqrt{d}(1+\sqrt{\log(2d)})}{\sqrt{N}} + B\sqrt{\frac{\log(2/\delta)}{N}}\right). \tag{3.15}$$

Expressed in terms of matrix ranks instead, we obtain:

$$\mathcal{R}(\hat{f}) \leqslant O\left(\frac{L_\ell \sqrt{r}C\sqrt{d}(1+\sqrt{\log(2d)})}{\sqrt{N}} + B\sqrt{\frac{\log(2/\delta)}{2N}}\right). \tag{3.16}$$

Proof. First let us present the following lemma extracted from the results of [64]:

Lemma 3.3.3. *Let a matrix $R \in \mathbb{R}^{m\times n}$, which is observed with i.i.d. noise, so that observing entry (i,j) results in an output of $R_{i,j} + \delta$ where $\delta \sim \Delta_{i,j}$ where the $\Delta_{i,j}$ are distributions. Without loss of generality, assume $n \leqslant m$. For any matrix Z we set $\widetilde{Z} = \widetilde{D}^{\frac{1}{2}} Z \widetilde{E}^{\frac{1}{2}}$. Let $\hat{f}$ be the solution to the following optimization problem*

$$\begin{aligned} &\min \quad \hat{\mathcal{R}}(f) \\ &f = \widetilde{\mathcal{F}}_{\widetilde{\mathcal{M}}} = \left\{ Z : \|\widetilde{Z}\|_* \leqslant \widetilde{\mathcal{M}} \right\}. \end{aligned} \tag{3.17}$$

Assume also $\widetilde{\mathcal{M}}$ is chosen (via cross-validation) to be large enough to ensure the ground truth matrix R is realisable, i.e. $\|Z\|_* \leqslant \widetilde{\mathcal{M}}$. With probability $\geqslant 1-\delta$ over the draw of the training set, the solution to the optimization problem (3.17) satisfies

$$\mathcal{R}(\hat{f}) \leqslant O\left(\frac{L_\ell \widetilde{\mathcal{M}}\sqrt{m}(1+\sqrt{\log(2m)})}{\sqrt{N}} + B\sqrt{\frac{\log(2/\delta)}{2N}}\right). \tag{3.18}$$

Similarly to the proof of Lemma 3.2.3, from Lemma 3.3.3 we applied to the following modified problem: let $f(i)$ (resp. $g(j)$) be the cluster to which user i (resp. item j) belongs for all $i \leqslant m$ (resp. $j \leqslant n$). For $u \leqslant a$ and $v \leqslant b$, an observation $\overline{R_{u,v}}$ will have the distribution $\left\{R_{i,j} + \Delta_{i,j} \middle| (i,j) \sim \mathcal{D}_{u,v}\right\}$ where $\mathcal{D}_{u,v}$ is the distribution $\mathcal{D}$ conditioned on $u = f(i)$ and $v = g(j)$. We also note that Lemma 3.3.3 allows for the observations of $\bar{R}_{u,v}$ to be perturbed by random variables with distributions $\overline{\Delta_{u,v}}$ conditioned on (u,v). The differences between the values of the ground truth matrix at different pairs (i,j) where the communities of i and j are fixed can be absorbed into this perturbation. Therefore, the first result yields from Lemma 3.3.3 by solving a problem where R has dimensions $d_1 \times d_2$, if $d_1 \geqslant d_2$. If $d_2 > d_1$, the Lemma 3.3.3 applies to $R^\top$.

Finally, to express the final result in terms of matrix ranks, we note that $\widetilde{\mathcal{M}}$ scales like $C\sqrt{r}$ where r and C are bounds on the rank and maximum entries, respectively. Indeed, note

$$\|\widetilde{D}^{\frac{1}{2}}\|_{\mathrm{Fr}}^2 \leqslant d_1 \frac{1}{2d_1} + \frac{1}{2}\sum \sqrt{D_{u,u}}^2 = (1/2) + (1/2) = 1, \tag{3.19}$$

and similarly $\|\widetilde{E}^{\frac{1}{2}}\|_{\mathrm{Fr}}^2 \leqslant 1$. Thus if $\|M\|_\infty \leqslant C$ and $\mathrm{rank}(M) \leqslant r$, we have $\|\widetilde{M}\|_* \leqslant \sqrt{r}\|\widetilde{M}\|_{\mathrm{Fr}} \leqslant \sqrt{r}\sqrt{\sum_{u,v}[\widetilde{D}_u^{\frac{1}{2}}]^2[\widetilde{E}_v^{\frac{1}{2}}]^2 M_{u,v}^2} \leqslant \sqrt{r}\|M\|_\infty\sqrt{\sum_{u,v}[\widetilde{D}_u^{\frac{1}{2}}]^2[\widetilde{E}_v^{\frac{1}{2}}]^2} \leqslant \sqrt{r}C$. Plugging back, we have the second result, as expected.

□

3.4 Remarks on proof techniques and comparison to IMC bounds

The bounds admittedly follow reasonably straightforwardly from similar techniques as those used for standard matrix completion, applied to auxiliary problems corresponding to each of the four terms ($X^{(2)}M^{(2,2)}(Y^2)^\top$, $X^{(2)}M^{(2,3)}(Y^3)^\top$, $X^{(3)}M^{(3,2)}(Y^2)^\top$ and $X^{(3)}M^{(3,3)}(Y^3)^\top$) independently and then merged via the subadditivity of Rademacher complexity. Indeed, in the distribution-free case state-of-the-art bounds for the number of known entries take the form (cf. [38, 61])

$$O\left(\mathcal{M}(\sqrt{n} + \sqrt{m})\right) \tag{3.20}$$

where $\mathcal{M}$ is a bound on the nuclear norm.

However, we note that the state-of-the-art bounds for *inductive matrix completion* (whose predictors take the form $XMY^\top$ for some fixed side information X and Y, with nuclear norm minimization at work on M), applied to any of the first three terms in question do **not** yield bounds as tight as ours.

Indeed, there is no suitable equivalent to (3.20) in the inductive case, and the state-of-the-art bounds for inductive matrix completion (cf. [34, 65]). The main contribution of our bounds is to provide specific learning guarantees for IMC with cluster side information. To better understand this phenomenon, consider the case where $C_{1,l} = C_{k,1} = 0$. Note that, in this case, the bounds presented in Corollary 3.2.1 improve with the quality of the side information: the better the ground truth matrix can be approximated by community behaviour, the closer the bound behaves to the bound one would obtain for an $a \times b$ matrix with each user and item being assimilated to its community. We note that for a fixed tolerance threshold ϵ, our distribution-free sample complexity bounds scale as

$$\mathcal{K}C_{2,2}^2 b\sqrt{ar_{2,2}} + \sqrt{\mathcal{K}}C_{2,3}^2 n\sqrt{ar_{2,3}} + \sqrt{\mathcal{K}}C_{3,2}^2 m\sqrt{br_{3,2}} + C_{3,3}^2 m\sqrt{r_{3,3}n}, \tag{3.21}$$

It means that, if $C_{2,2}$ is much larger than $C_{2,3}, C_{3,2}$ and $C_{3,3}$, the bound behaves similarly to the situation where the users and items are identified with their category. Whilst this is not very surprising, the bound does show that this remains true for small but non zero values of $C_{2,3}, C_{3,2}$ and $C_{3,3}$: the model can effectively learn a combination of community behaviour and user behaviour with no further difficulty than if it were learning both problems independently. Note also conversely that if a, b are very small (as in the particular case of biases where $a = b = 1$), the first three terms of (3.21) are very small and the bound essentially tells us that the model is about as hard as learning the low-rank residual alone.

Moreover, the bounds show that prior knowledge of the community structure helps more than knowledge of the generic low-rank structure. Indeed, consider a typical situation where the communities are of equal and small size, and also $C_{2,3} = C_{3,2} = 0$, $a = b$, $m = n$ and $r_{2,2} = a$, and assume that the absolute values of the maximum and minimum entries of $R^{(2,2)}$ are of the same order so that the maximum absolute value of an entry of R is $\simeq C_{2,2} + C_{3,3}$. With knowledge of the community structure, our model requires in the distribution-free case $O(C_{2,2}^2 a^2 + C_{3,3}^2 m^{3/2}\sqrt{r_{3,3}})$ entries. If we were to apply a generic low-rank method instead, the number of required entries would then be $O((C_{2,2} + C_{3,3})^2 m\sqrt{(r_{3,3} + a)m})$.

To better understand the impact of our bounds under a known distribution and applying adjusted nuclear norm regulariser (Section 3.3.3), consider the representative case when only $C_{2,2} \neq 0$, our distribution-free bound (3.8). In this case, our bound on the number of require entries takes the form

$$O\left(C_{2,2}^2 b\sqrt{a r_{2,2}}\right), \tag{3.22}$$

whereas knowing the distribution, our bound (3.3.2) takes the form

$$O\left(C^2 br\log b\right). \tag{3.23}$$

By ignoring log terms, assuming $a = b$ and $b \gg r$, we see that adjusted nuclear norm regulariser bounds scales as Cb instead of $Cb^{3/2}$.

3.5 Conclusion

In this chapter, we performed a theoretical analysis of OMIC. We, first, proved the uniqueness of the predictor's decomposition, which justifies our experimental interpretability analysis. For that, we proved that the subspaces are mutually orthogonal and that their direct sum is the whole of the matrix space.

Then, in the form of generalization bounds, we provide theoretical guarantees for our method when no assumption on the ground truth sampling distribution. Our bounds improve with the quality of the side information: the better the ground truth matrix can be approximated by community behaviour, the closer the bound behaves to the bound one would obtain for an $a \times b$ matrix with each user and item being assimilated to its community. Finally, we presented an analysis when the entries sample distribution is known and an adjusted nuclear norm regulariser is applied: our results show that the bounds with the knowledge of the sampling distribution scales as C^2b instead of $C^2b^{3/2}$, when no assumption about the sampling distribution is made.

Chapter 4

EXPERIMENTS AND APPLICATIONS

This chapter presents synthetic and real-world data experiments, as well as applications of the methods described in Chapter 2. We began by analyzing the OMIC algorithm in different ground-truth regimes with synthetic datasets. Then, we verify our method's performance in a context-aware and collaborative-filtering scenario on recommender datasets. Finally, we present a novel application of matrix completion in chemical engineer.

The **main** contributions of this chapter are the following:

- we provide empirical analysis which shows that OMIC exhibits better performance and flexibility across the whole spectrum of varying quality of side information;
- by considering a large array of real data, we show that our method surpasses the state-of-the-art in terms of accuracy;
- we demonstrate how our methods give rise to interpretable solutions, for instance, in terms of the amount of user bias, movie quality, and cluster effects in a learned matrix;
- we examined the application of OMIC in the natural sciences for the prediction of activity coefficients in thermodynamics. We are the first to use matrix completion to tackle this task.

Parts of this chapter are based on:

> Antoine Ledent*, **Rodrigo Alves***, and Marius Kloft. Orthogonal inductive matrix completion. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
>
> **Rodrigo Alves***, Antoine Ledent*, Renato Assunção, and Marius Kloft. An empirical study of the discreteness prior in low-rank matrix completion. In *Proceedings of Machine Learning Research*(1), volume 148: Pre-registration in ML, pages 111–125, 2021.
>
> Fabian Jirasek*, **Rodrigo Alves***, Julie Damay*, Robert Vandermeulen, Robert Bamler, Michael Bortz, Stephan Mandt, Marius Kloft, and Hans Hasse. Machine learning in thermodynamics: Prediction of activity coefficients by matrix completion. *The journal of physical chemistry letters*, 11(3):981–985, 2020.

* The authors contributed equally to this research.
(1) Formerly JMLR WCP proceedings.

4.1 Metrics

Let $R \in \mathbb{R}^{m\times n}$ denote the ground truth matrix, $\hat{R}^{(k)}$ the matrix predicted by method k and let $\bar{\Omega}$ be the test set (a subset of entries). Let $\bar{R}^{(k)}$, $\hat{B}_U^{(k)}$ and $\hat{B}_I^{(k)}$ be respectively the zero-order term ($X^{(1)}M^{(1,1)}Y^{(1)}$ in the case of Section 2.1.1), the vector of user biases and the vector of items biases estimated by method k. In the SoftImpute case we need an extra post-processing step to estimate the biases: $\bar{R}^{(SI)} = \sum_{ij} \hat{R}_{ij}^{SI}/mn$, $\hat{B}_{U_i}^{(SI)} = \sum_j (\hat{R}_{ij}^{(SI)} - \bar{R}^{(SI)})/n$ and $\hat{B}_{I_j}^{(SI)} = \sum_i (\hat{R}_{ij}^{(SI)} - \bar{R}^{(SI)})/m$. To assess the methods we used the metrics presented in the list bellow:

- **Root-mean-square Error (RMSE)**: RMSE $= \sqrt{\sum_{i,j\in\bar{\Omega}} (\tilde{R}_{ij} - R_{ij})^2/|\bar{\Omega}|}$
- **Mean Absolute Deviation (MAD)**: MAD $= \sum_{i,j\in\bar{\Omega}} |\tilde{R}_{ij} - R_{ij}|/|\bar{\Omega}|$
- **Matrix Bias Deviation (MBD)**: MBD $= \left|\bar{R} - \bar{R}^{(k)}\right|$
- **User Bias Deviation (UBD)** and **Item Bias Deviation (IBD)**: UBD $= \|B_U - \hat{B}_U^{(k)}\|_{\mathrm{Fr}}$ (resp. IBD $= \|B_I - \hat{B}_I^{(k)}\|_{\mathrm{Fr}}$)
- **Spearman Correlation (SPC)**: in recommender systems the RMSE is not the only important method to assess the quality of prediction. A method can have a smaller **RMSE** than another one but recommend less accurately. SPC $= \rho_S\left(R_{\bar{\Omega}}, \hat{R}_{\bar{\Omega}}^{(k)}\right)$, the Spearman correlation between two vectors composed of the entries of R and $\hat{R}^{(k)}$ on the test set.

To assess the agreement of our clustering method with the ground truth:

- **Random Index (RI)**: Let f_1, f_2 be two partitions of a set $\{1, 2, \ldots, m\}$, the Random index $\mathrm{RI}(f_1, f_2)$ between f_1 and f_2 is defined as the proportion of pairs of elements in $\{1, 2, \ldots, m\}$ which are either placed in the same cluster in both partitions f_1, f_2 or placed in a different cluster in both partitions f_1, f_2:

  $$\mathrm{RI}(f_1, f_2) = \#(\mathcal{S}_{f_1,f_2}) / \binom{n}{2}, \quad \text{where}$$

  $$\mathcal{S}_{f_1,f_2} = (\{i_1, i_2\} : [f_1(i_1) = f_1(i_2) \wedge f_2(i_1) = f_2(i_2)] \vee [f_1(i_1) \neq f_1(i_2) \wedge f_2(i_1) \neq f_2(i_2)]) \, .$$

Remark 4.1.1. *Note that lower values of RMSE, MAD, MBD, UBD and IBD and higher values of SPC and RI correspond to better performance. RI is well defined even if f_1 and f_2 return a different number of clusters.*

4.2 Baselines

Our model is a fundamental tool that relies only on an incomplete matrix and a set of side information. We compare our model with other similarly fundamental models such as IMC [14, 34, 38] and Softimpute [29], with the understanding that the basic ideas could be refined and incorporated into more sophisticated recommender systems.

- **SoftImpute (SI)** is a matrix completion method that uses nuclear-norm regularization. This is a standard baseline for non-inductive matrix completion [29].
- **SoftImpute with bias (B-SI)** is a popular approach that consists in first, training user and item biases, and then training the SoftImpute model on the residuals.
- **Inductive Matrix Completion with Noisy Features (IMCNF)** In this model [38] we train a sum of an IMC term and a residual SoftImpute term jointly (via alternating optimization). This model requires side information and was, therefore, only considered in real data experiments.

In the scenario where no explicit side information is provided for users or items, two branches of clustering frameworks are widely used in collaborative filtering-based recommendation systems: (1) matrix factorization methods and (2) nearest neighbor methods. We select as baselines a state-of-the-art representative example of each branch as follows:

- **Matrix Factorization**: Apply standard nuclear-norm matrix factorization [29] and then cluster the rows (resp. columns) of the recovered matrix to detect communities of users (resp. items).
- **Nearest Neighbors**: Nearest neighbor methods typically calculate a statistical distance between users (resp. items) using only the known entries and then group the users (resp. items) hierarchically. As a representative example, we used the Pearson correlation.

4.3 Synthetic experiments

We use synthetic data to examine the models proposed in Chapter 2 and the algorithms developed in Section 2.3 by considering several ground-truth regimes. To that end, we propose two matrix generation procedures to assess our model's ability (1) to detect the user and item biases (Section 2.1.1) and (2) to recover hidden clusters (Section 2.2).

4.3.1 User and item biases generation procedure

Our first generation procedure can be described as follows:

STEP 1: Let $\tilde{a}$ be the vector with components $\tilde{a}_i = i - \frac{m+1}{2}$ and let $\tilde{b}$ be the vector with components $\tilde{a}_j = i - \frac{n+1}{2}$. Then compute $a = \frac{\tilde{a}}{\|\tilde{a}\|}$ and $b = \frac{\tilde{b}}{\|\tilde{b}\|}$. We also write $v_1 \in \mathbb{R}^m$ for $(\frac{1}{\sqrt{m}}, \frac{1}{\sqrt{m}}, \dots, \frac{1}{\sqrt{m}})^\top$ and $v_2 \in \mathbb{R}^n$ for $(\frac{1}{\sqrt{n}}, \frac{1}{\sqrt{n}}, \dots, \frac{1}{\sqrt{n}})^\top$

STEP 2: Then we define $G = \frac{1}{2} a v_2^\top + \frac{1}{2} v_1 b^\top$ and $S \in \mathbb{R}^{m \times n}$ where $S_{i,j} = (1/mn)$, if $(i,j) \in \{1, \cdots, m/2\} \times \{1, \cdots, n/2\} \cup \{m/2+1, \cdots, m\} \times \{n/2+1, \cdots, n\}$, and $S_{i,j} = -(1/mn)$ otherwise.

STEP 3: Therefore, we can generate a matrix $R \in \mathbb{R}^{m \times n}$ as

$$R(\alpha) = \alpha c G + (1 - \alpha) c S, \tag{4.1}$$

where $\alpha \in [0, 1]$ is a parameter that controls the relative intensities of the user/item biases and the non-inductive component, and c is a scaling constant.

Note that G is composed of the sum of two terms. The first term is a matrix with all rows being equal, whilst the second term's columns are all equal. Thus G is made up of user and item biases. On the other hand, the S matrix can be divided into four blocks of equal sizes. The top left and bottom right blocks entries have a constant value of $(1/mn)$. The remaining block has entries with the value $-(1/mn)$.

4.3.2 Cluster side information generation procedure

We also generated square matrices in $\mathbb{R}^{m\times m}$ where the users and items are naturally divided into k clusters of size m/k. Without loss of generality, the first cluster consists of the first m/k entries, etc., and we assume f, g are defined according to this clustering arrangement. Our data generation procedure basis on the following three matrices:

STEP 1: To represent the cross-community affinities, first generate a *pure community component* matrix A. For that, construct a $k \times k$ matrix $\bar{A}$ with i.i.d. $N(0,1)$ entries. Then set $U_{i,j} = \bar{A}_{f(i),g(j)}$ and set A to be a normalized version of U of Frobenius norm m.

STEP 2: Construct the matrix $\tilde{B}^1 \in \mathbb{R}^{m\times k}$ whose columns are projections of independent Gaussian vectors in $\mathbb{R}^m$ onto the space $\{x \in \mathbb{R}^m : \sum_{i\in f^{-1}(c)} x_i = 0, \forall c \in \{1,2,\ldots,k\}\}$. $\tilde{B}^1$ represents the *User* × *(Item community)* term. Generate similarly $\tilde{B}^2$ for the *Item* × *(User community)* term. Set $U_{i,j} = \bar{B}^1_{i,g(j)} + \bar{B}^2_{f(i),j}$ and let B be a normalised version of U with Frobenius norm m.

STEP 3: In the next step, generate the *community-free behaviour* matrix C. It is simply constructed with i.i.d. $N(0,1)$ entries, then projected to the space $\{X \in \mathbb{R}^{m\times m} : \sum_{i\in f^{-1}(c)} x_{i,j} = 0, \forall c \in \{1,2,\ldots,k\}, j \leqslant m \wedge \sum_{j\in g^{-1}(c)} x_{i,j} = 0, \forall c \in \{1,2,\ldots,k\}, i \leqslant m\}$ and normalised to have unit Frobenius norm.

STEP 4: Using these basis matrices, we can construct matrices of the form:

$$R(\alpha,\beta) := A + \alpha B + \beta C, \tag{4.2}$$

where the parameters α and β regulate the importance of ground truth behaviours associated to A, B and C.

Note that for a given f, g, the matrices A, B, C belong to the (independent) subspaces corresponding, in (2.6), to C, $(M+U)$ and Z respectively. The parameters α and β in (4.2) regulate the importance of the community components and the community-free component.

4.3.3 Experimental setup and results

All the hyperparameter tuning was done through cross-validation. We split the set of observed entries Ω into two randomly sampled subsets: 95% for training and 5% for validation. The range of the parameters was adjusted according to each model's needs.

Recovering user and item biases: To assess our model's ability to detect the importance of user and item biases, we generate matrices according to the procedure described in Section 4.3.1. For that, we needed to select the parameters m, n, c and α. We chose $m = n = c = 100$. The parameter $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$ was empirically selected in such a way that the expected intensity of the biases' component varied. Note that in the extremes of the α interval the generated matrix is just composed of one of the components.

To determine the number of observed entries and the sampling distribution, we considered two extra parameters: the percentage of observed entries p_Ω and a parameter $\gamma \in \mathbb{N}$ that manages the sparsity distribution. Given a fixed p_Ω we randomly selected $\gamma(p_\Omega mn/(\gamma+1))$ entries in the first $m/2$ rows and $(p_\Omega mn/(\gamma+1))$ in the remaining ones. The parameter p_Ω was varied in $\{0.15, 0.30, 0.50\}$ and the parameter γ as varied in the range $\{1, 4\}$ ($\gamma = 1$ indicates uniformly sampled observations).

Note that for SI, we need an extra post-processing step to estimate the biases. In this case, we calculate the matrix bias as the average of the predicted matrix entries. After subtracting the SI matrix bias, we calculated the users and the items bias by averaging the columns and the rows, respectively.

For each combination of α, γ and p_Ω we generated 50 different samples of $R(\alpha)$. Given a sampled matrix, we recovered the unknown entries through OMIC, B-SI and SI. Figure 7.1 summarizes the results of the performed simulations.

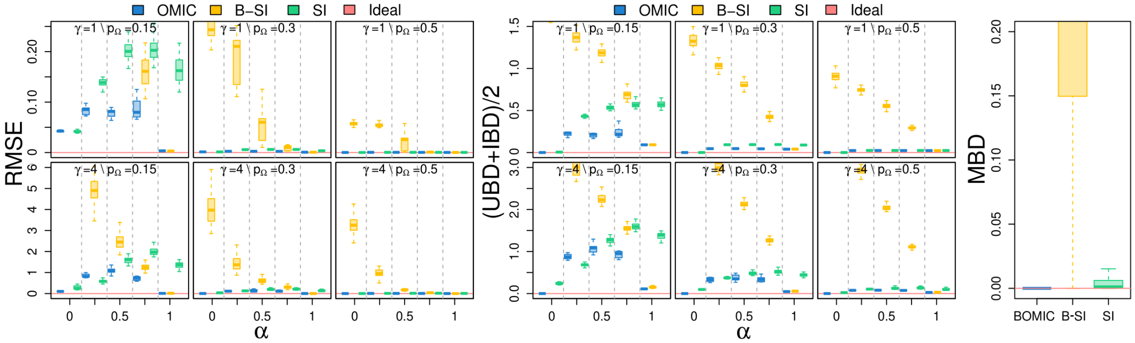


Figure 4.1: Summary of synthetic data simulations results. The first graph shows the relationship between all combinations of the parameters $(\alpha, \gamma, p_\Omega)$ and the RMSE. The second one shows how $(\alpha, \gamma, p_\Omega)$ influences the correct recovery of user and item biases ((UBD+IBD)/2). Each box plot in the first two graphs is obtained from 50 simulations. The third graph displays the distribution of the MBD over all of the simulations.

Remark 4.3.1. *In the presence of user and item biases, we observe that our method consistently outperforms SI and B-SI, in terms of RMSE, UBD and IBD, and performs as well as SI in the MBD case. In addition, OMIC's ability to recover the correct user and item biases (UBD and IBD in Figure 7.1) is particularly marked in the case of non uniformly sampled entries, as might be expected, in line with Corollary 3.2.1.*

Recovering hidden side information: In Section 4.3.2 we proposed a generation procedure. The parameters α and β in (4.2) regulate the importance of the community components and the community-free component. We varied α and β in $\{0, 0.25, 0.50, 0.75, 1\}$. We also varied the percentage of observed entries $p_\Omega \in \{0.15, 0.30\}$. For each each possible combination of α, β and p_Ω, we generated 50 samples of $R(\alpha, \beta) \in \mathbb{R}^{m \times m}$ with users (resp. items) divided into clusters of size m/k, where $m = 100$ and $k = 4$.

Figure 4.2 summarizes the results of the performed simulations with respect to the accuracy of the detected cluster by our method and the baselines. Since the baselines do not provide a clear way to select the number of clusters, we assume that the value is known, that is, $d_1 = d_2 = 4$.

Remark 4.3.2. *We observe that our method consistently outperforms the baselines, especially when the importance of the community-free component is low (β is low).*

We also investigate whether d_1 and d_2 can be determined through *cross validation* as suggested in Section 2.4.1. Figure 4.3 shows the distribution of the detected number of clusters obtained by our method through *cross validation*.

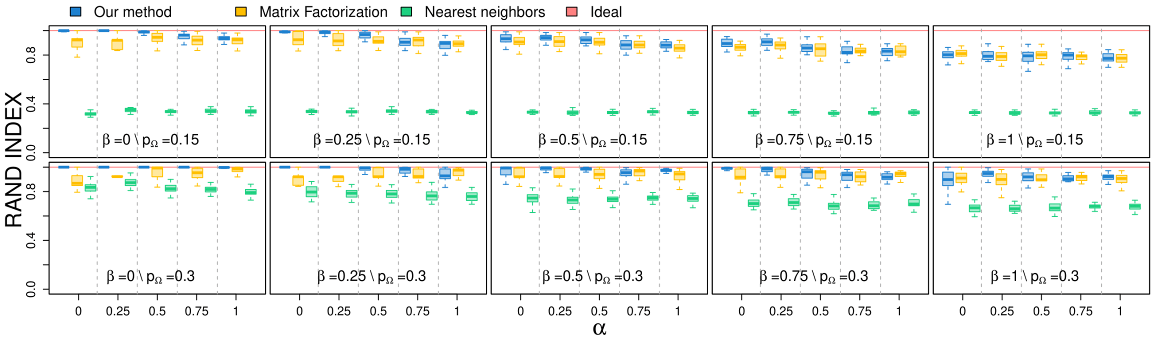


Figure 4.2: Rand index distribution grouped by the parameters *alpha*, β and $\mathbb{P}_\Omega$

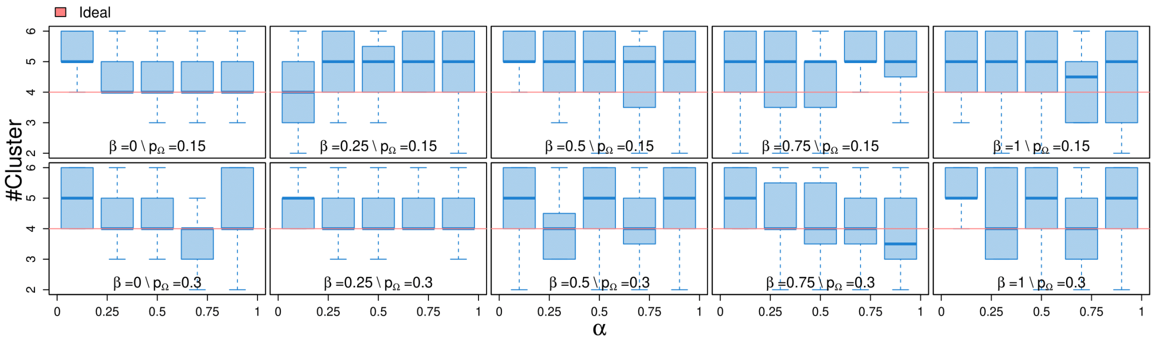


Figure 4.3: The distribution of the number of our model detected clusters according to the parameters α, β and $\mathbb{P}_\Omega$

Remark 4.3.3. *Note that our method can often accurately predict the number of clusters, especially when the number of observed entries is sufficient and the community-free component is small.*

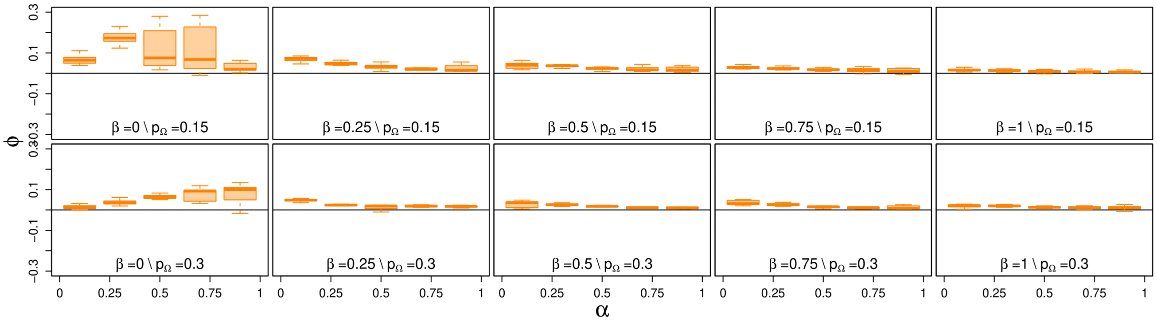


Figure 4.4: ϕ distribution grouped by the parameters α, β and $\mathbb{P}_\Omega$

Finally, we compared the accuracy of our method with the Matrix Factorization baseline[6] that can be seen as single optimization of Problem (2.6) with $f = g =$ null. Let R_{CC} and R_{MF} be the matrices R recovered through our method and Matrix Factorization, respectively. We proposed quantity

$$\phi = (||R_{MF} - R||_F - ||R_{CC} - R||_F)/||R||_F$$

---

[6] Note that Nearest Neighbors does not aim to predict the unknown entries but only recover clusters.

to make the comparison between the methods. Observe that ϕ is normalized by the Frobenius Norm of R, forcing the matrices to be of a similar scale. If ϕ is greater than 0 our method outperforms Matrix Factorization. Figure 4.4 shows, the distribution of ϕ grouped by α, β and P_Ω.

Remark 4.3.4. *We observe that our method consistently outperforms the baseline since it has high values of ϕ (in all scenarios).*

4.4 OMIC as a recommender system

In this section, we present our results on data from the field of recommender systems. We begin by introducing the datasets, then provide our results, and conclude with a practical exploration of the added interpretability benefits of our method.

4.4.1 Datasets

We worked with the following datasets:

- **Amazon** ($R \in \mathbb{R}^{164383 \times 101364}$): Amazon is a multinational technology company which mainly focuses on e-commerce. We used the "Electronics" dataset, which we obtained through [66]. Users are Amazon's clients and items are electronic products (e.g., smartphones). The rating range is from 1 to 5 and the entry (i, j) refers to the rating given by client i to the product j.

- **Douban** [7] ($R \in \mathbb{R}^{4999 \times 4577}$): Douban is a social network where users can produce content related to movies, music, and events. Douban users are members of the social network and Douban items are a subset of popular movies. The rating range is $[1, 5] \in \mathbb{N}$ and the entry (i, j) corresponds the rating of user i to movie j. The author of this dissertation collected the movies' genres from the Douban website.

- **Goodreads spoiler dataset** ($R \in \mathbb{R}^{4199 \times 3278}$): This dataset was released by [67] and it is available online. Goodreads is a social cataloging website that allows individuals to freely search its database of books, annotations, and reviews. In this case, an entry (i, j) represents the rating of the user i for the book j on a scale from 0 to 5. For each user-book pair, in addition to the rating score, the review text is also available. Each sentence of the review was annotated with respect to whether or not spoilers were present. We generated 89 features, such as the length of the review and which

---

[7] Rating matrix available in `https://doi.org/10.7910/DVN/JGH1HA`

percentage of the text contains spoilers. Based on the available side information, we performed clustering of the corresponding features to translate them into community information, which we then used as the X, Y.

- **LastFM** ($R \in \mathbb{R}^{1875 \times 4354}$): Last.fm is a British music website that builds a detailed profile of each user's musical taste based on music recommendations. Differently from the other datasets, an entry (i, j) represents the number of views of user i to band/artist j. We expressed the number of views o a log scale. The website allows users to tag artists, which allows us to group the items (artists) by their associated tags. The corresponding (tag-based) groups were then compared to the clusters obtained by our method.

- **MovieLens**: We consider the MovieLens 1M ($R \in \mathbb{R}^{6040 \times 3706}$) and MovieLens 20M ($R \in \mathbb{R}^{138493 \times 27032}$) datasets, which are broadly used and stable benchmark datasets. MovieLens is a non-commercial website for movie recommendations. Like in Douban, an entry (i, j) represents the rate of user i to movie j on a scale from 1 to 5. In MovieLens 1M, we chose cluster information such as age range for users and movies genres.

We also aim to observe correlations between our recovered clusters and explicitly or implicitly available categorical side information. We noticed that some of this categorical side information was highly unbalanced: some categories have fewer than ten items while others have thousands of items. To properly assess, we keep only items associated with categories that have a significant number of instances on Douban, LastFM and MovieLenz1M datasets.

4.4.2 Collaborative-filtering recommendation

Collaborative-filtering-based methods for recommender systems are one of the most important applications of matrix completion. In this section, we will analyze the behavior of our model as a recommender system as well as benchmark against the baselines in this same task.

Our model is a fundamental tool that relies only on the incomplete matrix and some high-level side information and has the benefit of interpretability. We compare our model with other similarly fundamental models such as SoftImpute (SI) (see [29]) and IMCNF (see [38]), with the understanding that the basic ideas could be refined and incorporated into more sophisticated recommender systems. To make the comparison, we considered the instances presented in sections 2.1.1, 2.1.3 and 2.2 that we respectively call in Table 4.1 by B-OMIC, BC-OMIC and CC-OMIC. Since no systematic side information was provided for Amazon and MovieLenz20M, we only investigated the performance of B-OMIC, CC-OMIC and SoftImpute. For each dataset, we split the set of observed entries (uniformly at random) into a

Table 4.1: Performance comparison of our methods vs baselines on the real datasets

		Amazon	**Douban**	**Google Reads**	**LastFM**	**MovieLenz1M**	**MovieLenz20M**
	p_Ω	0.0001	0.0305	0.0331	0.0051	0.0446	0.0051
RMSE	**B-OMIC**	1.0406	0.8832	1.0736	2.2009	0.8870	**0.7804**
	BC-OMIC	-	**0.8745**	**1.0540**	2.1915	**0.8776**	-
	CC-OMIC	**1.0387**	0.8796	1.0821	**2.1801**	0.8958	0.7893
	SI	1.0625	0.9582	1.0991	2.4109	0.9280	0.8025
	IMCNF	-	0.8825	1.0770	2.1937	0.9192	-
SPC	**B-OMIC**	**0.4121**	0.5292	0.5113	0.5256	**0.6278**	**0.6697**
	BC-OMIC	-	**0.5632**	**0.5120**	0.5351	0.6185	-
	CC-OMIC	0.4113	0.5443	0.5084	**0.5362**	0.6071	0.6423
	SI	0.4119	0.5090	0.4857	0.5040	0.5943	0.6521
	IMCNF	-	0.5345	0.5052	0.5087	0.5998	-
RI	**Users**	-	-	-	-	0.7036	-
	Items	-	0.5548	-	0.7305	0.6080	-

training set (85 %), a validation set (10%) and a test set (5%). Table 4.1 summarizes the results of the real-world data experiments.

Remark 4.4.1. *Observe that OMIC has the lowest RMSE and largest SPC on all datasets.*

We also observed correlations between our recovered clusters and available categorical side information.

Remark 4.4.2. *Note that correlations between our recovered clusters and available sets of categorical side information are high (RI, see Table 4.1), especially the item groups on the LastFM dataset and the users' age range on the MovieLenz dataset.*

It follows that real-world datasets do, in fact, exhibit a non-trivial combination of discrete (community) behaviour and continuous (generic low-rank) behaviour (although the extent to which that is the case is moderate). Furthermore, the groups recovered by our method are interpretable in the sense that they often correlate with available qualitative categorical side information (even though we did not feed this information to the model).

4.4.3 Illustration of OMIC's interpretability

As explained above, one advantage of our method is that it can provide partial explanations for its predictions: each prediction is a sum of terms coming from each of the model's components. Furthermore, this sum is uniquely determined since the components of the model live in mutually orthogonal spaces and correspond to well-defined, distinct intuitive phenomena. For example, if some auxiliary vectors are constructed from user community side information, the algorithm can disentangle the users' particular tastes from those of their respective communities. In particular, our method can discover facts about community-wide behavior.

We illustrate those effects in Figures 4.5 and 4.6. On the left of Figure 4.5, we show the norms of each of the components of the recovered matrix: our recovered matrix takes the form $R = \sum_{k,l\leqslant 3} X^{(k)}\hat{M}^{(k,l)}(Y^{(l)})^\top$ where the $\hat{M}^{(k,l)}$ are obtained as the solution to our optimization problem (2.1), and each component in $X^{(k)}\hat{M}^{(k,l)}(Y^{(l)})^\top$ in the sum corresponds to an interpretable concept. For instance, $X^{(2)}\hat{M}^{(2,1)}(Y^{(1)})^\top$ correspond to user community biases. The norms of each component can give us an idea of how important each component is globally. Thus we see that over the whole GoodReads dataset, the most important components (excluding the global bias) are: (1) the specific match between the user and the book, (2) user generosity, and (3) the quality of each book.

The second picture presents an explanation for an individual prediction. In other words, we chose one entry of R (say, $R_{i,j}$) and represented the corresponding entry of each of the above-mentioned components: for instance, the orange bar to the right of the graph represents the entry $(X^{(3)}\hat{M}^{(3,3)}(Y^{(3)})^\top)_{i,j}$, which corresponds to the same rating. This number represents the part of the rating (i,j) which is attributable to a specific preference of the user for the particular movie (discounting the parts of this preference which are shared by the other members of that user's community or other movies of the same genre).

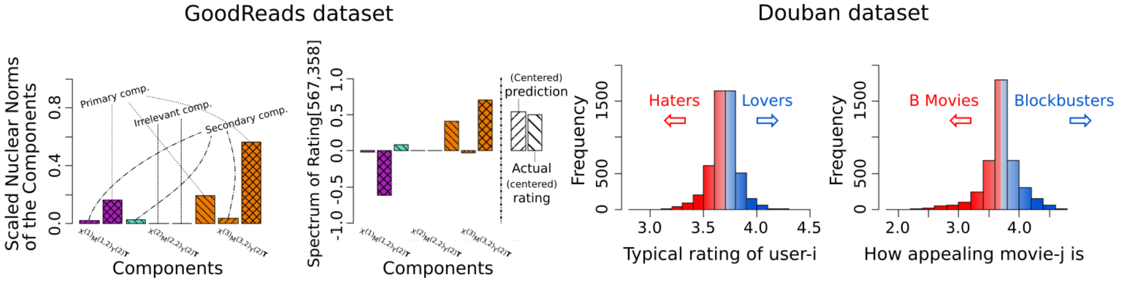


Figure 4.5: The first two graphs show the relative influence of the components on the predictions of the whole matrix and one individual entry, respectively. The last two graphs show the distribution of the users and item biases obtained by the Douban dataset.

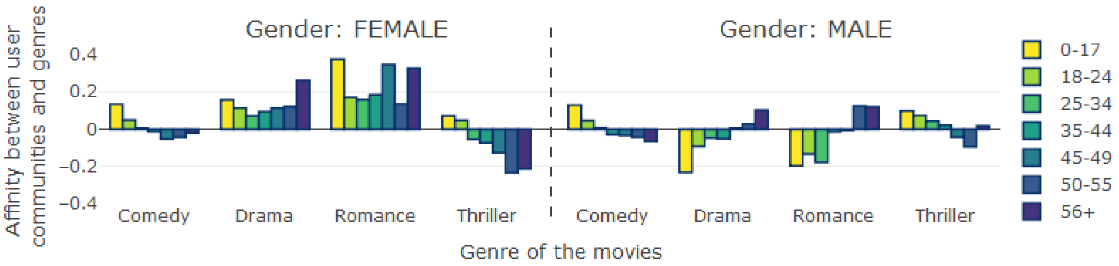


Figure 4.6: Affinity between user communities (grouped by gender and age) and movie genres for MovieLens. These biases can be directly read from the component $X^{(2)}M^{(2,2)}(Y^{(2)})^{\top}$ in the instance of Section 2.1.3.

Thus, in this case, the book is not generally considered good by the users (cf. large negative component corresponding to the purple bar). However, the individual is usually generous (first orange bar), and the specific book and user are a good match for each other (cf. large orange component corresponding to the rightmost bar).

Note that we specifically train our model in a way that treats each of the components as a separate entity, with its own cross-validated hyperparameter. So that the decomposition along those components is more finely tuned and intertwined with the optimization process (rather than collected as a statistic after applying a standard matrix completion method).

The last two graphs show the distribution of user biases and movie quality in the Douban dataset. The distribution is similar to a normal distribution (squished at the boundaries), allowing us to characterize the users (resp. movies) on a spectrum between haters and lovers (resp. B-movies and blockbusters).

Figure 4.6 shows bar charts illustrating the affinities between user communities (gender-age combinations) and four movie genres in the MovieLens dataset. Note that these affinity scores are part of our model and can be directly read from the component $X^{(2)}M^{(2,2)}(Y^{(2)})^{\top}$ (cross-communities component, Section 2.1.3). We observe that OMIC is able to detect noteworthy human behaviour. For instance, female users tend to prefer drama and romance movies while male users appreciate comedies and thrillers instead. One can also notice that the biases vary with users' ages: for instance, older male users like romance movies more than their younger counterparts. Among others, the impact of gender in the general model is stronger for the female gender than for the male gender. It can be noticed by the scale of the graphs.

4.5 Predicting activity coeffients via matrix completion

A key property in chemical engineering is the activity coefficients, which measure the non-ideality of liquid mixtures. Activity coefficients are important for modeling chemical and phase equilibria, but they are also important for transport processes in general. Although experimental data on thousands of binary mixtures are available, prediction methods are still required to calculate the activity coefficients in many relevant mixtures that were not previously investigated.

In this dissertation, we describe a novel application of machine learning to the field of physical chemistry and thermodynamics: the prediction of activity coefficients of binary liquid mixtures by matrix completion. The entry $\{i, j\}$ is the activity coefficient of a mixture of solute i and solvent j. Here, we use the instance of our method presented in Section 2.1.2 by considering the chemical family of the solute/solvent as cluster side information. We achieve comparable results to our previous method [58], with a (much) more efficient algorithm with the benefit of interpretability.

4.5.1 Dataset

We trained and evaluated our matrix completion method using data from the Dortmund Data Bank (DDB) (version 2019). The data consists of activity coefficients at infinite dilution in binary mixtures. We adopted temperatures between 297.15 K and 299.15 K, i.e., at 298.15 ± 1 K. Temperature dependence of activity coefficients over such narrow temperature ranges is typically negligible and, thus, we did not consider it.

The DDB is a private dataset. The application of Section 4.5 was developed in collaboration with the Thermodynamics Group (Lehrstuhl für Thermodynamik) of the TU Kaiserslautern, which has the license to use the dataset. We warmly thank Prof. Hans Hasse and his group for the collaboration on this work.

For several solute-i $\times$ solvent-j mixtures, multiple results on activity coefficients over the specified temperature range are observed. For training and evaluation purposes, we average the coefficients of these mixtures. Additionally, we modified the dataset to take into account only the molecular components. Non-molecular solutes and solvents were omitted from the data set: primarily salts and ionic liquids and metals and components for which no molecular formula was available. Additionally, to evaluate the proposed model's predictions using the leave-one-out method, all solutes and solvents for which we observe only one single mixture coefficient R_{ij} were excluded from the data set. The above conditions were met for a total of 4094 activity coefficients R_{ij} distributed in 240 solutes-i and 250 solvents-j.

We split the compounds into clusters of their chemical families (e.g., alcohols, alkanes, ketones, etc.). Then we constructed the induced matrices according to the procedure described in Section 2.1.2.

4.5.2 Baselines

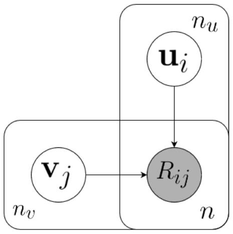
This section considers the following baselines:

- **UNIFAC: (see [68]):** phenomenological model the uses chemical group-contributions as information features. It is a stable state of art baseline;
- **Probabilistic matrix factorization (see [58]):** the model defines a probability distribution over all R_{ij} by specifying a stochastic process that hypothetically generates activity coefficients conditioned on the latent compounds features u_i (resp. v_j) of solute-i (solvent-j).

Figure 4.7 shows the probabilistic graph of our generative model with the following generating procedure:

STEP 1: For each solute i (and each solvent j), draw a latent feature vector u_i (v_j) of dimension K from a multivariate normal distribution with zero mean vector $\mathbf{0}_k$ and standard deviation Σ_u (Σ_v).

STEP 2: Generate each R_{ij} as a Cauchy distribution with scale λ and centered around the inner product of u_i and v_j.



$$\mathbf{u}_i \sim \mathcal{N}(\mathbf{0}_k, \Sigma_u)$$
$$\mathbf{v}_j \sim \mathcal{N}(\mathbf{0}_k, \Sigma_v)$$
$$(R_{ij}|\mathbf{u}_i, \mathbf{v}_j) \sim \text{Cauchy}(\mathbf{u}_i^\top \mathbf{v}_j, \lambda)$$

Figure 4.7: Probabilistic graph of our model. Each activity coefficient in R_{ij} is distributed as a Cauchy distribution with scale λ and centered around the inner product of u_i and v_j. Each solute (solvent) feature vector v_i (u_i) is distributed as a multivariate normal distribution with zero mean vector $\mathbf{0}_k$ and co-variance matrix Σ_u (Σ_v).

Therefore, we have a probabilistic matrix factorization model since our large matrix R is modeled in terms of the product of a (smaller) tall matrix, whose rows are the solute feature vectors u_i, and a narrow matrix, whose columns are the solvent feature vectors v_j. We empirically selected $K = 4$, $\lambda = 0.15$ and $\Sigma_u = \Sigma_v = 0.8 I_K$. Here, $I_K \in \mathbb{R}^{K \times K}$ is the identity matrix.

4.5.3 Experimental results

We applied OMIC to the DDB activity coefficients data and compared our findings to UNIFAC's predictions and our previous probabilistic matrix completion method. For the prediction of each of the 4094 activity coefficients R_{ij}, we fit our model considering only the remaining 4093 coefficients. Table 4.2 presents the results in terms of RMSE and MAD. We compute these metrics in two scenarios: (1) *full data*, by considering all data points described in the previous sections; and (2), *UNIFAC data*, by considering only the data points that UNIFAC can also predict.

Table 4.2: Comparison of our method's results to those of the UNIFAC (state-of-the-art method) **Full data**: all data points described in Section 4.5.1. **UNIFAC data**: only the data points that UNIFAC can also predict.

	Full data		**UNIFAC data**	
	MAD	**RMSE**	**MAD**	**RMSE**
OMIC	**0.3058**	**0.6916**	**0.2352**	**0.5618**
SoftImpute	0.3412	0.7933	0.3354	0.7626
Probabilistic MF	0.3363	0.8246	0.3156	0.7734
UNIFAC	–	–	0.3547	0.7912

Remark 4.5.1. *Our method outperformed the baselines since it produced results that have a lower deviation in terms of RMSE and MAD. Note that the UNIFAC method has been the state-of-the-art method for predicting activity coefficients for more than three decades.*

Remark 4.5.2. *UNIFAC can predict only* 5.73% *of the total number of entries of the matrix* R*. On the other hand, our method is capable of estimating the coefficients of the entire matrix.*

Our method is also explainable. Differently from the baselines, our method induces the learning process with the compounds' families to factorize the prediction of each single binary mixture coefficient as

$$M_{ij} = c_{ij} + u_i + v_j + z_{ij},$$

where c_{ij} is a component related to the combination between the *family* of solute-i and the *family* of solvent-j, u_i represents the combination of the solute-i and the *family* of solvent-j, v_i (respectively) represents the combination of the solvent-j and the *family* of solute-i, and z_{ij} is the special combining ability of solute-i and solvent-j, a component *not explained* by the families of the compounds. Figure 4.8 illustrate the interpretability of the predictor.

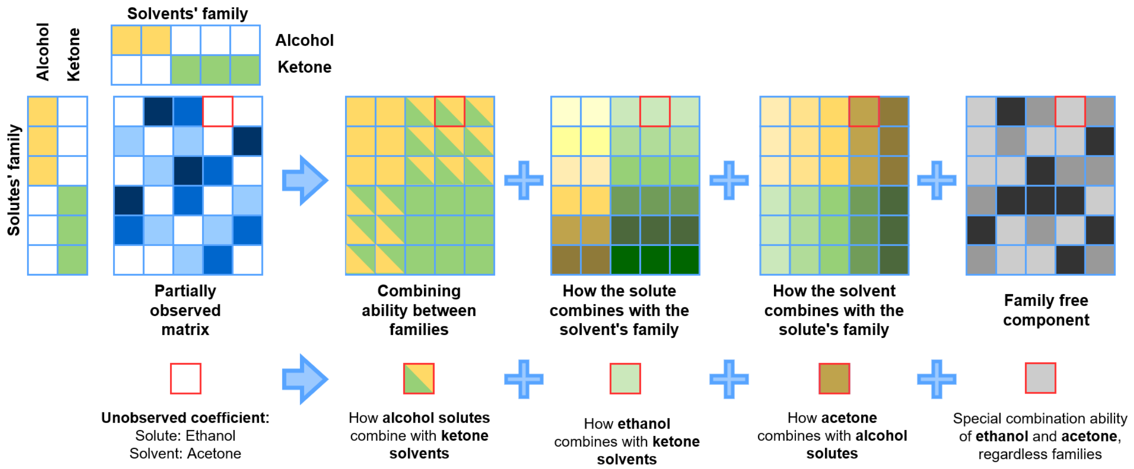

Solvents' family
Alcohol
Ketone
Alcohol
Ketone
Solutes' family
Partially observed matrix
Combining ability between families
How the solute combines with the solvent's family
How the solvent combines with the solute's family
Family free component
Unobserved coefficient:
Solute: Ethanol
Solvent: Acetone
How alcohol solutes combine with ketone solvents
How ethanol combines with ketone solvents
How acetone combines with alcohol solutes
Special combination ability of ethanol and acetone, regardless families

Figure 4.8: Visualization of orthogonal inductive matrix completion for the prediction of activity coefficients. The model is a sum of explainable matrix terms.

4.6 Conclusion

We conducted experiments to validate the approaches described in Chapter 2 under a wide range of regimes. To accomplish it, we presented generation strategies to simulate the environment for which our approaches were designed. Our methods were more accurate than alternative ones at recovering bias and cluster-side information.

Then, we demonstrated how we could apply our methods to recommender systems and chemical engineering. For the first application, we showed that OMIC has SOTA accuracy in recommendation prediction. Additionally, we illustrated how our strategy improves interpretability without requiring any additional post-processing procedure.

Regarding our second application, we show a novel application of machine learning for the field of chemical engineering. We introduced an inductive matrix completion approach for predicting the activity coefficients of liquid mixtures at infinity dilution. Our method outperforms the SOTA method with the added benefit of interpretability.

Chapter 5

Related work and discussion

5.1 Related work on inductive matrix completion

Inductive matrix completion [14, 35–37] is the problem of solving matrix completion with some side information: given some features $X \in \mathbb{R}^{m \times d_1}$ and $Y \in \mathbb{R}^{n \times d_2}$, it tries to find a low-rank matrix M such that $R = XMY^\top$ approximates the observed matrix well. It has found many successful applications in recent years [69–71]. Theoretical guarantees were provided in [72–74]. Note that in the basic model, successful IMC requires that the columns of X (resp. Y) span the left (resp. right) singular vectors of the SVD of the ground truth matrix (this case is often referred to as "perfect" side information). In [38] the extended model $R = XMY^\top + N$, with nuclear-norm regularization applied to both M and N was proposed. Recently, progress was made in the direction of matrix completion with side information with the need to extract features jointly [75]. In this dissertation, we proposed a method that efficiently incorporates user biases, an IMC term, and a pure low-rank term, jointly, in the form of $R = \sum X^{(k)} M Y^{(l)\top}$.

On the application side, [76] model user-item interactions over time using a tensor and exploited users' side information (such as demographics information) to improve accuracy. In the same line, [77] propose to improve the prediction accuracy by exploiting the user's demographic information. [78] developed an algorithm that utilizes item side information for selecting top-N recommender systems. For that, they use auxiliary matrices to induce the learning procedure and recover an aggregation coefficient matrix that is used within an item-based recommendation framework to generate recommendations for the users. Finally, [79] investigate the problem of exploiting heterogeneous side information for recommendations.

5.2 Related work on community discovery

Community discovery is a widely researched task in recommender systems. In [9], the authors propose a probabilistic model to solve binary matrix completion with graph side information based on the assumption that the users form communities. The clusters are recovered from the graph information via the stochastic block model, and the cluster preferences are then recovered from the observed data. Similar approaches can be observed in [59, 80–82]. The main difference between these works and ours is that they do not allow for non-random user-specific behaviour within each cluster (except [59]). That is, there is no difference between predicting the matrix and predicting the clusters. In that respect, our setting is more similar to the regularization-based techniques [83–85]. The paper [59] is, to our knowledge, the only work that incorporates item-specific behaviour in a community detection context. They do so in a discrete fashion with the concept of "atypical" movies and users. Our approach is a continuous one, which includes the possibility of representing any matrix (at a regularization cost). A deep learning approach to extracting community information from graphs is offered by graph neural networks [86, 87].

Another systematic work thatstudies collaborative clustering is [88]. The authors provide a deep theoretical analysis of a model. There, the items must be clusterized based on discrete ratings given by users. Here the ratings are iid for any fixed pair of communities and no specific algorithm is presented. In [89, 90], the authors detect user groups applying k-means on the user-latent factor matrix (imputing the unknown entries via collaborative filtering). Nearest neighbor techniques are also employed in aggregation methods: in [91], the authors use the Pearson correlation to define the similarity among the users while in [92], the cosine similarity is applied. One distinguishing characteristic of our model is that it is able to learn both ratings and communities *jointly*. Although some authors explore orthogonality and factorization to implement clustering in matrices [93–95], their works differ from ours since they start from a fully-known matrix.

5.3 Related work on the theory of matrix completion

A major step signaling the beginning of the construction of a formal theory of matrix completion was the introduction of the SoftImpute algorithm [29], which uses the nuclear norm as a regularizer. Around the same time, the field witnessed a series of breakthroughs in the study of how many entries are required to recover a low-rank matrix exactly [30, 31] or approximately from noisy entries [32, 33]. Those works assume that the entries of the matrix are sampled uniformly. A simpler and more complete approach to the same results was provided in both [96] and [97]. The conclusion of the works on exact recovery is that if the entries are sampled uniformly, it is possible to recover the matrix with high probability assuming $O(\mu rn\log(n)^2)$ entries are observed, where n is the dimension of the matrix, and μ is some notion of *coherence*, which is $O(1)$ if the singular vectors have roughly equal components.

Other works [63, 98–100] have focused on the case of non uniformly sampled entries. The general form of the results obtained is (similarly to the uniform case) that $O(rn\log(n)^2)$

observed entries are sufficient. However, these results come at the cost of making strong explicit assumptions on the distributions, sometimes with relevant constants showing up in the bounds.

The case of non-uniform entries with absolutely no assumption on the sampling distribution is an interesting one that commands a completely different approach. It was studied in [61, 101]. The most related work to ours is [38], where the authors study and provide generalization bounds for a model composed of a sum of an IMC term and a standard SoftImpute model. This model can be seen as a particular case of ours. Note we require to adapt proofs to obtain bounds with a tighter dependence on the dimensions of both left and right side information for the bounds to be non-trivial in case of user biases. Furthermore, no notion of interpretability or orthogonality was presented.

5.4 Related work on prediction of activity coefficients

Binary mixtures are fundamentally important in the field of chemical engineering because they are versatile. In general, the properties of mixtures cannot be described solely on the basis of the properties of their constituent parts. Alternatively, if it is possible to determine the respective properties of the binary constituent *sub-mixtures* of a multi-component mixture, then the properties of the multi-component mixture can often be predicted [102]. Therefore, the knowledge of the properties of binary mixtures is essential for the design and optimization of the vast majority of chemical engineering processes.

Because the experimental determination of physico-chemical properties is time-consuming, it is practically impossible to investigate binary mixtures containing all relevant components. Note that even the largest databases of physico-chemical properties, such as the DDB [103] and the NIST Chemistry WebBook [104], contain only a small fraction of the activity coefficients of relevant mixtures. As a result, predictive methods for physico-chemical properties are required.

Although machine learning approaches to chemical and engineering sciences date back more than 50 years, the true potential of artificial intelligence in these fields has only recently begun to be realized [105]. Recent advances in chemical and material sciences have been summarized, for example, by [106]. and [107]. Machine learning methods have previously been used to predict the physicochemical properties of mixtures, including their activity coefficients. However, most of these approaches are quantitative structure-property relationships techniques that use physical descriptors of the components as input data to characterize the considered mixtures and relate them to the desired property using a machine learning algorithm, such as a neural network. Note that the scope of these approaches is typically quite limited and restricted to a few groups of compounds.

Predicting properties of binary liquid mixtures from first principles is not possible yet, except for *very* simple cases. But there are phenomenological models for this, such as UNIFAC [68, 108] and COSMO-RS [109], which are used for this task. Process simulations often rely on the quality of these predictions and great effort has been taken over the last decades to parameterize these models using the available experimental data.

In contrast to previous approaches, we proposed using inductive matrix completion to predict activity coefficients. Our method also has the advantage of being able to predict the coefficients of compounds whose features vectors (e.g. chemical properties) are unknown. Additionally, we got interpretable solutions by inducing the learning mechanism with knowledge of the compounds' chemical families. Such characteristics are regarded as critical for the development of machine learning techniques capable of explaining physical and chemical events [105, 110].

Part Two:
Context-free recommendation

Chapter 6

Methods, optimization and algorithms

The second part of the dissertation begins with this chapter. We turn our focus to the development of a method that investigates an environment in which there is not just a lack of knowledge about user preferences and behavior, but also of *any* usable side information. Without the ability to profile users and items densely, a RS can rely only on recent user-item interaction [15, 16]. A typical example is when a *new* user visits a news website: a RS must select an item solely based on user-independent data (e.g., the average click-through rate) and then monitor whether the new user clicks on the recommended content. The website's goal is to capture the attention of users and increase the overall number of clicks. Providing effective recommendations in this situation involves determining the most popular 'trending' items among the website's readership.

We mimic this challenging environment by using a MAB, the classic reinforcement learning problem. At each trial (new user), the gambler (RS) selects an arm (e.g., news article) to pull (show to the user) and observes a reward (a click or lack thereof). Throughout the event history, an algorithm improves the policy to maximize the reward (e.g., the number of clicks). The standard MAB setting assumes that the (unknown) item popularity distribution is stationary [18–20]. This assumption implies there exists a most popular item fixed over time.

However, one might note that the traditional MAB is in most circumstances ineffective. For recommender systems, assuming that the reward distribution is stationary is highly unrealistic. The items' popularity *may* change over time [16, 26]. Therefore we model the problem as a *non-stationary* multi-armed bandit.

We can find in literature a variety of prior works on context-free non-stationary MAB [21–25]. By treating the MAB problem as a discrete-time one, they don't actively exploit continuous temporal dynamics in the reward environment. To address this flaw, we proposed BMAB, a non-stationary and context-free MAB problem, and a novel cluster-induced algorithm to solve it. The algorithm's core consists of two consecutive stages: first, BMAB groups the user-activity based on the system's state; and second, it induces the exploration and exploitation procedures based on user-activity categories.

Motivated by a phenomenon entitled *audience curiosity* (as illustrated in Section 1.2.2), the grouping procedure is base on existence and interaction of two types of audiences: the *loyal* audience and the *curious* audience [26, 42, 43]. The loyal audience is constituted by fans who assiduously follow the topic. In contrast, the curious audience only turned their attention to the topic due to an extraordinary event.

The **main** contributions of this chapter are the following:

- we proposed a non-stationary and context-free MAB problem in which the temporal dynamics of the recommender audience influences the reward distribution;
- we developed the Burst-induced Multi-Armed Bandit (BMAB) algorithm, a cluster-induced approach for solving the problem stated in the previous item;
- we prove regret guarantees for the BMAB algorithm when the states are recoverable and bursts are separable. We also experimentally analyze the proposed regret bounds;
- we proposed a model which is able to disentangle the slowly-varying regular activity of the *loyal audience* from the *curious* activity occurring in bursts. This model does not depend on hard-to-get external information but uses only a random series of events RSE. Then we developed a novel EM approach to cope with our intensity function's complex dependence on the history of the process.

Parts of this chapter are based on:

Rodrigo Alves, Antoine Ledent, and Marius Kloft. Burst Induced Multi-armed Bandit for Learning Recommendation. *Proceedings of the 15th ACM RecSys Recommender System Conference*, 2021.

Rodrigo Alves*, Antoine Ledent*, Renato Assunção, Pedro Vaz de Melo, and Marius Kloft. Are you here to stay? Disentangling the loyal audience from the curious on social media. **Submitted to** *ACM Transactions on Knowledge Discovery from Data (TKDD)*, Aug 2021.

Rodrigo Alves, Renato Assunção, and Pedro Vaz de Melo. Burstiness scale: A parsimonious model for characterizing random series of events. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1405–1414, 2016.

* The authors contributed equally to this research.

6.1 Description of the model and problem formulation

Basic notation: let $\mathcal{K} = \{1, 2, \cdots, K\}$ be a set of K arms and $\mathcal{T} = \{t_1, t_2, \cdots, t_N\}$ denote a sequence of N timestamps in the interval $(0, T]$. At each time t_i, a gambler chooses one of the K arms and observes the reward $r_i \in \{0, 1\}$. The reward distribution at time t depends on the state $s(t)$ of the system. We set $s(t) = 0$, if t occurs during the *loyal* state, and $s(t) = 1$, if t occurs during the *curious* state. We call by user-activity category $s_i = s(t_i)$, the category of the recommendation request that happens at time t_i.

6.1.1 Temporal dynamics assumptions

We assume that the time series $\mathcal{T}$ is generated by a mixture of two stochastic point processes: (1) a HPP with intensity[8] $\lambda(t) = \lambda_L$, and (2) a Piece-Wise Homogeneous Poisson process (PW-HPP) with intensity $\lambda_C(t)$. We assume that the intensity $\lambda_C(t)$ of the second process is piecewise constant, with the transitions occurring at the random and *unobserved* timestamps $M = \{m_1, m_2, \cdots, m_n\}$ (by convention we also set $m_0 = 0$), on whose distribution we make no formal assumption[9]. Thus, $\lambda_C(t)$ can assume $(n+1)$ values in the interval $(0, T]$, denoted by $\{c_0, c_1, c_2, \cdots, c_n\}$.We write c_j for the (unique) value the intensity $\lambda_C(t)$ takes in the interval $[m_j, m_{j+1})$. A key assumption of our work is that the underlying distribution has the property that $c_j \ll \lambda_L$ (w.p. 1), if $j \mod 2 \equiv 0$, and $c_j \gg \lambda_L$, otherwise. This implies that the PW-HPP alternates between silent mode (very low intensity) and bursty mode (very high intensity). Finally, write $B(t) = \sum_{j:m_j<t}(j \mod 2)$ for the number of PW-HPP transitions into the bursty mode which occurred before t (thus, if $s(t) = 1$, t belongs to the $B(t)^{\text{th}}$ burst).

The first graph of Figure 6.1 presents a realization of a mixture of two stochastic point processes with the properties described above. In this example, $\lambda_L = 3$, $\lambda_C(t)$ alternates between 0.15 and 15, $T = 100$ and the elements of M (vertical red lines) were randomly selected aiming for the expected number of events of both processes to be the same (cf. details in Section 7.1.2). The HPP models the loyal audience (second graph, green curve, stable throughout the observed interval) while the PW-HPP models the curious audience (last graph, yellow curve, unstable).

At this point, it is necessary to distinguish between three different and correlated concepts. The **first concept** is the *label of the audience* that event t_i correspond to. Even in a given stable or bursty period, the label of the audience is not observed. Thus, it is unknown whether $t_i \in$ HPP (loyal *audience*, second graph of Figure 6.1) or $t_i \in$ PW-HPP(curious *audience*, third graph of Figure 6.1). Our **second concept** is the *system (slot machine) state* $s(t)$. Our underlying assumption is that the state $s(t)$ of the slot machine at time t

---

[8] Intensity function is formally defined in Section 6.3.2

[9] Other than the fact that the total number of transitions n is finite with probability 1.

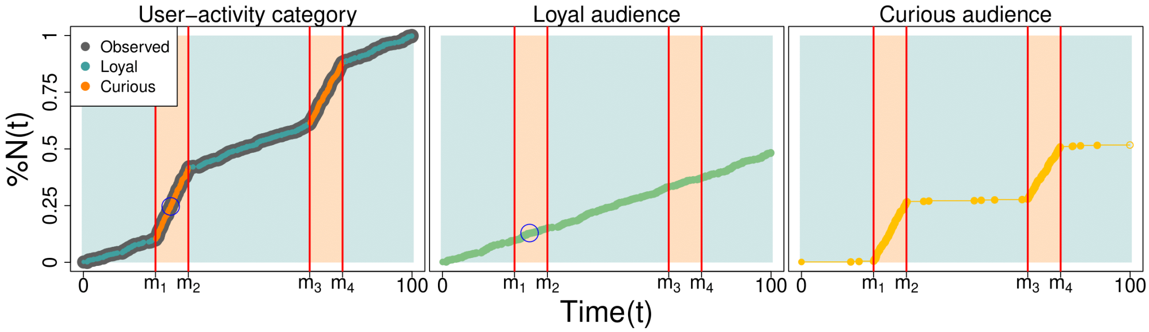


Figure 6.1: Difference between audiences, system's states and user-activity category. **First graph**: jointly activity of the loyal and curious audiences (gray curve, mixture of a HPP and a PW-HPP). The actual user-activity categories are shown by the dots on the first graph: blue for loyal *category*, and orange for the curious *category*. The jointly activity is disentangled in the following two graphs. **Second graph**: HPP (green curve), models the loyal *audience*. **Third graph**: PW-HPP (yellow curve), models the curious *audience*. In all graphs, the vertical lines indicate the actual state transitions (i.e., the set M) and the colored stripes at the background represents the system's states: blue for loyal *state*, and orange for the curious *state*.

depends on the current dominating audience dynamic: $s(t) = 0$ in the calm periods (loyal *state*), and $s(t) = 1$ in the bursty periods (curious *state*). Thus, $s(t)$ depends of the value of the intensities of HPP and PW-HPP at time t, and it is defined for the whole observed interval $[0, T)$. In Figure 6.1, the system's states are represented by the colored stripes at the background: blue for loyal *state*, and orange for the curious *state*. Observe that the event t_{150} (blue circle, second graph) belongs to the loyal *audience* while the slot machine is set to the curious *state* at same time $t = t_{150}$.

More precisely, $s(t)$ is determined by

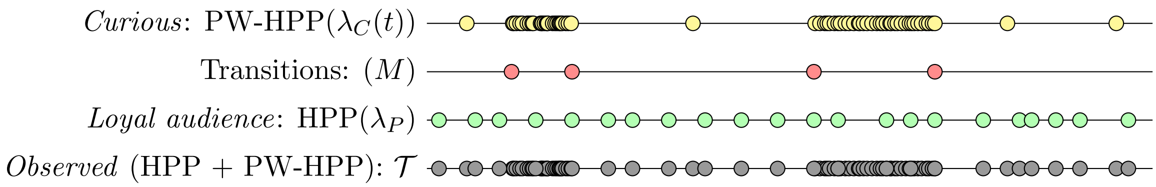
$$s(t) \equiv \max(i | m_i \leqslant s(t)) \mod 2. \tag{6.1}$$

Finally, the **third concept** is the *user-activity category* (s_i). Each event t_i corresponds to activity of an user (request of a recommendation). For a given user-activity, we have $s_i = s(t_i)$. Therefore, s_i is just defined in the events' timestamps (left graph of Figure 6.1, blue and orange dots). We categorize an *user-activity* as loyal, if $s_i = s(t_i) = 0$, and curious otherwise.

The timeline of events is another way to illustrate our temporal assumptions. Figure 6.2 depicts the main concept of the distribution of events in timelines. We observe the point process N timestamps of events (set $\mathcal{T}$) up to a time t_N (grey dots in the fourth row). Furthermore, we assume that these events are a mixture of events coming from the *loyal* *audience* and the *curious*, two independent point processes, shown yellow and green in the first and third rows, respectively. We model the *curious* generating the occasional bursts as a PW-HPP. The events associated with the *loyal audience* are modeled by a second process, a HPP, shown in the third row.

A third underlying process, about which we make no assumptions, controls the times when the *curious audience* (or PW-HPP) transitions occur. These transitions are shown as red dots in the second row in Figure 6.2. The main difficulty with this model is that we only observe the gray dots in the fourth row. The labels associated with each event (the green and yellow colors) and the transitions (the red dots) are not directly observed.

Figure 6.2: A mixture of point processes in timelines. The *curious* (PW-HPP) and *loal audience* (HPP) labels, as well as the transitions (events of M), are not observed.


Curious: PW-HPP($\lambda_C(t)$)
Transitions: (M)
Loyal audience: HPP(λ_P)
Observed (HPP + PW-HPP): $\mathcal{T}$

6.1.2 Reward distribution assumptions

In our model, the reward distribution is a consequence of the audience variation: we assume that reward distribution is stationary in the absence of bursts. Hence, in all pieces of the interval where $s(t) = 0$ we model the distribution of arm a's reward at time t as $r(a,t) \sim \text{Bernoulli}(\theta^0_a)$. In contrast, in the presence of bursts ($s(t) = 1$) the reward distribution varies: each burst has its own stationary reward distribution. Therefore, we model $r(a,t) \sim \text{Bernoulli}(\theta^{B(t)}_a)$ when $s(t) = 1$.

At each time t_i, the agent must choose an arm according to a policy $\pi(t|s(t), B(t))$. Let $\bar{\theta}^t_a = \mathbb{E}[r(a,t)]$ be the expected reward for arm a at time t given $s(t)$ and $B(t)$. As mentioned before, a common goal is to maximize the expected reward $R(T)$ over the entire horizon, which can be written as

$$\mathbb{E}\big(R(T)\big) = \mathbb{E}\sum_{i=1}^{N} \bar{\theta}^{t_i}_{\pi(t_i|s(t_i),B(t_i))},$$

where the expectation runs over both (1) the random choices made by the algorithm policy and (2) the reward distribution. Note that we do not let the expectation run over the distribution of the timestamps. Thus the left-hand side is technically a random variable. We also adapt the notion of (expected) regret, which relies on the concept of the optimal *fixed* arm. Let $\bar{\theta}_*^t = \max_j \bar{\theta}_j^t$ be the expected reward of the optimal arm at time t given $s(t)$ and $B(t)$, the expected regret is then defined as:

$$\mathbb{E}\big(\mathcal{R}(T)\big) = \mathbb{E}\sum_{i=1}^{N} \big[\bar{\theta}_*^{t_i} - \bar{\theta}^{t_i}_{\pi(t_i|s(t_i),B(t_i))}\big], \tag{6.2}$$

where the expectation is also over the random choices made by the algorithm policy and the random rewards.

6.2 Burst-induced multi-armed bandit

To solve the problem formulated in Section 6.1 we propose the BMAB algorithm. In this section, we will describe our algorithm according to the following structure: first, we will present the main ideas of the Thompson sampling algorithm, which forms the backbone of our method; second, we will describe the steps of BMAB and show its regret guarantees; finally, we will present a slot-machine state detector, which we use to group the user-activities.

6.2.1 Thompson sampling algorithm

The Thompson sampling (TS) algorithm is a classical stochastic approach to solve the MAB problem. In this setting, at time t, the slot machine has K arms and, when an arm a is played, the machine produces a reward $r(a)$. The reward distribution of arm a is a Bernoulli distribution with fixed and unknown parameter θ_a. In summary, an arm a has probability θ_a of returning 1 as a reward, and $1 - \theta_a$ of returning 0.

Also known as *posterior sampling*, TS assumes an independent prior belief over each θ_a. In this Bernoulli reward case, it is natural to choose a beta-distribution as a prior (since it is a conjugate prior). Thus for the MAB case, for each arm a, the prior probability density function of θ_a is beta-distributed with parameters α_a and β_a:

$$p(\theta_a) = \frac{\Gamma(\alpha_a + \beta_a)}{\Gamma(\alpha_a)\Gamma(\beta_a)} (\theta_a)^{\alpha_a - 1}(1 - \theta_a)^{\beta_a - 1},$$

where Γ is the gamma function. At each time t, the TS algorithm samples a vector $\Theta = \{\hat{\theta}_1, \hat{\theta}_2, \cdots, \hat{\theta}_K\}$, where $\hat{\theta}_i \sim \text{Beta}(\alpha_i, \beta_i)$ (i.i.d). Then the policy selects the arm $\pi(t) = \text{argmax}_i \hat{\theta}_i$.

Remark 6.2.1. *The exploration procedure is probabilistic tackled. At each step, the probability density function $f(\theta)$ of Beta(α, β) is greater than zero in the whole domain* $[0, 1]$*. Thus any arm has non-zero probability of being selected.*

Remark 6.2.2. *In the special case when $\alpha = \beta = 1$, $\theta \sim$ Uniform*$(0, 1)$.

Due the conjugacy properties of the beta distribution, the Bayesian update of the parameters α and β is particularly simple: at time t, after the algorithm selects arm $\pi(t)$ and observes the reward $r(\pi(t))$, the parameters of the prior distribution of $\theta_{\pi(t)}$ can be updated as follows:

$$[\alpha_{\pi(t)}, \beta_{\pi(t)}] = [\alpha_{\pi(t)} + r(\pi(t)), \beta_{\pi(t)} + (1 - r(\pi(t)))]. \tag{6.3}$$

6.2.2 The BMAB algorithm

The BMAB algorithm is described in Algorithm 6. The core idea is to use a different instance of the Thompson sampling algorithm on each stationary region with a separate count of α and β for each reward distribution. Then, for a given state, the reward distribution of an arm a is a Bernoulli(θ_a) with prior $\theta_a \sim$Beta(α, β). At time t, the vectors $\alpha_0 \in \mathbb{R}^K$ and $\beta_0 \in \mathbb{R}^K$ (resp. $\alpha_1 \in \mathbb{R}^K$ and $\beta_1 \in \mathbb{R}^K$) denote the parameters of the priors related to the K arms in the loyal (resp. curious) state. Each arm a has reward distribution $r(a) \sim$Bernoulli(θ_a). In the loyal state, the estimated distribution of θ_a is $\theta_a \sim$Beta$(\alpha_0[a], \beta_0[a])$, whereas in the curious state, we have $\theta_a \sim$Beta$(\alpha_1[a], \beta_1[a])$ instead.

To support our explanation we will illustrate our algorithm execution for the time series displayed in Figure 6.1. We assume that $K = 3$ and $[\theta_1, \theta_2, \theta_3] = [0.3, 0.4, 0.5]$ in the loyal state. Regarding the curious state, the parameters assume the values $[\theta_1, \theta_2, \theta_3] = [0.3, 0.9, 0.5]$ (in $(m_1, m_2]$) and $[\theta_1, \theta_2, \theta_3] = [0.9, 0.4, 0.5]$ (in $(m_3, m_4]$). Thus in this case, the best arm during the entire loyal state is arm 3, whilst in the first burst of events it is arm 2, and during the second burst of events it is arm 1. Figure 6.3 shows the prior distributions of θ_1, θ_2, and θ_3 at the times $\{0, m_1, m_2, m_3, m_4, t_N = 100\}$. The first row of graphs corresponds to the priors of the loyal state while the second row of graphs corresponds to the priors of the curious state.

Algorithm 6 BMAB

INPUT: Number of arms $K \in \{2, 3, 4, \cdots\}$, set of events' timestamps $\mathcal{T} = \{t_1, t_2, \cdots, t_N\}$ and forgetting-rate $\gamma \in [0, 1]$

```
$\alpha_0 = \beta_0 = \alpha_1 = \beta_1 = \{1\}^K$
for $i \in \{1, 2, \cdots, N\}$ do
    $s_i = \omega(t_i)$
    $\forall a \in \{1, 2, \cdots, K\}$ sample $\hat{\theta}_a \sim \text{Beta}(\alpha_{s_i}[a], \beta_{s_i}[a])$
    $a^* = \text{argmax}_a \hat{\theta}_a$
    Observe the reward $r(a^*)$
    $(\alpha_{s_i}[a^*], \beta_{s_i}[a^*]) = (\alpha_{s_i}[a^*] + r(a^*), \beta_{s_i}[a^*] + (1 - r(a^*)))$
    if $s_i == 0$ then
        $\alpha_1 = \gamma\alpha_1$ , $\beta_1 = \gamma\beta_1$
        $\forall a \in \{1, 2, \cdots, K\}$ if $\alpha_1[a] < 1$ make $\alpha_1[a] = 1$
        $\forall a \in \{1, 2, \cdots, K\}$ if $\beta_1[a] < 1$ make $\beta_1[a] = 1$
    end if
end for
```

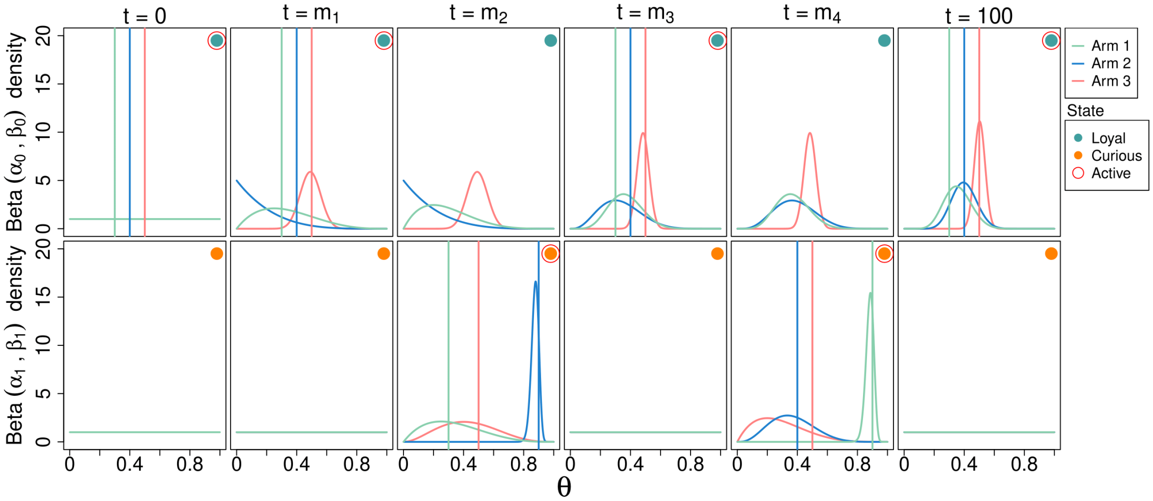


Figure 6.3: Realization of the BMAB algorithm in the mixture of point processes presented in Figure 6.1 ($K = 3$; Loyal state: $[\theta_1, \theta_2, \theta_3] = [0.3, 0.4, 0.5]$; and Curious state: $[\theta_1, \theta_2, \theta_3] = [0.3, 0.9, 0.5]$ (in $(m_1, m_2]$) and $[\theta_1, \theta_2, \theta_3] = [0.9, 0.4, 0.5]$ (in $(m_3, m_4]$)). **First row of graphs**: prior distribution of θ_1, θ_2, and θ_3 at the times $\{0, m_1, m_2, m_3, m_4, t_N = 100\}$ concerning the loyal state. **Second row of graphs**: prior distribution of θ_1, θ_2, and θ_3 at the times $\{0, m_1, m_2, m_3, m_4, t_N = 100\}$ concerning the curious state.

Our precise algorithm can be split into three main steps:

STEP 1 – Initialization [Line 1]: we initialize all entries of the vectors α and β as 1. Thus, priors are initialised as uniform distributions (at $t = 0$).

STEP 2 – Recommendation and learning procedures [Lines 3-7]: in order to maximize the reward, BMAB aims to learn (by updating its priors of) the piece-wise reward distributions with enough confidence to select the best arm at the event time. Therefore, at each event t_i the algorithm needs to detect the category of the user-activity $s_i = s(t_i)$. We assume that an oracle ω is available to provide an estimate of the system's state at each timestamp in $\{t_1, t_2, \cdots, t_i\}$ (line 3). When a *perfect* oracle (i.e. with $\omega(t) = s(t)$) is available, we refer to our algorithm as [**BMAB-O**]. In practice, the role of the oracle can be assumed by our realistic state detector from Section 6.2.4. We refer to the resulting instance of our algorithm as [**BMAB-R**]. For each of the user-activity categories, 0 for loyal and 1 for curious, we model a different instance of TS. Therefore, in the next step (line 4), we sample $\hat{\theta}_a$ (for each a) according to the current prior distribution $\text{Beta}(\alpha_{s_i}[a], \beta_{s_i}[a])$ corresponding to the current user-activity category s_i and the arm a. Our policy is to select (recommend) the arm a^* which has the highest $\hat{\theta}_a$ (line 5). Finally, we observe the reward $r(a^*)$ of the selected arm and update the priors of the TS related to the user-activity category s_i (in accordance with (6.3)).

Define m_i^n as the nth element of the ordered set $\{t_j | t_j > m_i\}$, so that m_2^5 is the fifth timestamp in the stationary section $(m_2, t_n]$.

STEP 3 – Burst separation [Lines 8-12]: we note that the loyal state is associated with a single stationary reward distribution: our method's estimate of the distribution corresponding to the loyal state keeps improving throughout the whole event horizon (first row of graphs, Figure 6.3). In the problem definition, we further assume that each bursty period comes with its own reward distribution. Accordingly, we aim to treat each bursty period as a separate MAB problem, resetting the Thompson priors at the beginning of each burst whilst keeping a global count for the periods where the loyal audience dominates. However, due to the uncertainty inherent in the state prediction method ($\omega(t)$), we engineer a *soft* transition procedure: whenever a burst appears to be ending, the priors corresponding to the burst are gradually forgotten rather than discarded immediately. In Figure 6.3, observe that from $t = 0$ to $t = m_1$ the loyal priors have significantly changed after some reward observations while the curious priors stay as initialized. During the first bursty period $(m_1, m_2]$, the curious priors changed as the model learned the reward distribution of the burst. One can observe at time m_2 that loyal priors remain the same as those at m_1, since the bursty dynamic governs the interval $(m_1, m_2]$. This fact can also be seen at the second bursty period $((m_3, m_4])$. Similarly, when the bursty period appears to taper off, the learned priors are gradually forgotten as the model gains confidence in its observation of a return to normality. To accomplish it, BMAB employs a forgetting-rate $\gamma \in [0, 1]$: at each step with $s_i = 0$, we compute $\alpha_1 = \gamma\alpha_1$ and $\beta_1 = \gamma\beta_1$. If some entry of α_1 or β_1 is less than 1, we round it to 1 (lines 8-12). The effects of this forgetting procedure can be observed in Figure 6.4. At time $t = m_2^5$, i.e., five loyal user-activities after m_2, an increase of the variance around the expected value of θs is seen for the curious state. By observing $t = m_2^{10}$, $t = m_2^{15}$ and $t = m_3$, we note that our smooth forgetting procedure eventually leads to a return to a uniform prior after sufficiently many loyal-state timestamps.

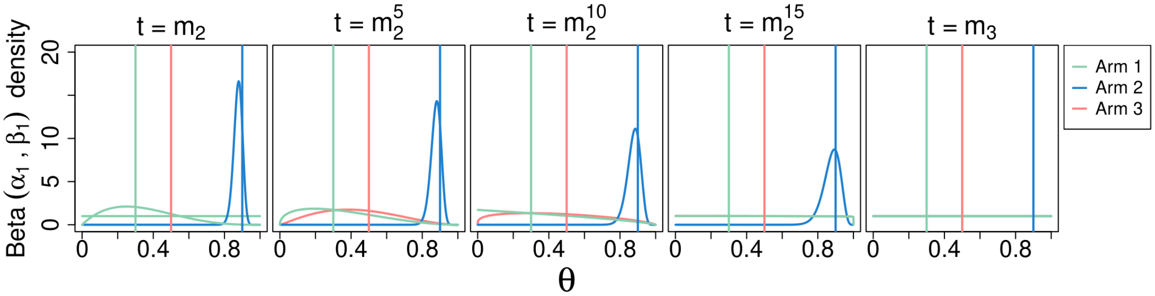


Figure 6.4: Forgetting procedure of the BMAB algorithm: in the interval $(m_2, m_3]$ (loyal state active) of the the mixture of point processes presented in Figure 6.1 ($K = 3$; Curious state: $[\theta_1, \theta_2, \theta_3] = [0.3, 0.9, 0.5]$ (in $(m_1, m_2]$)). Prior distribution of θ_1, θ_2, and θ_3 at the times $\{m_2, m_2^5, m_2^{10}, m_2^{15}, m_3\}$ concerning the curious state.

6.2.3 BMAB regret guarantees

In this section, we present regret guarantees for the **[BMAB-O]** algorithm (with a perfect oracle). For completeness reasons, let us recall the following theorem from [111]:

Theorem 6.2.1. *For the K-armed stochastic bandit problem, TS using Beta priors has expected regret*

$$\mathbb{E}\big(\mathcal{R}(T)\big) \leqslant O\Big(\sqrt{KN \log N}\Big),$$

where N is the total number of events and the big-Oh notation hides only absolute constants.

Now we will prove the regret guarantees for the **[BMAB-O]** that are stated in the following theorem:

Theorem 6.2.2. *Write n_b for the total number of bursts and let the set $\mathcal{N} = \{\mathcal{N}_0, \mathcal{N}_1, \cdots, \mathcal{N}_{B(t_N)}\}$ contain the number of timestamps in each period (with $\mathcal{N}_0$ corresponding to the entire calm period and $\mathcal{N}_i$ corresponding to the ith burst: $\mathcal{N}_0 = \sum_j 1_{s(t_j)=0}$, and for all $i > 0$, $\mathcal{N}_i = \sum_j 1_{s(t_j)=1 \wedge B(t_j)=i}$). We assume access to an optimal oracle ω with $\omega(t) = s(t)$ for all t and set $\gamma = 0$. For all configurations $n_b, \mathcal{N}$ satisfying the (mild) condition $\sum_j 1_{s(t_j)=0 \wedge m_i < t_j < m_{i+1}} > 0$ for all $i \bmod 2 = 0$, we have*

$$\mathbb{E}\big(\mathcal{R}(T)\big) \leqslant O\Big(\sqrt{n_b KN \log K}\Big),$$

where $N = \sum_{i=0}^{n_b} \mathcal{N}_i$ is the total number of events.

Proof. From (6.2) we have:

$$\begin{aligned}
\mathcal{R}(T) &= \sum_{i=1}^{N} \big[\bar{\theta}_*^{t_i} - \bar{\theta}_{\pi(t_i|s(t_i),B(t_i))}^{t_i}\big] \\
&= \sum_{i\in\Omega_0} \big[\bar{\theta}_*^{t_i} - \bar{\theta}_{\pi(t_i|0,-)}^{t_i}\big] + \sum_{l=1}^{n_b} \sum_{j\in\Omega_1(l)} \big[\bar{\theta}_*^{t_j} - \bar{\theta}_{\pi(t_j|1,l)}^{t_j}\big] \\
&= \mathcal{R}_0(T) + \sum_{l=1}^{n_b} \mathcal{R}_l(T), \qquad (6.4)
\end{aligned}$$

where $\Omega_0 = \{a | s(t_a) = 0\}$, $\Omega_1(l) = \{b | s(t_b) = 1 \text{ and } B(t_b) = l\}$, $\mathcal{R}_0(T)$ is the loyal-state regret and for $l > 0$, $\mathcal{R}_l(T)$ is the component of the regret corresponding to the ith burst. The condition $\sum_j 1_{s(t_j)=0 \wedge m_i < t_j < m_{i+1}} > 0$ for all $i \bmod 2 = 0$ guarantees that the bursts are separable in the sense that $B(t) = \#\{j : \omega(t_j) = 0 \wedge \omega(t_{j+1}) = 1 \wedge t_{j+1} \leqslant t\}$ can be computed by the oracle. Therefore, we essentially have $(n_b + 1)$ stationary Thompson sampling algorithms.

Accordingly, applying Theorem 6.2.1 to equation (6.4) we obtain:

$$\begin{aligned}
\mathbb{E}[\mathcal{R}(T)] &= \mathbb{E}[\mathcal{R}_0(T)] + \sum_{l=1}^{n_b} \mathbb{E}[\mathcal{R}_l(T)] \\
&= \sum_{i=0}^{n_b} \mathbb{E}[\mathcal{R}_i(T)] \\
&= O\left(\sum_{i=0}^{n_b} \sqrt{K \mathcal{N}_i \log K} \right) \\
&\leqslant O\left(\sqrt{(n_b + 1) \sum_{l=0}^{n_b} K \mathcal{N}_l \log K} \right) \\
&= O\left(\sqrt{n_b K \log K \sum_{l=0}^{n_b} \mathcal{N}_l} \right) \\
&= O\left(\sqrt{n_b K N \log K} \right), \qquad (6.5)
\end{aligned}$$

where at the fourth line, we have used Jensen's inequality (more precisely, $\|x\|_1 \leqslant \sqrt{d}\|x\|_2$ for all $x \in \mathbb{R}^d$). The theorem follows. ☐

Remark 6.2.3. *As expected, we observe that stable systems, where bursts are rare, expect to have lower regret, as the number of bursts influences the regret bound by a factor of* $\sqrt{n_b}$.

To empirically verify our regret guarantees, we performed 1000 simulations. For more details on the generation procedure, see Section 7.1.2. We explore the following parameter values: $\lambda_L = 1$, $\mathbb{E}[N] \in \{1000, 1001, \cdots, 5000\}$, $P_H \in [0.4, 0.6]$ (percentage of events that belongs to HPP), $n_b \in \{1, 2, 3\}$, $b \in \{5, 6, \cdots, 20\}$, $K \in \{2, 3, 4\}$. The reward distribution parameters θ_i^0, θ_i^p $(i \leqslant K, p \leqslant n_b)$ were generated as iid $U(0, 1)$. For each simulation, we chose a random combination of the above parameters and compared the regret the bound to the regret observed when running BMAB. The two first graphs of Figure 6.5 are plots of the theoretical regret bound versus the empirical regret related to the 1000 sampled simulations.

Remark 6.2.4. *As expected, all graphs exhibit a linear relation that matches with our theoretical results.*

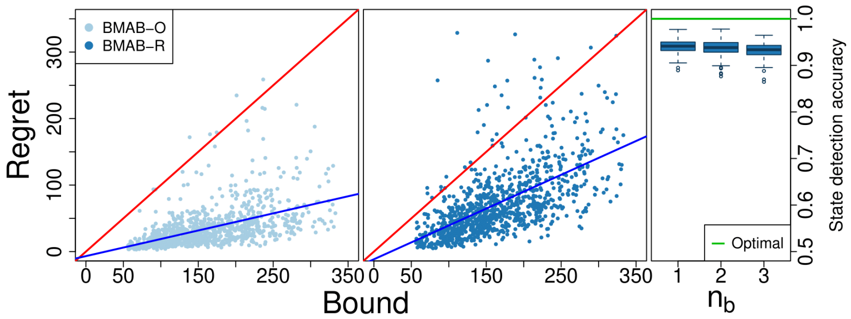


Figure 6.5: **First two graphs:** comparison between theoretical guarantees and experimental results of the regret ($\mathcal{R}(T)$). The red line is the function $f(x) = x$. The second line shows the function $f(x) = ax + b$, where a and b are the coefficients of the linear regression fitting of $\mathcal{R}(T)$ versus $O(\mathcal{R}(T))$. **Last graph:** boxplot of the dispersion of the state detector accuracy.

6.2.4 A realistic state detector

In this section, we will propose a method to group the user-activities into two categories: loyal and curious. For that, we need to develop a realistic state detector that is a crucial step of [**BMAB-R**]. We assume that the loyal audience rate λ_L is known. In Section 6.3 we discuss a method to recover λ_L. Such a rate does not require prior knowledge of the reward distributions and can be measured through the system's traffic logs. Note that, although the system's state is defined at any time t, we only need to predict it when an event t_i happens.

Our method requires a positive integer sensitivity hyperparameter Δ as well as a confidence parameter $\delta \in (0, 1)$. For all i, we then write $\Delta_i = t_i - t_{i-\Delta+1}$ (if $i < \Delta$, $\Delta_i = t_i$). Then, we write $q_\delta(\mu, \rho) = q_{\text{Gamma}}(\mu, \rho, \delta)$ for the (left) quantile function of the Gamma distribution with shape μ and scale ρ, i.e., $\mathbb{P}(X \leqslant q_\delta(\mu, \rho)) = 1 - \delta$ where X follows a Gamma distribution with shape μ and scale ρ.

In order to detect the system state at the time of event t_i we aim to test the hypothesis that the elements of the set $\mathcal{T}_{\Delta_i} = \{t_{i-\Delta+1}, t_{i-\Delta+2}, \cdots, t_i\}$ (if $i < \Delta$, $\mathcal{T}_{\Delta_i} = \{0, t_1, t_2, \cdots, t_i\}$) are timestamps generated by a uniform Poisson process with intensity λ_L.

Our state detector is described in Algorithm 7.

STEP 1 – Compute Δ_i and $\mathcal{T}_{\Delta_i}$ [Lines 1-5]: Firstly, we compute the size of the interval Δ_i that is covered by the set $\mathcal{T}_{\Delta_i}$.

STEP 2 – Hypothesis test [Lines 6-11]: If the timestamps in $\mathcal{T}_{\Delta_i}$ were indeed generated by a Poisson process with intensity λ_L, then the distribution of $\Delta_i = t_i - t_{i-\Delta+1}$ will be a Gamma distribution with shape $\Delta - 1$ and scale λ_L. Accordingly, we calculate the quantile function $q_\delta(\Delta - 1, \lambda_L)$ and test the hypothesis stated by comparing $q_\delta(\Delta - 1, \lambda_L)$ and Δ_i (lines 6-11), returning the state 0 (loyal), if the hypothesis is accepted, and 1 (curious) otherwise.

The three last graphs of Figure 6.6 illustrate our state detector for a fixed $\delta = 0.95$ and different values of $\Delta \in \{5, 10, 15\}$. Experiments show that our state detector has comparable performance to the optimal oracle: the boxplots on the right side of Figure 6.5 show the distribution of the state detector accuracy as a function of the number of bursts. As can be seen, we achieved high accuracy (on average, more than 90% of the states were recovered correctly). The simulations are identical to those used to evaluate our regret guarantees (Section 6.2.3).

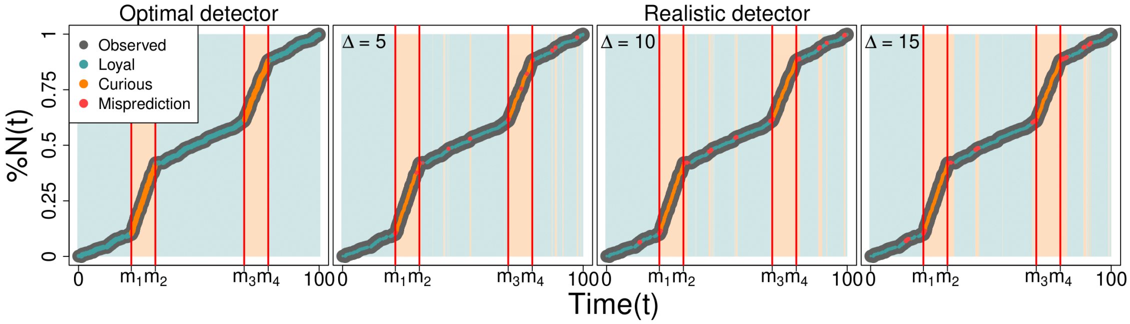


Figure 6.6: State detector and activity categorization. **First graph**: optimal detector scenario. **Three last graphs:** point-wise user-activity categorization (loyal and curious) by using the detector proposed in Section 6.2.4 (we fixed $\delta = 0.95$ and varied $\Delta \in \{5, 10, 15\}$). Red dots means wrong categorization. In all graphs, the vertical lines indicate the actual state transitions (i.e., the set M), the colored stripes at the background represents the system's states detection and the user-activity categories are shown by the dots.

Algorithm 7 State Detector ω-R

INPUT: Event set $\{t_1, t_2, \cdots, t_i\} \in \mathbb{R}^i$, sensitive parameter $\Delta \in \{2, 3, 4, \cdots\}$, confidence parameter $\delta \in (0, 1)$ and loyal audience intensity $\lambda_L \in \mathbb{R}$ (and $\lambda_L > 0$)

OUTPUT: State of the slot machine $s(t_i) \in \{0, 1\}$

```
if $i < \Delta$ then
    $\Delta_i = t_i$
else
    $\Delta_i = t_i - t_{i-\Delta+1}$
end if
$q_\delta(\Delta - 1, \lambda_L) = q_{\text{Gamma}}(\Delta - 1, \lambda_L, \delta)$
if $q_\delta(\Delta - 1, \lambda_L) \geqslant \Delta_i$ then
    return 1
else
    return 0
end if
```

6.3 Disentangling loyal and curious audiences via an EM-approach

This section will describe a method to disentangle loyal and curious audiences and, therefore, calculate λ_L. We start with a brief description of the SFP process. Then we formally define our model and a EM approach to solve it. Finally, we discuss algorithmic details.

6.3.1 Self-feeding process

In our temporal dynamics assumptions (Section 6.1.1) we considered that the events $\mathcal{T}$ can be split into a HPP and a PW-HPP. However, between transitions, we have mixtures of homogeneous Poisson processes. Such a mixture is unstable to separate because we would have infinite compositions with the same probability. Therefore, it might be impossible to separate if we do not know at least one intensity function of the considered processes. Furthermore, in order to calculate the loyal intensity λ_L, we may need to make some assumptions about the transition distribution. To overcome these drawbacks, we approximate our PW-HPP to a SFP. A common characteristic of the SFP instances is the mix of bursty periods alternating with quiet intervals [55].

The conditional intensity of the SFP has a simple dependency on its past. Locally, it behaves similarly to a homogeneous Poisson process, but its conditional intensity rate is inversely proportional to the temporal interval between the two previous events. More precisely, the conditional intensity function is defined as follows:

$$\lambda_s(t|\mathcal{H}_t) = \frac{1}{\mu/e + \Delta t_i} \tag{6.6}$$

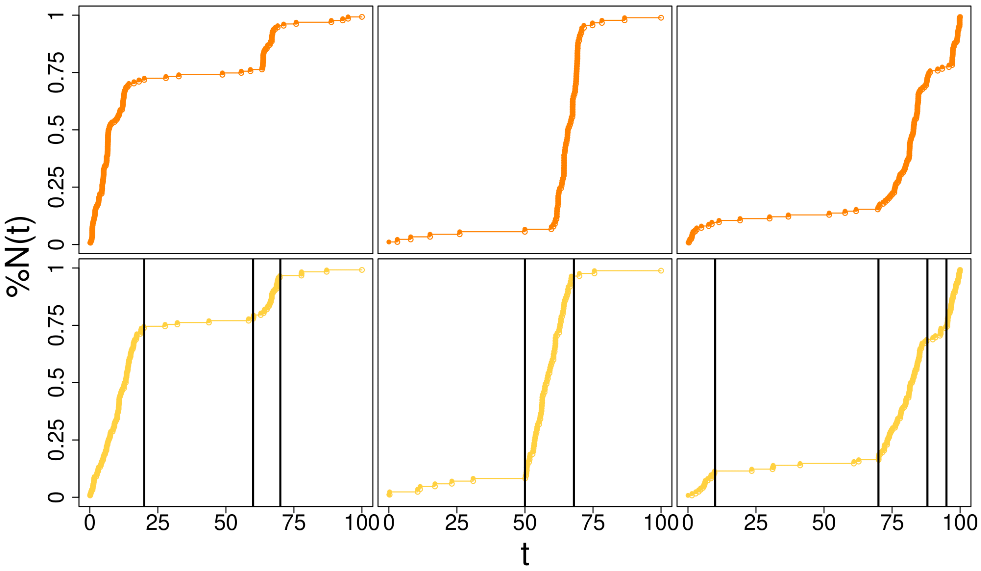


Figure 6.7: Approaching an SFP by a PW-HPP. **First row of graphs:** three different SFP genarations with $mu = 0.1$ and $t \in (0, 100]$; **Second row of graphs:** approximation of the SFP above by a PW-HPP. The vertical lines on the second row of graphs represents the Poisson transitions

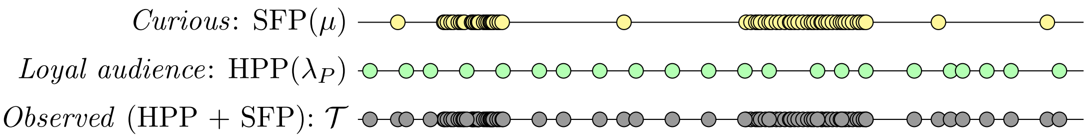
where $\Delta t_i = t_i - t_{i-1}$ and $t_i = \max_k\{t_k : t_k \leqslant t\}$. This implies that the inter-event times $\Delta t_{i+1} = t_{i+1} - t_i$ are exponentially distributed with expected value $\mu/e + \Delta t_i$. The inter-event times Δt_i follow a Markovian property. The constant μ is the median of the inter-event times and $e \approx 2.718$ is the Euler constant.

The first row of graphs in Figure 6.7 shows three realizations of the SFP process in the interval $(0, 100]$ with a parameter $\mu = 0.1$. One striking aspect of this plot is its variability. To illustrate how a SFP can be similar to a PW-HPP, we made an approximation for each SFP instance. As a result, the second row of graphs in Figure 6.7 depicts corresponding of PW-HPP by considering the intensities of the above SFP. As can be seen, the SFP process behaves similarly to a non-homogeneous Poisson process with step-wise fix intensities.

6.3.2 Notation and formal EM construction

We always observe $T = \{t_1, t_2, \cdots, t_n\}$, $0 < t_1 < t_2 < \ldots < t_n$, event timestamps from the mixture of the SFP (*curious*) and the HPP (*loyal audience*). Note that, in practical applications, $T \neq \mathcal{T}$ since we need to predict λ_L before the first recommendation requisition. We use the convention that $z_i = 0$, if $t_i \in$ HPP, and $z_i = 1$, if $t_i \in$ SFP. Recall we define $N(t) = \sum_{i=1}^{n} 1_{t_i \leqslant t}$ to be a function that computes the cumulative number of events up to time t. Finally, we set $\theta = Z$, where $Z = \{z_1, z_2, \cdots z_n\}$ (the latent variables), $\theta_{-i} = \theta \setminus \theta_i$

Figure 6.8: Mixture of point process in timelines for our EM approach. The *curious* (PW-HPP) and *loal audience* (HPP) labels, as well as the transitions (events of M), are not observed.


Curious: SFP(μ)
Loyal audience: HPP(λ_P)
Observed (HPP + SFP): $\mathcal{T}$

where $\theta_i := \{z_i\}$.

The core concept of our EM setup is depicted in Figure 6.8, which corresponds to the BMAB model depicted in Figure 6.2. Similarly, we observe n timestamps of events up to a time t_n (grey dots in the third row). Furthermore, we assume that these events are a mixture of events coming from the *loyal audience* and the *curious*, two independent point processes, shown as yellow and green in the first and second rows, respectively. The primary distinctions are as follows: (1) we approximate our PW-HPP to a SFP; and (2) our EM approach does not address the transitions of the the BMAB's PW-HPP. In practice, we model the PW-HPP's transitions implicitly by the SFP's intensity function. The events associated with the *loyal audience* are also modeled by a second process, a HPP, shown in the second row.

Based on that, consider a general continuous-time Markov process adapted to the filtration $(\mathcal{H}_t)_{t \in \mathbb{R}^+}$, and let $N(a, b)$ be the random number of events in $(a, b]$. The conditional intensity rate function characterizes the distribution and is given by

$$\lambda(t|\mathcal{H}_t) = \lim_{h \to 0} \mathbb{E}\left(N(t, t+h)|\mathcal{H}_t\right)/h.$$

For a small time interval $(t, t+h)$, the value of $\lambda(t|\mathcal{H}_t) \times h$ is approximately the expected number of events in $(t, t+h)$. The simplest example is the homogeneous Poisson process where $\lambda(t|\mathcal{H}_t) = \lambda_p(t) = \lambda_p$, a deterministic (and constant) function, independent of the history of events.

At time t, the history of the process is composed of the observed event timestamps $\{t_1, t_2, \ldots\} < t$ and unobserved labels $\{z_1, z_2, \ldots\}$. Define the following two intensity functions: (1) the SFP intensity $\lambda_s(t) = 1/[(t_{g(t)} - t_{g(g(t))}) + \mu/e]$ where $g(u) = \max(v \in \{-\mu, 0\} \cup T | v < u)$ and $\mu > 0$ is the SFP parameter; and (2) $\lambda_p(t) = \lambda_L \geqslant 0$, is the intensity of the HPP. Similarly we can define $\lambda_s^+(t) = \lim_{\delta \to 0} \lambda_s(t+\delta)$ (resp. $\lambda_p^+(t)$) as the intensity of the SFP (resp. HPP) immediately after t.

Now we can describe our generative model:

STEP 1: Assume $t = 0$;

STEP 2: Generate two exponential variables E_s and E_p with intensities $\lambda_s^+(t)$ and $\lambda_p^+(t)$ respectively. Make $E = \min(E_s, E_p)$.

STEP 3: The next event will take place at $t + E$ and it will belong to the SFP (resp. HPP) component if $E = E_s$ (resp. E_p).

STEP 4: Let $t = t + E$ and continue the generation procedure from STEP 2.

6.3.3 E-STEP

Considering the described generative model we aim to infer the SFP and HPP parameters, respectively, μ and λ_L. To optimize the likelihood, we will use the EM algorithm, relying on Gibbs sampling in the E-step. However, the EM algorithm in the case of point processes requires great care since the events are not independent data and the usual derivations are not appropriate.

This section explains how to use Gibbs sampling to draw a set of latent variables Z (also referred to as θ) from the conditional distribution given a fixed set of parameters μ and λ_L. We start with an initialized value for θ, and we perform a large number N_{Gibbs} of updates on its components. At each update step, we pick $i \leqslant n$ and update the value of the components θ_i according to the conditional distribution of θ_i given the current value of θ_{-i} (and, as always, the value of T). After a large number of iterations, this procedure yields a sample whose distribution is approximately that of a sample of θ given T only. To perform this procedure, we need to compute the conditional probability $\mathbb{P}(\theta_i|T, \theta_{-i})$ for any i, θ_i, θ_{-i}.

Those conditional probabilities and densities are proportional to the corresponding likelihoods. Note that we have the following expression for the likelihood of our model:

$$\begin{aligned}\mathcal{L}(\theta) &= \prod_{i=1}^{n} \lambda_s(t_i)^{z_i} \lambda_p(t_i)^{1-z_i} \times e^{-\int_0^{t_n} \lambda_s(t)+\lambda_p(t)dt} \\ &= \prod_{i=1}^{n} \lambda_s(t_i)^{z_i} \lambda_L^{1-z_i} \times e^{-(\int_0^{t_n} \lambda_s(t)dt+t_n\lambda_L)}. \end{aligned} \tag{6.7}$$

This naturally factorizes as $\mathcal{L}(\theta) = \mathcal{L}_s(\theta)\mathcal{L}_p(\theta)$ where $\mathcal{L}_s(\theta)$ and $\mathcal{L}_p(\theta)$ are, respectively, the components of the likelihood function (evaluated at θ) corresponding to the SFP and HPP components: $\mathcal{L}_s(\theta) = \prod_{i=1}^{n} \lambda_s(t_i)^{z_i} e^{-\int_0^{t_n} \lambda_s(t)dt}$. $\mathcal{L}_p(\theta)$ is defined similarly.

A key observation now is that the factors in (6.7) corresponding to the intervals $(0, t_i]$ and $(t_{f(f(i))}, t_n]$, do not depend on the value of z, where $f(u) = \text{argmin}_j\{t_j | t_u < t_j \wedge z_j = 1\}$ denotes the index of the next SFP event after t_u. Indeed, whether t_i is an SFP or Poisson event only influences the SFP intensity of the next two SFP events. Therefore, we can drastically reduce the computation time by writing equivalently:

$$\mathbb{P}(z_i = z | T, \theta_{-i}) \propto \mathcal{L}_s^i(\Omega_z)\mathcal{L}_p^i(\Omega_z), \tag{6.8}$$

where $\mathcal{L}_s^i(\Omega_z)$ (resp. $\mathcal{L}_p^i(\Omega_z)$) corresponds to the component of the likelihood corresponding the SFP (resp. HPP) and to the interval $[t_i, t_{f(f(i))})$ and $\Omega_z := \theta_{-i} \cup \{z\}$.

$\mathcal{L}_s^i(\Omega_z)$ and $\mathcal{L}_p^i(\Omega_z)$ can be computed directly. This concludes the explanation of the computation of $\mathbb{P}(z_i = z, | T, \theta_{-i})$. Since $z_i \in \{0, 1\}$ is a discrete random variable, it is then straightforward to sample from the corresponding distribution. We, therefore, must perform several Gibbs updates which yield θ_i. This concludes the generation procedure for the E step.

6.3.4 M-STEP

In this section, we explain how to maximize the log-likelihood (corresponding to the current estimate of the conditional distribution of the θ) over the set of parameters $\{\mu, \lambda_L\}$. The procedure described in the E-STEP section allows us to draw N_θ samples $\{\theta_1, \theta_2, \cdots, \theta_{N_\theta}\}$ from the conditional distribution of θ given the current estimate of $\{\mu, \lambda_L\}$. We then update μ via the formula

$$\hat{\mu} = \frac{\sum_{j=1}^{N_\theta} \text{argmin}_\mu \log(\mathcal{L}_s(\theta_j))}{N_\theta}, \tag{6.9}$$

where the likelihood minimization steps are performed via binary search. Note that μ is an easy parameter to estimate as it affects the whole interval. Thus $\text{argmin}_\mu \mathcal{L}_s(\theta_j)$ is already a good estimate even for a single value of j.

Regarding the parameter λ_L, first, let $U(\theta) = \sum_j 1_{z_j=0}$. Averaging over all values of θ, we immediately obtain the following formula for the λ_L:

$$\hat{\lambda}_L = \frac{\sum_{j=1}^{N_\theta} \frac{U(\theta_j)}{t_n}}{N_\theta}. \tag{6.10}$$

6.3.5 Algorithmic details

Algorithm 8 describes how to compute μ and λ_L for a fixed parameter set N_θ and N_{Gibbs}. The last parameter controls how many updates on the latent variables components need to be performed during the E-STEP. In practice, we can decrease N_{Gibbs} gradually during the EM-Algorithm execution to speed up convergence. Note that we set $N_{Gibbs} = \mathcal{O}(n)$ so that each timestamp is expected to be updated a constant number of times. N_θ, the size of the warm-start set, can be adapted depending on the available computer resources. However, a small number proved to be sufficient. We refer to [112] for more information about EM-Algorithm convergence.

Algorithm 8 Desintangler

INPUT Random series of events $T \in \mathbb{R}^n$, size of the set of warm-starts $N_\theta \in \{1, 2, 3, \cdots\}$ and number of Gibbs updates $N_{Gibbs} = \mathcal{O}(n)$

OUTPUT: SFP (curious) parameter μ and HPP (loyal) parameter λ_L

```
$\mu, \lambda_L, \theta_1, \theta_2, \cdots, \theta_{N_\theta} \leftarrow$ warmStarts($T, N_\theta$)
while Not converged do
    for $i \in \{1, 2, \cdots, N_\theta\}$ do
        $\theta_i$ = GibbsSampler($\theta_i, \mu, \lambda_L, N_{Gibbs}$)
    end for
    $\mu, \lambda_L$ = M-STEP($\theta_1, \theta_2, \cdots, \theta_{N_\theta}$)
end while
return $\mu, \lambda_L$
```

Remark 6.3.1. *In the E-STEP, to prevent the model from getting stuck in low-likelihood regions, we performed likelihood-based re-sampling: after several iterations, we replace the current estimate of the set* $\{\theta_1, \ldots, \theta_{N_\theta}\}$ *by a set of* N_θ *elements drawn with replacement from that set with probabilities proportional to the likelihoods.*

6.3.6 Warm-starts for θ and parameter initialization

Instead of an initial random choice of θ, we propose a warm start strategy that can significantly reduce the convergence time of our algorithm. For generating a set θ we proceed as follows:

STEP 1: First, we divide the interval $[0, t_n)$ into n_k sub-intervals $I_1, \ldots, I_{n_k}$ of size t_n/n_k, i.e. $I_i = [(i-1)t_n/n_k, it_n/n_k)$.

STEP 2: For each $i \leqslant n_k$, we define $u_i = \sum_{j \leqslant n} 1_{t_j \in I_i}$, the number of events belonging to interval I_i.

STEP 3: Then we sample one sub-interval I^* from $I^1, I^2, \ldots, I^{n_k}$ with $\mathbb{P}(I_i) \propto 1/u_i \forall u_i \neq 0$.

STEP 4: Estimate $\lambda_L = u_*/(t_n/n_k)$.

STEP 5: To estimate μ and Z, we can now calculate p_i and draw $z_i \sim \text{Bernoulli}(1-p_i)$. If $t_i \in I_j$, then $p_i = \min(\lambda_L \times (t_n/n_k)/u_j, 1)$. Thus, we can now estimate μ based on our estimated sample of the underlying SFP process with ordered timestamps $V = \{v_1, v_2, \cdots\} \equiv \{t_i | z_i = 1\}$, i.e. $\mu = \text{median}(\{\Delta_i\})$, where $\Delta_i = v_i - v_{i-1}$ and $\Delta_1 = v_1$ (for details, see [55]).

We now have an initial value for the parameters λ_L, μ and the set of latent variables $\theta = Z$, which concludes the warm start and initialization phase.

6.3.7 Complexity analysis

The execution time of the algorithm is highly dependent on the observed data. Nevertheless, if we restrict the number of iterations of the EM algorithm to $N_{EM} \ll n$, our algorithm has complexity $\mathcal{O}(N_{EM} N_{Gibbs} n)$ (indeed, each component of the likelihood calculations involved in the computation of the conditional probabilities consists of a sum over each event in the interval $[t_i, t_{f(f(i))})$). In the worst case, this interval is the whole of T and the computation complexity for one Gibbs iteration is $O(n)$).

6.4 Conclusion

In this chapter, we introduced a novel MAB problem and an algorithm to solve it, which we called BMAB. Our approach is context-free, as it does not incorporate any side information about users or items. Additionally, it is non-stationary, as the distribution of rewards changes with time. Unlike previous approaches, our method models the time in a continuous setting and considers the audience's dynamics to explore and exploit the reward environment.

BMAB handles reward distribution by segmenting user activity into two types: loyal and curious. A critical phase in our methodology is determining the rate λ_L at which events are transmitted from the user's loyal audience. This rate enables us to determine which audience is the most prevalent at a given time t. To find λ_L, we developed a method for modeling audiences as a hybrid of HPP and SFP. Following that, we presented an EM technique to learn the parameters of the aforementioned processes' parameters.

Additionally, we introduced a state detector capable of determining with high accuracy which audience member (loyal or curious) is dominant. Finally, we provided regret guarantees for our algorithm under mild constraints.

Chapter 7

EXPERIMENTS AND APPLICATIONS

This chapter presents synthetic and real-world data experiments, as well as applications of the methods described in Chapter 6. We began by analyzing the BMAB algorithm in different ground-truth regimes. Then, we verify our method's performance in a context-free recommendation scenario on real-world datasets versus several state-of-the-art baselines. Finally, we characterize online time series to show how our technique can discriminate between loyal and curious audiences.

The **main** contributions of this chapter are the following:

- we evaluate the BMAB algorithm and compare it to several baselines in two experimental strands: synthetic data simulations and real-world datasets. We compare our method to six state-of-the-art baselines and achieve competitive results;
- regarding the task of disentangling loyal and curious audiences: (1) we performed extensive empirical work, in which we show that our model fits real-world datasets with better results than alternative models; (2) we also performed synthetic data experiments to analyse the behaviour of our method in different ground-truth regimes;
- we proposed indices able to describe and quantify the *loyal audience* and we computed them for eleven real-world data sets.

Parts of this chapter are based on:

> **Rodrigo Alves**, Antoine Ledent, and Marius Kloft. Burst Induced Multi-armed Bandit for Learning Recommendation. *Proceedings of the 15th ACM RecSys Recommender System Conference*, 2021.
>
> **Rodrigo Alves***, Antoine Ledent*, Renato Martins Assuncao, Pedro Vaz de Melo, and Marius Kloft. Are you here to stay? disentangling the loyal audience from the curious on social media. **Submitted to** *ACM Transactions on Knowledge Discovery from Data (TKDD)*, Aug 2021.

* The authors contributed equally to this research.

7.1 Context-free recommendation

To compare our method with the baselines, we conducted experiments with two data strands: synthetic and real-world datasets. In the first case, we performed several simulations to verify the performance of BMAB in different ground truth regimes. In the second strand, we validated our model on four recommender systems datasets. We show that our methods exhibit SOTA performance in all cases.

7.1.1 Baselines and parameter selection

We evaluated the following baselines:

- **TS – Thompson Sampling:** traditional stationary MAB algorithm [18]. For more details, see Section 6.2.1.
- **EXP3:** broadly used MAB algorithm that considers a non-stationary environment. EXP3 uses a parameter γ to control exploration and exploitation during all the period (γ was selected following Corollary 3.2 of [113]).
- **EXP3DD – EXP3 with Drift Detection:** EXP3 with a reward distribution shift detection procedure. When the best arm changes, EXP3DD re-initializes the algorithm. Hyperparameter selection was performed according to Section V of [22].
- **DUCB – Discounted UCB:** A UCB-type method that tackles non-stationarity by maintaining exploratory behavior throughout the event horizon. Hyperparameter tuning following Section 3.1 of [25].
- **MUCB – Monitored UCB:** MUCB detects the change on the arms' reward distribution by comparing the rewards in the two last time intervals of the same size. The parameters w and b were setting according to Section 5 (Remark 1)[23].

- **WMD – Windowed mean-shift detection:** WMD is a framework that uses time windows to detect shifts in the arms' reward distribution. As in [21] we set ϵ and τ according to Section 5 and Theorem 4.1.

In all cases, the cited parameters and sections follow the notation of the respective papers. We can split the baselines into three groups depending on which rewards environment they were designed to work in: stationary **TS**; non-stationary **EXP3**; and piece-wise stationary, **EXP3DD**, **DUCB**, **MUCB** and **WMD**. For **BMAB-O** and **BMAB-R**, we empirically selected the forgetting-rate $\gamma = 0.70$, the detector confidence index $\delta = 0.95$, and the detector window $\Delta = 10$. We assume we know λ_L in the synthetic strand. To find λ_L in the real-world data, we use the method described in Section 6.3.

7.1.2 Generation of synthetic audience dynamics

The generation procedure consists of five steps:

STEP 1: Choose a set of parameters $\{\lambda_L, \tilde{N}, P_H, n_b, b\}$. λ_L will be the loyal audience intensity. The (curious audience) intensity in the bursty periods will be set to $b\lambda_L$. P_H and $\tilde{N}$ will have the following properties: $\tilde{N}$ will be the expected number of timestamps so that $\tilde{N} = \mathbb{E}(N)$, and the expected number of timestamps attributed to the loyal audience (outside bursts) will be $P_H\tilde{N}$. n_b will be the (fixed) number of bursts.

STEP 2: Generate a Poisson process with event rate λ_L along the time interval $(0, T]$, where $T = P_H \times \mathbb{E}[N]/\lambda_L$.

STEP 3: Split the interval $(0, T]$ into the set of n_b contiguous sub-intervals of the same size $\mathcal{U} = \{(0, T_1], (T_1, T_2], \cdots, (T_{n_b-1}, T]\}$.

STEP 4: Let $\lambda_C(t) = b\lambda_L$, when $s(t) = 1$, and $\lambda_C(t) = 0.05\lambda_L$, otherwise. We consider burst intervals of size $T_b = ((1 - P_H) \times T)/(n_b \times b)$. This guarantees that the expected number of (curious) events during bursts is $(b\lambda_L)n_n((1 - P_H) \times T)/(n_b \times b) = (1 - P_H)\lambda_L T = (1 - P_H)\tilde{N}$. For each $\mathcal{U}_i = (x, y]$ generate $m_{2(i-1)+1} \sim \text{Uniform}(x, y - T_b)$ and $m_{2(i-1)+2} = m_{2(i-1)+1} + T_b$.

STEP 5: Generate the PW-HPP given the set M and $\lambda_C(t)$ of step 4.

Our approach can ensure a relationship between the activities of the users and the popularity of the items (as described in Section 6.1) if the parameters are properly chosen. Figure 6.1 depicts an example of this generating procedure.

7.1.3 Synthetic data experiments

To compare our method with baselines, we designed a reward setting where the rewards depend strongly on the state of the system. We set the parameters as follows. By using the generation procedure proposed in Section 7.1.2, we set $\lambda_L = 1$ and varied $\mathbb{E}[N] \in \{1000, 2000, 5000\}$, $P_H \in \{0.25, 0.5, 0.75\}$, $n_b \in \{1, 2, 3\}$ and $b \in \{5, 10, 20\}$. We set $K = 3$ and set the loyal-state parameters as follows $[\theta_1^0, \theta_2^0, \theta_3^0] = [0.3, 0.4, 0.5]$. During the each burst, the reward parameters are the same as in the loyal state except for one arm, whose parameter is set to 0.9. The sequence of arms whose reward changes is selected from $\{1, 2, 3\}$ by uniform sampling without replacement. Figure 6.1 shows an instance of the generation procedure for $\lambda_L = 3$, $\mathbb{E}[N] = 500$, $P_H = 0.5$, $n_b = 2$ and $b = 5$. The effect of step 5 can be visualized in Figure 6.3.

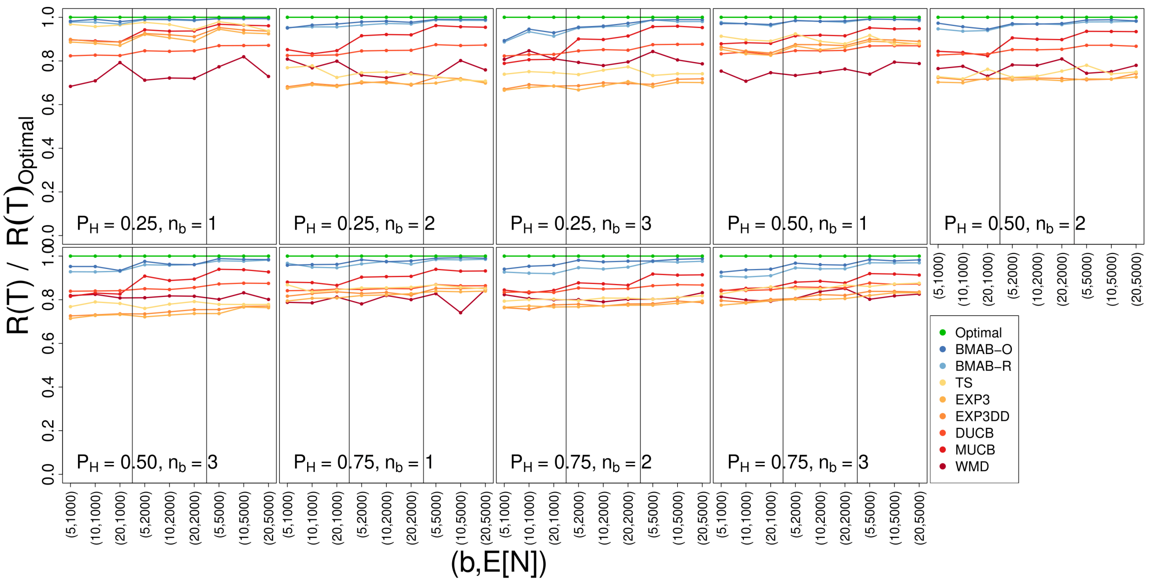


Figure 7.1: Summary of the results of the synthetic data experiments. The graphs are organized according to the tuple $\{\mathbb{E}[N], P_H, n_b, b\}$. The green line (”Optimal”) is the theoretical reward of the best possible algorithm. Each data point corresponds to the average of 20 simulations.

For each set of $\{\lambda_L, \mathbb{E}[N], P_H, n_b, b\}$ we generated 20 samples. For each sample, we performed **[BMAB-O]**, **[BMAB-R]** and the baseline algorithms. Figure 7.1 show the performed simulation results. The green lines ("Optimal") in Figure 7.1 show the performance of a hypothetical algorithm with the ability to always select the best arm (with the largest θ). Thus, in theory, no algorithm can achieve better performance. Each point in Figure 7.1 is the average *normalized* reward $R(T)/R_{\text{Optimal}}(T)$, averaged over 20 samples.

Remark 7.1.1. *Note that both **[BMAB-O]** and **[BMAB-R]** consistently outperform the baselines in all the considered scenarios. In addition, the proximity of the two BMAB curves reveals **[BMAB-R]**'s ability to recover the correct states with high accuracy, with this fact being particularly marked in the case where there is an even mix of both point processes ($P_H = 0.5$). Smaller values of n_b led to better performance. This result matches with our theoretical results presented in Section 6.2.3.*

7.1.4 Real-world data experiments

In this section, we present our results on real data. We selected the four following recommender systems datasets:

- **Behance:** Behance is a social media platform devoted to the dissemination and discussion of creative work. In this RS, each user has the option to appreciate ("like") a piece of art.
- **Google trends:** We collected the time series related to the singers Psy and David Bowie ($K = 2$) from 2008 to 2020 (YouTube search engine, only USA). The Google trends API only returns a normalized audience (an integer value, maximum 100) for each month. As a result, we modeled each month as 1 unit of time: Jan/2008 $\in (0, 1]$, Feb/2008 $\in (1, 2], \cdots$, Dec/2020 $\in (155, 156]$. To convert normalized audiences into timestamps, we generated events uniformly along the corresponding month. For instance, if the API returned x events for Feb/2008, we generated $10x$ events between 1 and 2.
- **Outbrain:** Outbrain is a web advertising platform that displays content within websites. The data set contains users' clicks at recommended content.[10]

---

[10] The data is available at `https://www.kaggle.com/c/outbrain-click-prediction/data`

- **MovieLens:** MovieLens is a non-commercial website for movie recommendations. We used MovieLens (25M) which is a broadly used recommendation dataset.[11]

For the Behance, Outbrain and MovieLens datasets, we selected the five most popular items as the arms of our bandit problem. The number of arms K, the observed time T and the number of events N of all datasets are available in Table 7.1. In all cases, we split the time T into two subsets: the first one (T_{λ_L}, cf. Table 7.1) is used to estimate λ_L; while the rest of the data is used to evaluate **[BAMB-R]** and the baselines.

Remark 7.1.2. *We note that our general strategy can be adapted and incorporated in different RS contexts. For example, if rewards or user-reward pairs can be embedded in a dictionary space, we could use our method with a modified version of Thompson Sampling where at each observation, each positive component of a virtual feature vector is Thompson Sampling-updated.*

Note that both our method and the baselines are stochastic. As a result, we performed all the algorithms for each dataset 50 times. For the real datasets, the reward is deterministic. At each time t_i, we consider that $r(t_i) = 1$, if the algorithm (correctly) recommends the item that the user liked (Behance, MovieLens), searched (Google trends) or clicked on (Outbrain), and $r(t_i) = 0$ otherwise.

We present the results in Table 7.1. We evaluated the performance of **[BMAB-R]** against the baselines by considering the average reward ($R(T)/N$). In real datasets, Optimal and **[BMAB-O]** cannot be defined due to the lack of well-defined bursts.

---

[11] The data is avaiable at https://grouplens.org/datasets/movielens/

Table 7.1: Description of the real-world databases and summary of the results of the two experiments with real-world data. **Metric:** average of the observed reward ($R(T)/N$) and its standard deviation (higher values are better). For synthetic data, the rewards are normalized by the reward of the Optimal algorithm ($R(T)/R_{\text{Optimal}}(T)$).

	Behance	**Google Trends**	**Outbrain**	**MovieLens**
(K,N)	$(5, 7122)$	$(2, 19850)$	$(5, 86689)$	$(5, 270403)$
T	[Jun to Nov/2011]	[2008,2020]	[14 to 28/Jun/2016]	[Sep/2001,Oct/2019]
$\mathbf{T_{\lambda_L}}$	[Jun/2011]	[2008]	[14/Jun/2016]	[Sep/2001,Dez/2003]
BMAB-R	$\mathbf{0.5937 \pm 0.004}$	$\mathbf{0.7756 \pm 0.002}$	$\mathbf{0.5449 \pm 0.008}$	$\mathbf{0.22656 \pm 0.001}$
TS	0.3975 ± 0.042	0.6972 ± 0.033	0.4123 ± 0.018	$\mathbf{0.22652 \pm 0.002}$
EXP3	0.2202 ± 0.011	0.5249 ± 0.020	0.3053 ± 0.021	0.2223 ± 0.001
EXP3DD	0.2320 ± 0.013	0.5337 ± 0.030	0.3062 ± 0.024	0.2244 ± 0.001
DUCB	0.5014 ± 0.003	0.7616 ± 0.001	0.4852 ± 0.001	0.1701 ± 0.001
MUCB	0.5055 ± 0.006	0.7640 ± 0.002	0.4637 ± 0.001	0.2182 ± 0.001
WMD	0.4197 ± 0.037	0.6951 ± 0.021	0.4237 ± 0.014	0.2039 ± 0.005

Remark 7.1.3. *We observe that our algorithm significantly outperformed all the baselines in all the real datasets except MovieLens, where the performance was comparable to that of the TS baseline. It is likely to happen because the behavior of the five most popular items is stationary. Thus, our algorithm detects no bursts and behaves exactly like TS in that case. Note that in particular, in the Outbrain and Behance datasets, our method outperformed the second best algorithm ([**MUCB**] and [**DUCB**], respectively) by* 12% *and* 17% *(respectively).*

7.2 Characterizing online timeseries

To analyze our disentangling model in different ground-truth regimes and compare it with the baselines, we conducted two experiment strands: synthetic data simulations and real-world datasets. In the first case, we use the generation procedure explained in Section 6.3.2, varying μ and λ_L. This allows us to evaluate how our model performs in different regimes. We chose the parameters in order to control (1) the number of events (n) and (2) the expected percentage (P_H) of the observed events that belong to each of the latent processes (SFP and HPP). We also validate our model on fourteen real datasets split into four groups: collaborative recommendation systems (*Digg* and *Reddit*), forums (*AskMe*, *MetaFilter*, and *MetaTalk*), hashtag-based chat (*Twitter*) and collaborative working (Github(Users) and Github(Project)). In total, we analyzed more than 63 million events.

7.2.1 Synthetic data generation and evaluation

Let $P_H = \sum_{i=1}^{n}(1 - z_i)/n$ be the proportion of observed events that belong to HPP. We chose sets of μ and λ_L corresponding to estimated values of $(\mathbb{E}(n), \mathbb{E}(P_H))$ in the set $\{500, 750, 1000\} \times \{0, 0.25, 0.5, 0.75, 1\}$. For each pair $(\mathbb{E}(n), \mathbb{E}(P_H))$, we conducted 50 simulations (we use the generation procedure described in Section 6.3.2) and assessed our methods via two metrics: $\delta(\mu, \hat{\mu})$ and $\delta(\lambda_L, \hat{\lambda}_L)$. The function $\delta(a, b)$ is defined as

$$\delta(a, b) = \tfrac{|a-b|}{a}. \tag{7.1}$$

Therefore, the metrics assess our method's ability to accurately recover the parameters of the ground truth model given the observations. The reason we do this instead of simply counting the proportion of correct labels is as follows. Correctly classifying the timestamps is both more difficult and less interesting inside a burst compared to calm periods. On the other hand, $\delta(\mu, \hat{\mu})$ and $\delta(\lambda_L, \hat{\lambda}_L)$ are far less sensitive to the label assignments in a short bursty period, but a small value of $\delta(a, b)$ still indicates excellent performance.

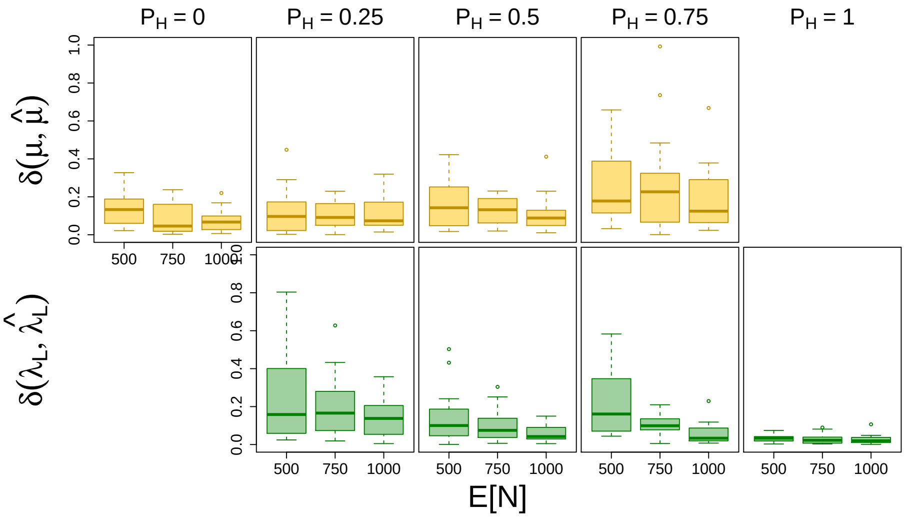


Figure 7.2: Summary of the results of our experiments with synthetic data. $\delta(\mu, \hat{\mu})$ (top, yellow) and $\delta(\lambda_L, \hat{\lambda}_L)$ (bottom, green) distributions grouped by $(\mathbb{E}(P_H))$. The x-axis shows the expected number of observed events.

We report the results of our experiments evaluated with both metrics in Figure 7.2.

Remark 7.2.1. *The box plots show that our method has a strong ability to recover the underlying parameters of SFP and HPP based only on the observed timestamps. Larger values of the number of events (n) correspond to smaller values of* $\delta(\mu, \hat{\mu})$ *and* $\delta(\lambda_L, \hat{\lambda}_L)$*. Mixtures with higher* P_H *tend to produce fewer bursts and therefore have a more uniform behavior over the whole observed period. Consistently with this, we observe that larger values of* P_H *correspond to smaller values of* $\delta(\lambda_L, \hat{\lambda}_L)$*.*

7.2.2 Baseline models and fitting metrics for real datasets

Our model relies only on the observed event timestamps. We compare our model with the following similar models:

- **Hawkes processes** [42, 50, 54, 114, 115]: Hawkes processes are a class of self-exciting processes which are widely used for modeling web communications. The Hawkes process model assumes that any event increases the probability of additional events. Its conditional intensity is $\lambda(t|\mathcal{H}_t) = \lambda + \sum_{t_i<t} K(t - t_i)$, where $K(x) > 0$ is the kernel function, which satisfies $\int_0^\infty K(x)dx < 1$, to ensure stationarity.

- **BuSca** [26]: similarly to our model, BuSca is also a mixture process involving a Poisson process and a self-exciting process. The conditional intensity of the BuSca model is given by $\lambda(t|\mathcal{H}_t) = \lambda + \frac{1}{\Delta_t + \mu/e}$, where $\lambda \geqslant 0$ and $\mu > 0$ are constants and Δ_t is the last SFP interval before t. The primary distinction between BuSca and our model is in the optimization procedure: we present here an approach that requires far less approximations when computing the EM likelihood.

To assess the Hawkes Process method's performance we used the random time change theorem to transform a HP into a unit rate Poisson process (see [50]). After the transformation we computed the determination coefficient R^2_{Hawkes} corresponding to the linear regression problem predicting the cumulative number of events $N(t)$ for all ts in the transformed process. Similarly, for BuSca, we computed R^2_S (SFP) and R^2_{HPP} (homogeneous Poisson process) for the disentangled processes (see [26]) and computed the final coefficient $R^2_{BP} = (R^2_S + R^2_{HPP})/2$.

To check the goodness of fit of our model, we first output the $\{\hat{Z}\} \subset \theta_i$, where $i = \text{argmax}_j\{\mathcal{L}(\theta_j)\ ;\ 1 \leqslant j \leqslant N_\theta\}$, which allows us to disentangle the HPP from the SFP. For the SFP fitting, we took the inter-event times sample and built the empirical cumulative distribution function $\mathbb{F}(t)$ leading to the odds-ratio function $OR(t) = \mathbb{F}(t)/(1-\mathbb{F}(t))$. Then, we computed the R^2_{SFP} coefficient of the linear regression problem predicting the cumulative number of events $N(t)$ versus the $OR(t)$ (see [55]).

The computation of R^2_{HPP} consists in estimate R^2 as the determination coefficient corresponding to the linear regression problem predicting $N(t)$ from t on the interval by considering the timestamps $\{t_i|z_i = 0\}$. Finally, we compute $R^2 = (R^2_{SFP} + R^2_{HPP})/2$.

Remark 7.2.2. *All R^2 coefficients vary between 0 (worst case) and 1 (best case).*

7.2.3 Data Sets

We show the usefulness of our model by fitting the following eleven real-world data sets, which we also described in Table 7.2.

- **AskMe**, **MetaFilter** and **MetaTalk**: the time-series are topics of an online discussion forum and the events consist of comments timestamps[12].

- **Digg**, each time-series corresponds to a different news post in the website and the events are the *diggs*, similar to Facebook *likes*, given to the respective post.

---

[12] Avaliable at `http://stuff.metafilter.com/infodump/`

- **Enron**, a sequence of events is associated with an e-mail account. They are the incoming and outgoing messages timestamps [13].

- For **Github**, we split this dataset into two parts: **Github (Users)** and **Github (Projects)**. In the first one, the events are activities of a user (in different projects). In the second one, the events correspond to the activities of different users on the corresponding project.

- **Google Trends**: the time series corresponds to the fraction of YouTube views over time. We consider only USA users. Each topic is related to famous people such as singers and politicians, which were defined and collected by the author of this dissertation.

- **Twitter**: the event timestamps correspond to tweets featuring a given hashtag.

- **Youtube**: each time series corresponds to a YouTube video and the events are the timestamps of users' comments.

- **Yelp**: dataset consists of timestamps of user ratings for several restaurants.

Table 7.2 shows the total number of RSEs (#RSE), the average number of events (n), as well as the average of the indices *absolute loyalty* κ and *relative loyalty* $\tilde{\kappa}$ (see Section 7.2.5) for each data set.

Table 7.2: Fitting and characterization of the datasets

	#RSE	**n**	$\boldsymbol{\kappa}$	$\boldsymbol{\tilde{\kappa}}$	$\boldsymbol{R^2}$(our model)	$\boldsymbol{R^2_{BP}}$	$\boldsymbol{R^2_{Hawkes}}$
AskMe	490	133	0.37	0.42	**0.8496 ± 0.11**	0.6210 ± 0.11	0.3224 ± 0.15
Digg	972	122	0.31	0.43	**0.8947 ± 0.10**	0.7234 ± 0.09	0.4828 ± 0.18
Enron	131	1208	0.60	0.37	**0.9465 ± 0.07**	0.9147 ± 0.06	0.5879 ± 0.25
GitHub(U)	47450	603	0.77	0.50	**0.9568 ± 0.03**	0.9550 ± 0.04	0.8966 ± 0.08
GitHub(P)	21852	629	0.74	0.50	**0.9546 ± 0.03**	0.9484 ± 0.04	0.8634 ± 0.10
G. Trends	450	1297	0.63	0.52	**0.9491 ± 0.07**	0.9392 ± 0.01	0.8317 ± 0.29
MetaFilter	8232	175	0.43	0.48	**0.8927 ± 0.09**	0.7125 ± 0.11	0.3909 ± 0.18
MetaTalk	2452	203	0.43	0.49	**0.9174 ± 0.08**	0.7921 ± 0.11	0.4537 ± 0.20
Twitter	16762	1078	0.71	0.49	**0.9472 ± 0.05**	0.8877 ± 0.12	0.8107 ± 0.23
Yelp	1522	135	0.25	0.37	0.9088 ± 0.13	**0.9443 ± 0.08**	0.8461 ± 0.14
YouTube	250	3241	0.59	0.49	**0.9701** ± 0.02	**0.9701 ± 0.01**	0.7010 ± 0.18

---

[13] Avaliable at: `https://www.cs.cmu.edu/~./enron/`

7.2.4 Fitting and characterization of real-world datasets

Table 7.2 shows the goodness-of-fit statistics (average and standard deviation) for our model and the baselines grouped by dataset.

Remark 7.2.3. *Our model surpasses the Hawkes process method in all datasets considered. It also consistently outperforms BuSca (better fitting in 9 out of 11 datasets).*

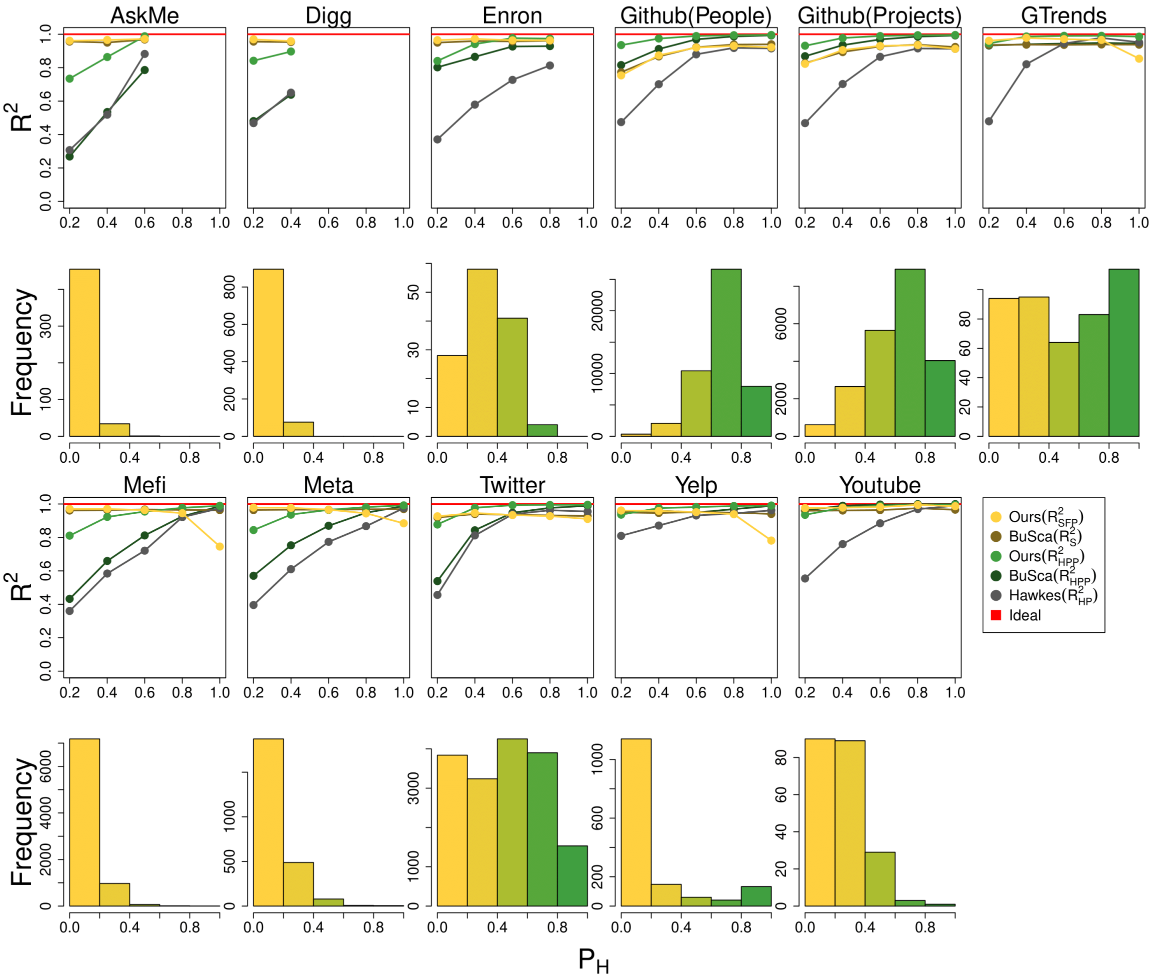


Figure 7.3: **Top:** fitting performance (determination coefficient) as a function of the proportion $P_H = |PP|/n$ of Poisson events per dataset. **Bottom:** distribution of the proportion $P_H = |PP|/n$ per dataset.

Figure 7.3 show how our disentangled models behave under different regimes. To construct the figure, we computed P_H and rounded the value considering the range $\{0, 0.2, 0.4, \cdots, 1\}$. The extremes correspond, respectively, to an approximately a pure SFP and a pure Poisson process (see histograms on the bottom of Figure 7.3).

Remark 7.2.4. *We can observe in Figure 7.3 that our model improves significantly when the bursty behavior dominates the mixture. Indeed, the high concentration of the* R_2 *(as* R^2_{SFP} *and* R^2_{HPP}*) statistics of our model close to the maximum possible value shows that our model can accurately fit the time series considered, as well as disentangle the mixed process into its two hidden components (HPP and SFP).*

7.2.5 Absolute and relative loyalty

After learning the component intensities $\lambda_s(t)$ and $\lambda_p(t)$ we can contrast their absolute and relative influence on the observed events. We define two indexes, both in the interval $[0, 1]$:

$$\kappa = \frac{1}{b-a}\int_a^b \frac{\lambda_p(t)}{\lambda_p(t)+\lambda_s(t)}dt \text{ and } \widetilde{\kappa} = \frac{\int_a^b \lambda_p(t)dt}{\int_a^b(\lambda_p(t)+\lambda_s(t))dt}. \tag{7.2}$$

The *absolute loyalty* κ tells us the importance of the *loyal audience* averaged over the time interval $[a, b]$, whilst the *relative loyalty* $\widetilde{\kappa}$ describes the proportion of the activity in the interval $[a, b]$ that is assigned to the *loyal audience*. The plot on the left-hand side of Figure 7.4 shows the average $\tilde{\kappa}$ versus the average κ for each of the real datasets. For

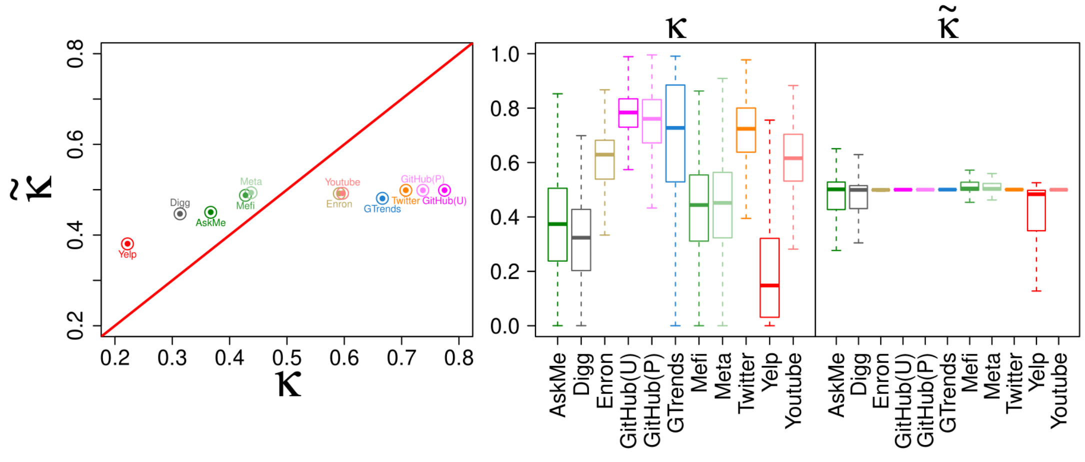


Figure 7.4: Left: Scatterplot of the average $\tilde{\kappa}$ versus the average κ for each of the real datasets. Right: Dispersion of the indicators κ and $\tilde{\kappa}$ per datasets.

instance, consider the two pairs associated with GitHub: $\kappa \approx 0.8$ but $\tilde{\kappa} \approx 0.5$. It shows that while the *loyal audience* (HPP component) dominates most of the time, only half of the activities are carried out by them, i.e., the other half comes from the *curious*.

We observe a significant dispersion on the *absolute loyalty* κ when related to the topic of the time series (second graph, Figure 7.4). However, *relative loyalty* $\tilde{\kappa}$ is relatively stable between topics (third graph, Figure 7.4). Thus, the percentage of activity performed by the loyal audience does not change much among topics. The distribution of these activities, on the other hand, differs according to item.

7.3 Conclusion

Using the methods given in Chapter 6, we conducted synthetic and real-world data experiments. By assessing the empirical reward, we observed that the BMAB algorithm consistently beats the baselines in all synthetic circumstances. Additionally, in the real-world recommender scenario, our algorithm outperformed all baseline algorithms except for one dataset where performance was comparable to that of the Thompson Sampling algorithm. We determined that the cause was that audience dynamics remained stable during the studied period.

In terms of separating loyal and curious audiences, our model outperforms by better fitting the great majority of the investigated real-world datasets. We then proposed two indices to characterize the system's audience: absolute and relative loyalty.

Chapter 8

RELATED WORK AND DISCUSSION

8.1 Related work on multi-armed bandits

MABs were introduced in [18] and more formally defined in [116]. Some classic and broadly used algorithms to solve this problem is the Thompson sampling (TS) algorithm [18, 117], ϵ-greedy policies [118], Exp3 [113] and strategies based on Upper Confidence Bounds (UCB) [17]. TS algorithm enjoys strong empirical performance [119], regret guarantees [111] and has been successfully applied in a wide variety of RSs domains [120–124].

In *non-stationary* MABs, the reward distribution is allowed to change through time. There are two major classes of non-stationary MABs: *adversarial* MABs and *piece-wise* stationary MABs. In adversarial MABs, an adversary controls the payoff generation, so no statistical assumptions are imposed [113]. However, the problem is still stationary in the sense that the aim is to return a single arm that is the globally optimal action at a fixed time horizon. In contrast, the reward generation is non-stationary on the whole time horizon in *piece-wise* stationary MABs, but it is stationary on several unknown intervals [21]. Our model belongs to the latter category. Previous work on *piece-wise* stationary MABs can be further categorized into *context-aware* and *context-free* approaches, depending on whether side information (user/item features) are exploited.

In related work on (context-aware) piecewise stationary MABs, [45] proposed LogUCB, an extension of UCB that estimates the average reward of a topic through a logistic regression on its features. In line with our work, [16] presents a MAB algorithm that also considers the temporal influence on the item consumption probability. They construct the policy algorithm as a probabilistic framework that uses as context a high-dimensional vector containing side information about the users (demographic information) and the items (query keywords). There are two key differences between this work and ours: (1) our method is context-free (does not need feature vectors); and (2) instead of treating the problem as a discrete-time one, our model actively exploits continuous temporal dynamics to detect possible changes in the reward environment.

Related work on *context-free* MABs includes [22], who performed constant exploration inspired by the EXP3 algorithm to detect changes in which arm is the best, while [23] achieved this by simply comparing the rewards in the two last time-intervals of size w. In both cases, whenever such a distributional change is detected, the backbone MAB algorithm is restarted with the aim of finding the best arm under the new distribution. In the same direction, [21] proposed a general framework that can be used together with several algorithms. After also dividing the event horizon into several equal-sized windows (of size w), they compute scores for each arm based on both the rewards and the number of observations in the following two intervals: (1) the last window in which a change in the best arm was observed, and (2) the last observed window. If the absolute difference between the scores in the two windows is greater than a hyperparameter ϵ, the algorithm is re-initialised. Similarly, [47] performed change detection by comparing two previous time windows. Their model also relies on estimates of the probability of false alarm and the probability of missed detection to improve robustness. Instead of resetting the algorithm altogether, [24] adopted a fixed sliding training window while [125] used a different window size for each arm. With a slightly different approach, [25] proposed the so-called 'Discounted-UCB (DUCB)' algorithm. The main idea is to give a higher selection probability to two classes of arms: the arms which recently returned high rewards and the arms which were not recently selected. Therefore, instead of resetting the whole procedure, DUCB tackles the non-stationarity by maintaining a minimum amount of exploration throughout the event horizon. By mixing the two previous approaches (discounted reward and sliding window), [49] assigns more relevance to the recent rewards. The last w rewards are not discounted whilst the remaining ones are. [48] proposed a piece-wise MAB algorithm that detects abrupt changes in the reward distribution through a hypothesis test. As a criterion for this hypothesis test, they rely on the Page-Hinkley statistic, which involves a random variable defined as the difference between the reward time t and the average reward, cumulated in the last m steps. Our method's main difference from other context-free methods lies in our shift detection procedure. Instead of detecting changes only in the reward distribution, we analyze the system's temporal dynamics to identify behavioral changes in the audience. Our hypothesis is that such behavioral change is associated with the items' popularity.

8.2 Related work on the dynamics of human communication

Popularity prediction and online trend detection [27, 126–129] are fundamentally linked to the recommendation task, especially when no context is available [15]. Previous works show that item popularity increases and decreases over time [27, 40, 130] and it is triggered by bursts [43, 126, 129, 131, 132]. One of the first attempts to associate human communication with the emergence of bursts was [43]. They proposed that human activities tend to alternate between periods of calm and intense activity. Plenty of works substantiate this premise [42, 54, 55, 133, 134]. Such alternating behavior points at the presence of two distinct types of audiences: the *loyal audience* which corresponds to the stable activity which occurs during the calm periods and *curious audience*, which is highly unpredictable, and responsible for the burst activity in the system [26].

The stochastic point process form the statistical framework to model random sequences of events [50]. Poisson processes, for example, are broadly used to measure stable audiences [26, 52, 53]. On the other hand, power-law distributions and self-exciting point processes have been used to model the unexpected behavior of bursts [42, 43, 54, 55]. In this work, we propose that the loyal and the curious audiences form a mix of two stochastic point processes. The difference in the intensity of the point processes defines the state of the MAB problem.

Chapter 9

CONCLUSION

The recommender systems research field is highly dynamic and challenging. Frequently, researchers confront multiple and new recommendation domains, each with its own set of social and behavioral dynamics. Additionally, civil and academic societies have been expressing a need for more transparent machine learning methods. Therefore, to the establishment of the field's future, it is crucial to consistently develop methodologies and algorithms that are not only accurate, but also comprehensive and interpretable.

In this thesis, we addressed the problem of proposing two comprehensive, theoretically inspired and cluster-induced recommendation methods: *Orthogonal Inductive Matrix Completion* and *Burst-induced Multi-armed Bandit*. The first one is a context-aware method, whereas the second one is categorized as context-free. Apart from being highly accurate, these methods improve interpretability by describing and quantifying features of the datasets they characterize.

We built our model's assumptions under solid theoretical foundations. For OMIC we provided theoretical guarantees in the form of generalization bounds. We showed that one needs to know just a few entries to accurately recover the rating matrix if the structure of the ground-truth highly correlates to the clustering side information. For the BMAB algorithm, we provided regret guarantees under mild conditions. We outlined how the system's stability affects the expected reward.

Finally, we conducted extensive experiments to validate our proposed methodologies. In a controlled environment, we implemented synthetic data generation techniques capable of replicating the domains for which OMIC and BMAB were designed. As a result, we were able to analyze our algorithms' performance across a broad ground-truth spectrum. Additionally, we replicate a real-world scenario by utilizing well-established recommender datasets. After comparing our approaches to several baselines, we observe that they achieved SOTA results in terms of accuracy.

9.1 Summary of Results

The main results of this thesis can be summarized as follows.

PART ONE: CONTEXT-AWARE RECOMMENDATION – **Chapter 2:** we introduced OMIC, a framework of inductive matrix completion learning methods, which imposes orthogonal constraints on the columns of the inductive matrices. Aligned to [29], our method imposes nuclear-norm regularization as an effective convex relaxation of the rank constraint. OMIC is a comprehensive method because it models biases, cluster side information, and a pure low-rank term simultaneously. Notably, we are the first to apply such refinements in combination. Additionally, our model gives rise to an interpretable solution, as each ground truth matrix can be uniquely represented. Thus, the magnitude of a predictor's terms can be interpreted as to its relevance to the model. Furthermore, we also provided an efficient optimization algorithm to deal with our optimization problem, proved its convergence and developed a scalable implementation. Finally, we extended our core methodology by developing a technique for recovering latent clusters in the absence of side information. Differently from previous approaches [88–90], we showed that community behavior and continuous low-rank structure could coexist in the same matrix.

Chapter 3: we provide a theoretical analysis of OMIC. At first, we proved that the predictor has a unique decomposition. This fact theoretically justifies the experiment section's interpretability analysis. Furthermore, we provide theoretical guarantees for our method in the distribution-free case (with no assumption on the ground truth distribution). Observe that, when the ground truth matrix can be approximated by community behavior, the given sample complexity bound makes excellent use of the side information by remaining independent of the matrix's size while further refining the dependence on the side information's dimensions. We also provided bounds for cluster side information by applying an adjusted trace norm regulariser in the case that sampling distribution is known [64].

Chapter 4: we conducted extensive experiments using both synthetic and real-world data. Through the use of synthetic data, we demonstrated that OMIC outperforms Soft-Impute [29] and is more flexible across a broad range of side information. Additionally, our method was capable of detecting latent biases and clusters with high accuracy. When applied to real-world recommender data, we demonstrated that our method outperforms the SOTA [29, 38] in terms of accuracy, with lower RMSE and higher SPC than the baselines. Then, we illustrate how OMIC produces interpretable solutions through examples taken from real-world datasets. At the end of the chapter, we presented an application of our method in the natural sciences: the prediction of activity coefficients in thermodynamics. We are the first to use matrix completion for this task, and our error deviation was lowest among the benchmarked models [58, 68].

Part Two: context-free recommendation – **Chapter 6:** we introduced a non-stationary and context-free multi-armed bandit problem. Different from prior work [21–25], in our model the recommender audience's temporal dynamics influence the reward distribution. To solve this problem, we developed the BMAB algorithm, a cluster-induced approach. By extending the remarkable TS's bounds of [111], we established regret guarantees for the BMAB algorithm, under the conditions that the states are recoverable and bursts are separable. Further, we experimentally analyzed the proposed regret bounds. Our bounds demonstrate that stable environments with fewer bursts expect to have a higher reward. This fact is rather intuitive because also more occasional changes in the distribution of rewards are expected.

A crucial step of the BMAB algorithm is to determine which audience is most dominant at time t. To do so, the algorithm requires knowledge of λ_L that corresponds to the system's expected rate of requisition if just the loyal audience is considered. As a result, we proposed an EM approach that is capable of disentangling the loyal audience's slowly varying regular activity from the curious activity occurring in bursts. Our method requires significantly fewer approximations than alternatives [26].

Chapter 7: we conducted synthetic and real-world data experiments with the methods described in Chapter 6. By analyzing the empirical reward, we observed that the BMAB algorithm consistently outperforms the baselines in all synthetic scenarios. Also, on the real-world recommender scenario, our algorithm outperformed all the baselines [18, 21–23, 25, 113] except in one dataset, where the performance was comparable to that of the Thompson Sampling algorithm [18]. After carefully analyzing the MovieLenz dataset, we concluded that the reason was that the audience dynamics were stationary at the observed period.

Regarding the task of separating loyal and curious audiences, our model surpasses the baselines [26, 50, 114, 115] by better fitting 9 out of 11 analyzed real-world datasets. To interpret our results, we proposed two indices to characterize the system's audience: absolute and relative loyalty. For instance, based on the GitHub dataset, we could conclude that the *loyal audience* dominates most of the time (on average, around 80%). However, only half of the activities are carried out by them, i.e., the other half comes from the *curious*.

9.2 Future work

As extensively discussed in this work, orthogonal constraints benefit IMC in a variety of properties. However, some of the available side information is naturally not orthogonal [38]. Therefore, there is a lack of any non-orthogonal comprehensive method that combines biases, non-orthogonal side information, and a pure low-rank term. Although such a method would lose interpretability properties, exploring its own properties and special guarantees would be interesting.

Besides the recommendation task, we explored how to recover latent clusters. We do not make any assumptions on the group properties, other than the fact they are disjointed. However, the great part of the recommendation system has some spatial features (e.g., the user location, warehouse positioning, etc.). Thus, future work could also observe if constrained cluster methods can compose the rating structure. For instance, one could establish that only users of contiguous areas are able to be aggregated inside of a cluster [135, 136].

Regarding our context-free methods, a future perspective to investigate is how to exploit audience dynamics in more complex reinforcement learning scenarios. One possibility is to explore the model-free reinforcement learning methods, such as *Q-learning* [137]. For instance, the update of the Q-function can be influenced by the audience dynamics: the learning rate and discount factor might assume different values according to which audience is dominant at the moment of the update. Another possibility is to have more deeply analyzed the interaction between the state detector and the rewarding process. [138] models the loyal audience also as a PW-HPP. In this case, the audience considers that a topic loyal audience's rate can increase or decrease after transitions. Although the major part of the analyzed time-series does not present transitions, it might be a significant refinement for a specific domain.

At last, adversarial MABs have gained significant importance in the RS community [139–141]. However, the emphasis is once again on the distribution of rewards. Therefore, another relevant question is how susceptible our approach might be to false popularity attacks. This discussion is related to how connected the popularity of the items is to the users' ratings. Thus a reasonable extension is to assume that, instead of each burst having a stationary rewards distribution, an adversary influences the observation of the rewards.

Appendix A

THE OMIC ALGORITHM CONVERGENCE GUARANTEES

We will now prove Theorems 2.3.1 and 2.3.2. The proofs rely mostly on adaptations of the techniques from [29], together with extensive use of the rotational invariance of the Frobenius and nuclear norms, as well as the linear independence of the spaces corresponding to each side information pairs.

A.1 The fully-known case

Proof of Proposition 2.3.1. Equation (2.9) follows from the fact that $M^{(k,l)}$ in the decomposition is unique and determined by the formula $M^{(k,l)} = (X^{(k)})^\top Z Y^{(l)}$. This itself follows from the orthogonality of the side information matrices after multiplying each side of equation (2.12) by $(X^{(k)})^\top$ on the left and $Y^{(l)}$ on the right. The equivalence between the next two problems also follows.

As to the fact that $S_\Lambda(Z)$ is the solution to problem (2.11), let us first note that the case $K = L = 1$ with identity side information is just lemma 1 in [29].

Now, note that

$$\begin{aligned} &\|\tilde{Z} - Z\|_{\mathrm{Fr}}^2 \\ &= \sum_{k,l} \|X^{(k)} M^{(k,l)} (Y^{(l)})^\top - X^{(k)} \tilde{M}^{(k,l)} (Y^{(l)})^\top\|_{\mathrm{Fr}}^2 \\ &= \sum_{k,l} \|M^{(k,l)} - \tilde{M}^{(k,l)}\|_{\mathrm{Fr}}^2, \end{aligned} \tag{A.1}$$

where at the first equality, we have used the orthogonality of the terms of the sum with respect to the Frobenius inner product, at the second equality, we have used the rotational invariance of the Frobenius norm. Here $\tilde{M}^{(k,l)} = (X^{(k)})^\top Z Y^{(l)}$, so that $Z = \sum_{k,l} X^{(k)} \tilde{M}^{(k,l)} (Y^{(l)})^\top$.

Using this, we can reformulate the problem (2.11) as follows:

$$\min \quad \sum_{k,l} \frac{1}{2} \|M^{k,l} - (X^{(k)})^\top Z Y^{(l)}\|_{\text{Fr}}^2 + \sum_{k=1}^{K} \sum_{l=1}^{L} \lambda_{k,l} \left\| M^{(k,l)} \right\|_*, \tag{A.2}$$

which can be solved as KL independent optimization problems, with the solution corresponding to index (k,l) being given by $M^{(k,l)} = S_{\lambda_{k,l}}((X^{(k)})^\top Z Y^{(l)})$, by an application of lemma 1 from [29]. The proposition follows.

□

A.2 Convergence guarantees

Let us dispose with the following straightforward observation:

Lemma A.2.1. *The generalized singular value thresholding operator* S_Λ (2.3.1) *satisfies, for any two matrices* $Z_1, Z_2 \in \mathbb{R}^{m \times n}$,

$$\|S_\Lambda(Z_1) - S_\Lambda(Z_2)\|_{\text{Fr}} \leqslant \|Z_1 - Z_2\|_{\text{Fr}}, \tag{A.3}$$

and in particular, $S_\Lambda(\cdot)$ *is a continuous map.*

Proof. This follows from the corresponding lemma 3 in [29], together with the definition of the operator S_Λ:

$$\begin{aligned}
&\|S_\Lambda(Z_1) - S_\Lambda(Z_2)\|_{\text{Fr}}^2 \\
&= \left\| \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} S_{\lambda_{k,l}} \left((X^{(k)})^\top Z_1 Y^{(l)} \right) (Y^{(l)})^\top - \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} S_{\lambda_{k,l}} \left((X^{(k)})^\top Z_2 Y^{(l)} \right) (Y^{(l)})^\top \right\|_{\text{Fr}}^2 \\
&= \left\| \sum_{k=1}^{K} \sum_{l=1}^{L} X^{(k)} \left(S_{\lambda_{k,l}} \left((X^{(k)})^\top Z_1 Y^{(l)} \right) - S_{\lambda_{k,l}} \left((X^{(k)})^\top Z_2 Y^{(l)} \right) \right) (Y^{(l)})^\top \right\|_{\text{Fr}}^2 \\
&= \sum_{k=1}^{K} \sum_{l=1}^{L} \left\| S_{\lambda_{k,l}} \left((X^{(k)})^\top (Z_1 - Z_2) Y^{(l)} \right) \right\|_{\text{Fr}}^2 \\
&\leqslant \sum_{k=1}^{K} \sum_{l=1}^{L} \left\| (X^{(k)})^\top (Z_1 - Z_2) Y^{(l)} \right\|_{\text{Fr}}^2 \\
&= \|Z_1 - Z_2\|_{\text{Fr}}^2,
\end{aligned} \tag{A.4}$$

where at the fourth line, we have used Lemma 3 from [29].

□

Now, let us define the quantity

$$Q(A|B) = \frac{1}{2} \|P_\Omega(R) + P_{\Omega^\perp}(B) - A\|_{\text{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top A Y^{(l)}\|_*.$$

We have that the loss $\mathcal{L}(Z)$ corresponding to a matrix Z can be written $Q(Z|Z)$. Furthermore, let us define $Z^{i+1} = \arg\min_Z Q(Z|Z^i)$ (since this is an instance of the fully known case, the solution is unique and given by the operator S_Λ above). We now have the following lemma, which shows that the loss decreases monotonically with i:

Lemma A.2.2. *Define the sequence* Z^i *by* $Z^{i+1} = \arg\min_Z Q(Z, Z^i)$ *(with any starting point, for instance* $Z^0 = 0$*), which is equivalent to definition* (2.18)*. We have*

$$\mathcal{L}(Z^{i+1}) \leqslant Q(Z^{i+1}|Z^k) \leqslant \mathcal{L}(Z^i). \tag{A.5}$$

Proof. The proof is based on the proof of Lemma 2 in [29]. We have

$$\begin{aligned}
\mathcal{L}(Z^i) &= Q(Z^i|Z^i) \\
&= \frac{1}{2}\|R_\Omega + P_{\Omega^\perp}(Z^i) - Z^i\|_{\mathrm{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z^i Y^{(l)}\|_* \\
&\geqslant \min_Z \frac{1}{2}\|R_\Omega + P_{\Omega^\perp}(Z^i) - Z\|_{\mathrm{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z Y^{(l)}\|_* \\
&= Q(Z^{i+1}|Z^i) \\
&= \frac{1}{2}\|(R_\Omega - P_\Omega(Z^{i+1}) + (P_{\Omega^\perp}(Z^i) - P_{\Omega^\perp}(Z^{i+1}))\|_{\mathrm{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z^{i+1} Y^{(l)}\|_* \\
&= \frac{1}{2}\left\|(R_\Omega - P_\Omega(Z^{i+1})\right\|_{\mathrm{Fr}}^2 + \frac{1}{2}\left\|(P_{\Omega^\perp}(Z^i) - P_{\Omega^\perp}(Z^{i+1}))\right\|_{\mathrm{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z^{i+1} Y^{(l)}\|_* \\
&\geqslant \frac{1}{2}\left\|(R_\Omega - P_\Omega(Z^{i+1})\right\|_{\mathrm{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z^{i+1} Y^{(l)}\|_* \\
&= Q(Z^{i+1}, Z^{i+1}) = \mathcal{L}(Z^{i+1}).
\end{aligned} \tag{A.6}$$

□

Next, we have the following lemma:

Lemma A.2.3. *The sequence* $\|Z^i - Z^{i-1}\|_{\mathrm{Fr}}$ *is monotone decreasing:*

$$\|Z^i - Z^{i+1}\|_{\mathrm{Fr}} \leqslant \|Z^i - Z^{i-1}\|_{\mathrm{Fr}}. \tag{A.7}$$

Furthermore,

$$Z^i - Z^{i+1} \to 0 \quad \textit{as} \quad i \to \infty. \tag{A.8}$$

Proof. We have

$$\begin{aligned}
&\|Z^i - Z^{i+1}\|_{\mathrm{Fr}}^2 \\
&= \|S_\Lambda\left(P_{\Omega^\perp}(Z^{i-1}) + R_\Omega\right) - S_\Lambda\left(P_{\Omega^\perp}(Z^i) + R_\Omega\right)\|_{\mathrm{Fr}}^2
\end{aligned}$$

$$
\begin{aligned}
&\leqslant \| \big(P_{\Omega^\perp}(Z^{i-1}) + R_\Omega\big) - \big(P_{\Omega^\perp}(Z^i) + R_\Omega\big) \|_{\text{Fr}}^2 \\
&\qquad \text{By Lemma A.2.1} \\
&= \|P_{\Omega^\perp}(Z^{i-1}) - P_{\Omega^\perp}(Z^i)\|_{\text{Fr}}^2 && \text{(A.9)} \\
&\leqslant \|Z^i - Z^{i-1}\|_{\text{Fr}}^2, && \text{(A.10)}
\end{aligned}
$$

which proves the first statement (A.7). As for the second statement (A.8), it will follow from the following two claims:

Claim 1: $P_\Omega(Z^i - Z^{i+1}) \to 0$.
Claim 2: $P_{\Omega^\perp}(Z^i - Z^{i+1}) \to 0$.

Proof of Claim 1: Note that by inequality (A.7), the sequence $\|Z^i - Z^{i+1}\|_{\text{Fr}}$ must converge. In particular, $\|Z^i - Z^{i+1}\|_{\text{Fr}} - \|Z^i - Z^{i-1}\|_{\text{Fr}} \to 0$, and by inequalites (A.9) and (A.10),

$$
\|P_{\Omega^\perp}(Z^{i-1}) - P_{\Omega^\perp}(Z^i)\|_{\text{Fr}} - \|Z^i - Z^{i-1}\|_{\text{Fr}} \to 0,
$$

from which we conclude that

$$
\|P_\Omega(Z^i) - P_\Omega(Z^{i+1})\|_{\text{Fr}}^2 \to 0.
$$

Claim 1 follows.
Proof of Claim 2: We know by inequality (A.5) that $\mathcal{L}(Z^i)$ must converge, and thus $\mathcal{L}(Z^i) - \mathcal{L}(Z^{i+1}) \to 0$, from which it follows that

$$
Q(Z^{i+1}|Z^i) - Q(Z^{i+1}|Z^{i+1}) \to 0. \tag{A.11}
$$

Now,

$$
\begin{aligned}
&Q(Z^{i+1}|Z^i) - Q(Z^{i+1}|Z^{i+1}) \\
&= \frac{1}{2}\|R_\Omega + P_{\Omega^\perp}(Z^i) - Z^{i+1}\|_{\text{Fr}}^2 + \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z^{i+1} Y^{(l)}\|_* \\
&- \frac{1}{2}\|R_\Omega + P_{\Omega^\perp}(Z^{i+1}) - Z^{i+1}\|_{\text{Fr}}^2 - \sum_{k,l} \lambda_{k,l} \|(X^{(k)})^\top Z^{i+1} Y^{(l)}\|_* \\
&= \frac{1}{2}\|P_{\Omega^\perp}(Z^{i+1}) - P_{\Omega^\perp}(Z^i)\|_{\text{Fr}}^2, && \text{(A.12)}
\end{aligned}
$$

which, together with (A.11), implies claim 2. □

The next step is to prove that each limit point of the sequence Z^i is a solution to the optimization problem (2.1). To prove this, we will need the following lemma:

Lemma A.2.4. *Let* $Z_{n_i} \to Z^\infty$ *be a convergent subsequence of* Z_i. *Let* $p_{n_i} \in \partial \sum_{k,l} \|(X^{(k)})^\top Z^i Y^{(l)}\|_*$ *be a sequence of subgradients of our regularizer* $\sum_{k,l} \|M^{(k,l)}\|_*$ *evaluated at* Z^i. *There exists a convergent subsequence of* p_{m_i} *which converges to some*

$$p \in \partial \sum_{k,l} \|(X^{(k)})^\top Z^\infty Y^{(l)}\|_*,$$

a subgradient of our regularizer, evaluated at the limit Z^∞.

Proof. First, recall from [142] and [29] that the set of subgradients of the nuclear norm of a matrix A is given by

$$\partial\|A\|_* = \left\{UV^\top + W, U^\top W = 0 = VW^\top, \|W\|_\sigma \leqslant 1\right\},$$

where $UDV^\top$ is the SVD of the matrix A. Using the chain rule and the fact that the side information matrices $X^{(k)}, Y^{(l)}$ are constant, we can calculate the set of subgradients of our regularizer evaluated at both Z^i and Z^∞ as follows:

$$\partial \sum_{k,l} \|(X^{(k)})^\top Z^i Y^{(l)}\|_* = \left\{ \sum_{k,l} U^i_{k,l}(V^i_{k,l})^\top + W^i_{k,l}, (U^i_{k,l})^\top W^i_{k,l} = 0, W^i_{k,l} V^i_{k,l} = 0, \|W^i_{k,l}\|_\sigma \leqslant 1 \right\} \tag{A.13}$$

and

$$\partial \sum_{k,l} \|(X^{(k)})^\top Z^\infty Y^{(l)}\|_* = \left\{ \sum_{k,l} U_{k,l} V^\top_{k,l} + W_{k,l}, U^\top_{k,l} W_{k,l} = 0, W_{k,l} V_{k,l} = 0, \|W_{k,l}\|_\sigma \leqslant 1 \right\}, \tag{A.14}$$

where $U_{k,l} D_{k,l} V^\top_{k,l}$ (resp. $U^i_{k,l} D^i_{k,l} (V^i_{k,l})^\top$) is the singular value decomposition of $(X^{(k)})^\top Z^\infty Y^{(l)}$(resp. $(X^{(k)})^\top Z^i Y^{(l)}$).

By compactness, there exists a subsequence m_i of n_i such that W^{m_i} converges to a value W. By continuity of the spectral norm, we also have $\|W\|_* \leqslant 1$. Furthermore, it follows from the convergence of Z^{n_i} (and in particular, of Z^{m_i}) to Z^∞ that $\sum_{k,l} U^{m_i}_{k,l}(V^{m_i}_{k,l})^\top \to \sum_{k,l} U_{k,l}(V_{k,l})^\top$. The result follows.

□

Proposition A.2.5. *Every limit point of the sequence* $(Z^i)_{i\in\mathbb{N}}$ *defined in* (2.18) *is a stationary point of the loss function* $\mathcal{L}(Z) = \frac{1}{2}\|\tilde{Z} - Z\|_{\text{Fr}}^2 + \sum_{k=1}^{K}\sum_{l=1}^{L} \left\|(X^{(k)})^\top Z Y^{(l)}\right\|_*$ *defined in* (2.10)*. Hence, it is also a solution to the fixed point equation*

$$Z = S_\Lambda\left(R_\Omega + P_{\Omega^\perp}(Z)\right). \tag{A.15}$$

Proof. Let Z^∞ be such a limit point. There exists a subsequence Z^{n_i} such that $Z^{n_i} \to Z^\infty$.

By Lemma A.2.3 ,we have $Z^{n_i} - Z^{n_i-1} \to 0$, which by continuity of the operator S_Λ implies that

$$R_\Omega + P_{\Omega^\perp}(Z^{n_i-1}) - Z^{n_i} \to R_\Omega - P_\Omega(Z^\infty). \tag{A.16}$$

Now, note that by definition of Z^i,

$$\forall i, 0 \in \partial Q(Z^i|Z^{i-1}) = -(P_\Omega(R) + P_{\Omega^\perp}(Z^{i-1}) - Z^i) + \partial \sum_{k,l} \|(X^{(k)})^\top Z^i Y^{(l)}\|_*.$$

Thus, we can choose, for all i, a $p_i \in \partial \sum_{k,l} \|(X^{(k)})^\top Z^i Y^{(l)}\|_*$ such that $p_i - (P_\Omega(R) + P_\Omega(Z^{i-1}) - Z^i) = 0$. Now, by Lemma A.2.4, there exists a subsequence Z^{m_i} of Z^{n_i} such that $p_{m_i} \to p$ for some

$$p \in \partial \sum_{k,l} \|(X^{(k)})^\top Z^\infty Y^{(l)}\|_*. \tag{A.17}$$

Putting equations (A.16) and (A.17) together, we obtain

$$\begin{aligned} 0 &= p_{m_i} - (P_\Omega(R) + P_{\Omega^\perp}(Z^{m_i-1}) - Z^{m_i}) \\ &\to p - R_\Omega - P_\Omega(Z^\infty). \end{aligned} \tag{A.18}$$

Thus, 0 is a subgradient of $\mathcal{L}$ evaluated at Z^∞. The first statement of the Proposition follows. As for the second statement, note that

$$Z^{m_i} = S_\Lambda\left(R_\Omega + P_{\Omega^\perp}(Z^{m_i-1})\right). \tag{A.19}$$

Furthermore, by Lemma A.2.3, $Z^{m_i} - Z^{m_i-1} \to 0$, and therefore $Z^{m_i-1} \to Z^\infty$. Thus, using the continuity of the generalized singular value thresholding operator, we obtain by passing to the limits in (A.19):

$$Z^\infty = S_\Lambda\left(R_\Omega + P_{\Omega^\perp}(Z^\infty)\right), \tag{A.20}$$

as expected. □

Proof of Theorem 2.3.1. In Proposition A.2.5, we have already proved that any limit point of the sequence $(Z^i)_{i\in\mathbb{N}}$ (defined in equation (2.18)) is a stationary point of the loss function, and therefore a solution to the optimization problem (2.1). Thus, the only thing left to prove is that the sequence $(Z^i)_{i\in\mathbb{N}}$ converges: indeed, if that is the case, its limit will be its (only) limit point, and will be a solution to problem.

Let us first dispense with the following simple observation: by Lemma A.2.2, for any i, we have

$$\mathcal{L}(Z^i) \leqslant \mathcal{L}(Z^0). \tag{A.21}$$

Since the objective function $\mathcal{L}$ is a continuous function of the matrix Z, the set of matrices Z satisfying equation (A.21) is compact. Thus, by compactness, there exists at least one limit point $\bar{Z}$.

Now, by the continuity of S_Λ and the definition of Z^i, we have, for any i:

$$\begin{aligned}
&\|\bar{Z} - Z^i\|_{\mathrm{Fr}}^2 \\
&= \left\|S_\Lambda\left(R_\Omega + P_{\Omega^\perp}(\bar{Z})\right) - S_\Lambda\left(R_\Omega + P_{\Omega^\perp}(Z^{i-1})\right)\right\|_{\mathrm{Fr}}^2 \\
&\leqslant \left\|\left(R_\Omega + P_{\Omega^\perp}(\bar{Z})\right) - \left(R_\Omega + P_{\Omega^\perp}(Z^{i-1})\right)\right\|_{\mathrm{Fr}}^2 \\
&= \|P_{\Omega^\perp}(\bar{Z} - Z^{i-1})\|_{\mathrm{Fr}}^2 \leqslant \|\bar{Z} - Z^{i-1}\|_{\mathrm{Fr}}^2,
\end{aligned} \tag{A.22}$$

where at the first line, we have used Proposition A.2.5 and the definition of Z^i.

We will now show that the sequence $(Z^i)_{i\in\mathbb{N}}$ actually converges to $\bar{Z}$. To do this, we proceed by contradiction. Assume Z^i doesn't converge towards $\bar{Z}$. By definition of convergence, this implies that there must exist an $\epsilon_* > 0$ such that there exists an infinite subsequence $Z^{I_1}, Z^{I_2}, \ldots$ such that for all i, $\|Z^{I_i} - \bar{Z}\|_{\mathrm{Fr}} \geqslant \epsilon_*$. Since the subsequence $Z^{I_1}, Z^{I_2}, \ldots$ is contained in the compact set

$$\mathcal{C}_{\epsilon_*} := \left\{Z : \mathcal{L}(Z) \leqslant \mathcal{L}(Z^0)\ \wedge\ \|Z^{I_i} - \bar{Z}\|_{\mathrm{Fr}} \geqslant \epsilon_*\right\},$$

it must have a limit point $\widetilde{Z} \in \mathcal{C}_{\epsilon_*}$ inside that set. In particular, we have

$$\|\widetilde{Z} - \bar{Z}\|_{\mathrm{Fr}} \geqslant \epsilon_* \tag{A.23}$$

Set $\epsilon = \frac{\epsilon_*}{3}$. Since $\bar{Z}$ is a limit point of $(Z^i)_{i\in\mathbb{N}}$, there certainly exists an index k such that

$$\|Z^k - \bar{Z}\|_{\mathrm{Fr}} \leqslant \epsilon. \tag{A.24}$$

Since $\widetilde{Z}$ is also a limit point of $(Z^i)_{i\in\mathbb{N}}$ (specifically, a limit point of the subsequence $(Z^{I_i})_{i\in\mathbb{N}}$), there exists an index l such that $l > k$ and

$$\|Z^l - \widetilde{Z}\|_{\mathrm{Fr}} \leqslant \epsilon. \tag{A.25}$$

On the other hand, since $l > k$ by iteratively applying equation (A.22) $l - k$ times, we obtain:

$$\|\bar{Z} - Z^l\|_{\mathrm{Fr}} \leqslant \|\bar{Z} - Z^k\|_{\mathrm{Fr}} \leqslant \epsilon, \tag{A.26}$$

where the last inequality follows from equation (A.24).

Now, by equations (A.26) and (A.25) and the triangle inequality, we obtain:

$$\begin{align}\|\widetilde{Z} - \bar{Z}\|_{\mathrm{Fr}} &\leqslant \|\widetilde{Z} - Z^l\|_{\mathrm{Fr}} + \|Z^l - \bar{Z}\|_{\mathrm{Fr}} \tag{A.27}\\ &\leqslant \epsilon + \epsilon = 2\epsilon = \frac{2\epsilon_*}{3} < \epsilon_*, \tag{A.28}\end{align}$$

which is in contradiction with equation (A.23). Thus, we deduce by contradiction that Z^i indeed converges to its only limit point $\bar{Z}$ (which we refer to as Z^∞ in the rest of the appendix). As explained at the beginning of the proof, this, together with Proposition A.2.5, implies Theorem 2.3.1, as required.

□

We can now proceed with the proof of our Theorem 2.3.2 on the worst-case convergence.

Proof of Theorem 2.3.2. The proof is exactly the same as that of theorem 2 in [29] (and also takes inspiration from [143]), and we reformulate it into our notation here for the sake of completeness only.

For $\theta \in [0, 1]$, we write $Z^i(\theta)$ for $(1-\theta)Z^i + \theta Z^\infty$. Note that by convexity of our loss function $\mathcal{L}$, we have $\mathcal{L}(Z^i(\theta)) \leqslant (1-\theta)\mathcal{L}(Z^i) + \theta\mathcal{L}(Z^\infty)$.

Note also that we have

$$\|P_{\Omega^\perp}(Z^i - Z^i(\theta))\|_{\mathrm{Fr}}^2 = \theta^2\|P_{\Omega^\perp}(Z^i - Z^\infty)\|_{\mathrm{Fr}}^2 \leqslant \theta^2\|Z^i - Z^\infty\|_{\mathrm{Fr}}^2 \leqslant \theta^2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2, \tag{A.29}$$

where we have used Lemmas A.2.3 and A.2.1.

Using these facts and the definition in the construction of the sequence Z^i, we can derive the following key inequalities:

$$\begin{aligned}\mathcal{L}(Z^{i+1}) &= \min_{Z}\left[\mathcal{L}(Z) + \frac{1}{2}\|Z - Z^i\|_{\mathrm{Fr}}^2\right]\\ &\leqslant \min_{\theta\in[0,1]}\left[\mathcal{L}(Z^i(\theta)) + \frac{1}{2}\|Z^i(\theta) - Z^i\|_{\mathrm{Fr}}^2\right]\\ &\leqslant \min_{\theta\in[0,1]}\left[\mathcal{L}(Z^i) + \theta(\mathcal{L}(Z^\infty) - \mathcal{L}(Z^i)) + \frac{1}{2}\theta^2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2\right].\end{aligned} \tag{A.30}$$

The last expression is minimised for $\theta = \theta^i$ where

$$\theta^i = \min\left(\frac{\mathcal{L}(Z^i) - \mathcal{L}(Z^\infty)}{\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2}, 1\right). \tag{A.31}$$

(If $\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2 = 0$, then $Z^i = Z^\infty \quad \forall i$ and there is nothing to prove.) Recall also that θ^i is a decreasing sequence (cf. Lemma A.2.2): if $\theta^i \leqslant 1$, then $\theta^j \leqslant 1 \quad \forall j > i$. Suppose $\theta^0 = 1$. Then, plugging this back into equation (A.30), we obtain:

$$\mathcal{L}(Z^1) - \mathcal{L}(Z^\infty) \leqslant \frac{1}{2}\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2, \tag{A.32}$$

and therefore $\theta^1 \leqslant \frac{1}{2}$. Thus, in all cases, $\theta^i < 1 \quad \forall i \geqslant 1$. Note also that if $\theta^0 = 1$, inequality (2.19) is satisfied (this follows from inequality (A.32)).

Now, for $i \geqslant 1$, we can just use the explicit expression (A.31) for θ, which, plugged back into equation (A.30), gives:

$$\mathcal{L}(Z^{i+1}) - \mathcal{L}(Z^i) \leqslant -\frac{(\mathcal{L}(Z^i) - \mathcal{L}(Z^\infty))^2}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2}. \tag{A.33}$$

Now, writing α_i for $(\mathcal{L}(Z^i) - \mathcal{L}(Z^\infty))$ (which is a decreasing sequence, as shown by Lemma A.2.3) and using the above expression, we obtain

$$\alpha_i \geqslant \frac{\alpha_i^2}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} + \alpha_{i+1} \geqslant \frac{\alpha_i\alpha_{i+1}}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} + \alpha_{i+1}, \tag{A.34}$$

which yields:

$$\alpha_{i+1}^{-1} \geqslant \frac{1}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} + \alpha_i^{-1}. \tag{A.35}$$

Summing both sides for the index running from 1 to $i-1$, we obtain:

$$\alpha_i^{-1} \geqslant \frac{i-1}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} + \alpha_1^{-1}. \tag{A.36}$$

Since $\theta_1 < 1$, by definition of θ_1, we obtain $\frac{\alpha_1}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} \leqslant \frac{1}{2}$. Plugging this back into equation (A.36), we obtain:

$$\begin{aligned}\alpha_i^{-1} &\geqslant \frac{i-1}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} + \alpha_1^{-1} \\ &\geqslant \frac{i-1}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} + \frac{1}{\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2} \\ &= \frac{i+1}{2\|Z^0 - Z^\infty\|_{\mathrm{Fr}}^2},\end{aligned}$$

which yields inequality (2.19) after inverting both sides.

□

Bibliography

[1] Kurt D Bollacker, Steve Lawrence, and C Lee Giles. Citeseer: An autonomous web agent for automatic retrieval and identification of interesting publications. In *Proceedings of the second international conference on Autonomous agents*, pages 116–123, 1998.

[2] Industry Arc. Recommendation engine market - industry analysis, market size, share, trends,application analysis, growth and forecast 2020 - 2025, 2019.

[3] Yan-Ying Chen, An-Jung Cheng, and Winston H Hsu. Travel recommendation by mining people attributes and travel group types from community-contributed photos. *IEEE Transactions on Multimedia*, 15(6):1283–1295, 2013.

[4] Ingrid A Christensen and Silvia Schiaffino. Entertainment recommender systems for group of users. *Expert Systems with Applications*, 38(11):14127–14135, 2011.

[5] Shunichi Seko, Manabu Motegi, Takashi Yagi, and Shinyo Muto. Video content recommendation for group based on viewing history and viewer preference. In *ITE Technical Report 35.7*, pages 25–26. The Institute of Image Information and Television Engineers, 2011.

[6] Evgeny Frolov and Ivan Oseledets. Hybridsvd: when collaborative information is not enough. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 331–339, 2019.

[7] Sriharsha Dara, C Ravindranath Chowdary, and Chintoo Kumar. A survey on group recommender systems. *Journal of Intelligent Information Systems*, pages 1–25, 2019.

[8] Jyun-Yu Jiang, Patrick H Chen, Cho-Jui Hsieh, and Wei Wang. Clustering and constructing user coresets to accelerate large-scale top-k recommender systems. In *Proceedings of The Web Conference 2020*, pages 2177–2187, 2020.

[9] Kwangjun Ahn, Kangwook Lee, Hyunseung Cha, and Changho Suh. Binary rating estimation with graph side information. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 4277–4288, 2018.

[10] Qiaosheng Zhang, Vincent YF Tan, and Changho Suh. Community detection and matrix completion with social and item similarity graphs. *arXiv preprint arXiv:1912.04099*, 2019.

[11] James Bennett, Stan Lanning, et al. The netflix prize. In *Proceedings of KDD cup and workshop*, volume 2007, page 35. Citeseer, 2007.

[12] Charu C Aggarwal et al. *Recommender systems*, volume 1. Springer, 2016.

[13] Robert M Bell, Yehuda Koren, and Chris Volinsky. The bellkor solution to the netflix prize. *KorBell Team's Report to Netflix*, 2007.

[14] Xiao Zhang, Simon Du, and Quanquan Gu. Fast and sample efficient inductive matrix completion via multi-phase procrustes flow. In *International Conference on Machine Learning*, pages 5756–5765. PMLR, 2018.

[15] Crícia Z Felício, Klérisson VR Paixão, Celia AZ Barcelos, and Philippe Preux. A multi-armed bandit model selection for cold-start user recommendation. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*, pages 32–40, 2017.

[16] Chunqiu Zeng, Qing Wang, Shekoofeh Mokhtari, and Tao Li. Online context-aware recommendation with time varying multi-armed bandit. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 2025–2034, 2016.

[17] Aleksandrs Slivkins. Introduction to multi-armed bandits. *arXiv preprint arXiv:1904.07272*, 2019.

[18] William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3/4):285–294, 1933.

[19] Eyal Even-Dar, Shie Mannor, and Yishay Mansour. Pac bounds for multi-armed bandit and markov decision processes. In *International Conference on Computational Learning Theory*, pages 255–270. Springer, 2002.

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

[21] Jia Yuan Yu and Shie Mannor. Piecewise-stationary bandit problems with side observations. In *Proceedings of the 26th annual international conference on machine learning*, pages 1177–1184, 2009.

[22] Robin Allesiardo and Raphaël Féraud. Exp3 with drift detection for the switching bandit problem. In *2015 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 1–7. IEEE, 2015.

[23] Yang Cao, Zheng Wen, Branislav Kveton, and Yao Xie. Nearly optimal adaptive procedure with change detection for piecewise-stationary bandit. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 418–427. PMLR, 2019.

[24] Omar Besbes, Yonatan Gur, and Assaf Zeevi. Stochastic multi-armed-bandit problem with non-stationary rewards. *Advances in neural information processing systems*, 27:199–207, 2014.

[25] Aurélien Garivier and Eric Moulines. On upper-confidence bound policies for switching bandit problems. In *International Conference on Algorithmic Learning Theory*, pages 174–188. Springer, 2011.

[26] Rodrigo Alves, Renato Assuncao, and Pedro Vaz de Melo. Burstiness scale: A parsimonious model for characterizing random series of events. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1405–1414, 2016.

[27] Honglin Yu, Lexing Xie, and Scott Sanner. The lifecyle of a youtube video: Phases, content and popularity. In *Ninth international AAAI conference on web and social media*, 2015.

[28] Nathan Srebro, Noga Alon, and Tommi S Jaakkola. Generalization error bounds for collaborative prediction with low-rank matrices. In *NIPS*, volume 4, pages 5–27. Citeseer, 2004.

[29] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. Spectral regularization algorithms for learning large incomplete matrices. *The Journal of Machine Learning Research*, 11:2287–2322, 2010.

[30] Emmanuel J Candès and Terence Tao. The power of convex relaxation: Near-optimal matrix completion. *IEEE Transactions on Information Theory*, 56(5):2053–2080, 2010.

[31] Emmanuel J Candès and Benjamin Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717–772, 2009.

[32] Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from noisy entries. *The Journal of Machine Learning Research*, 11:2057–2078, 2010.

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

[34] Prateek Jain and Inderjit S Dhillon. Provable inductive matrix completion. *arXiv preprint arXiv:1306.0626*, 2013.

[35] Aditya Krishna Menon and Charles Elkan. Link prediction via matrix factorization. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer, 2011.

[36] Tianqi Chen, Weinan Zhang, Qiuxia Lu, Kailong Chen, Zhao Zheng, and Yong Yu. Svdfeature: a toolkit for feature-based collaborative filtering. *The Journal of Machine Learning Research*, 13(1):3619–3622, 2012.

[37] Mark Herbster, Stephen Pasteris, and Lisa Tse. Online matrix completion with side information. *arXiv preprint arXiv:1906.07255*, 2019.

[38] Kai-Yang Chiang, Inderjit S Dhillon, and Cho-Jui Hsieh. Using side information to reliably learn low-rank matrices from missing and corrupted observations. *The Journal of Machine Learning Research*, 19(1):3005–3039, 2018.

[39] Jürgen Lohmann, Ralph Joh, and Jürgen Gmehling. From unifac to modified unifac (dortmund). *Industrial & engineering chemistry research*, 40(3):957–964, 2001.

[40] Bo Wu, Wen-Huang Cheng, Yongdong Zhang, and Tao Mei. Time matters: Multi-scale temporalization of social media popularity. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1336–1344, 2016.

[41] Rodrigo Alves, Raquel Prates, and Elaine França. A game to support science teaching: A case study(in portuguese). In *Anais do Workshop de Informática na Escola*, 2012.

[42] Yasuko Matsubara, Yasushi Sakurai, B Aditya Prakash, Lei Li, and Christos Faloutsos. Rise and fall patterns of information diffusion: model and implications. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 6–14, 2012.

[43] Albert-Laszlo Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 435(7039):207–211, 2005.

[44] John Gittins. A dynamic allocation index for the sequential design of experiments. *Progress in statistics*, pages 241–266, 1974.

[45] Dhruv Kumar Mahajan, Rajeev Rastogi, Charu Tiwari, and Adway Mitra. Logucb: an explore-exploit algorithm for comments recommendation. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 6–15, 2012.

[46] Xu He, Bo An, Yanghua Li, Haikai Chen, Qingyu Guo, Xin Li, and Zhirong Wang. Contextual user browsing bandits for large-scale online mobile recommendation. In *Fourteenth ACM Conference on Recommender Systems*, pages 63–72, 2020.

[47] Gourab Ghatak. A change-detection based thompsonsampling framework for non-stationary bandits. *IEEE Transactions on Computers*, 2020.

[48] Cédric Hartland, Sylvain Gelly, Nicolas Baskiotis, Olivier Teytaud, and Michele Sebag. Multi-armed bandit, dynamic environments and meta-bandits. *HAL Id: hal-00113668*, 2006.

[49] Emanuele Cavenaghi, Gabriele Sottocornola, Fabio Stella, and Markus Zanker. Non stationary multi-armed bandit: Empirical evaluation of a new concept drift-aware algorithm. *Entropy*, 23(3):380, 2021.

[50] Daryl J Daley and David Vere-Jones. *An introduction to the theory of point processes: volume II: general theory and structure*. Springer Science & Business Media, 2007.

[51] Donald L Snyder and Michael I Miller. *Random point processes in time and space*. Springer Science & Business Media, 2012.

[52] Jon Kleinberg. Bursty and hierarchical structure in streams. *Data mining and knowledge discovery*, 7(4):373–397, 2003.

[53] R Dean Malmgren, Jake M Hofman, Luis AN Amaral, and Duncan J Watts. Characterizing individual communication patterns. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 607–616, 2009.

[54] Shuang-Hong Yang and Hongyuan Zha. Mixture of mutually exciting processes for viral diffusion. In *International Conference on Machine Learning*, pages 1–9. PMLR, 2013.

[55] Pedro Olmo S Vaz de Melo, Christos Faloutsos, Renato Assunção, and Antonio Loureiro. The self-feeding process: a unifying model for communication dynamics in the web. In *Proceedings of the 22nd international conference on World Wide Web*, pages 1319–1330, 2013.

[56] Zhao Kang, Chong Peng, and Qiang Cheng. Top-n recommender system via matrix completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30, 2016.

[57] Cho-Jui Hsieh, Kai-Yang Chiang, and Inderjit S Dhillon. Low rank modeling of signed networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 507–515, 2012.

[58] Fabian Jirasek, Rodrigo Alves, Julie Damay, Robert A Vandermeulen, Robert Bamler, Michael Bortz, Stephan Mandt, Marius Kloft, and Hans Hasse. Machine learning in thermodynamics: Prediction of activity coefficients by matrix completion. *The journal of physical chemistry letters*, 11(3):981–985, 2020.

[59] Vincent YF Tan, Changho Suh, et al. Community detection and matrix completion with two-sided graph side-information. *arXiv e-prints*, pages arXiv–1912, 2019.

[60] Trevor Hastie, Rahul Mazumder, Jason D Lee, and Reza Zadeh. Matrix completion and low-rank svd via fast alternating least squares. *The Journal of Machine Learning Research*, 16(1):3367–3402, 2015.

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

[62] Rafał Latała. Some estimates of norms of random matrices. *Proceedings of the American Mathematical Society*, 133(5):1273–1282, 2005.

[63] Yuanyu Wan, Jinfeng Yi, and Lijun Zhang. Matrix completion from non-uniformly sampled entries. *arXiv preprint arXiv:1806.10308*, 2018.

[64] Rina Foygel, Ruslan Salakhutdinov, Ohad Shamir, and Nathan Srebro. Learning with the weighted trace-norm under arbitrary sampling distributions. *arXiv preprint arXiv:1106.4251*, 2011.

[65] Pere Giménez-Febrer, Alba Pagès-Zamora, and Georgios B Giannakis. Generalization error bounds for kernel matrix completion and extrapolation. *IEEE Signal Processing Letters*, 27:326–330, 2020.

[66] Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*, pages 507–517, 2016.

[67] Mengting Wan, Rishabh Misra, Ndapa Nakashole, and Julian McAuley. Fine-grained spoiler detection from large-scale review corpora. *arXiv preprint arXiv:1905.13416*, 2019.

[68] Aage Fredenslund, Russell L Jones, and John M Prausnitz. Group-contribution estimation of activity coefficients in nonideal liquid mixtures. *AIChE Journal*, 21(6):1086–1099, 1975.

[69] Rong Li, Yongcheng Dong, Qifan Kuang, Yiming Wu, Yizhou Li, Min Zhu, and Menglong Li. Inductive matrix completion for predicting adverse drug reactions (adrs) integrating drug–target interactions. *Chemometrics and Intelligent Laboratory Systems*, 144:71–79, 2015.

[70] Nagarajan Natarajan and Inderjit S Dhillon. Inductive matrix completion for predicting gene–disease associations. *Bioinformatics*, 30(12):i60–i68, 2014.

[71] Donghyuk Shin, Suleyman Cetintas, Kuang-Chih Lee, and Inderjit S Dhillon. Tumblr blog recommendation with boosted inductive matrix completion. In *Proceedings of the 24th ACM international on conference on information and knowledge management*, pages 203–212, 2015.

[72] Prateek Jain, Praneeth Netrapalli, and Sujay Sanghavi. Low-rank matrix completion using alternating minimization. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 665–674, 2013.

[73] Miao Xu, Rong Jin, and Zhi-Hua Zhou. Speedup matrix completion with side information: Application to multi-label learning. In *Advances in neural information processing systems*, pages 2301–2309, 2013.

[74] Kai Zhong, Prateek Jain, and Inderjit S Dhillon. Efficient matrix sensing using rank-1 gaussian measurements. In *International conference on algorithmic learning theory*, pages 3–18. Springer, 2015.

[75] Kai Zhong, Zhao Song, Prateek Jain, and Inderjit S Dhillon. Provable non-linear inductive matrix completion. 2019.

[76] Dimitrios Rafailidis and Alexandros Nanopoulos. Modeling users preference dynamics and side information in recommender systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 46(6):782–792, 2015.

[77] Anupriya Gogna and Angshul Majumdar. Matrix completion incorporating auxiliary information for recommender system design. *Expert Systems with Applications*, 42(14):5789–5799, 2015.

[78] Xia Ning and George Karypis. Sparse linear methods with side information for top-n recommendations. In *Proceedings of the sixth ACM conference on Recommender systems*, pages 155–162, 2012.

[79] Tianqiao Liu, Zhiwei Wang, Jiliang Tang, Songfan Yang, Gale Yan Huang, and Zitao Liu. Recommender systems with heterogeneous side information. In *The world wide web conference*, pages 3027–3033, 2019.

[80] Paul W Holland, Kathryn Blackmond Laskey, and Samuel Leinhardt. Stochastic blockmodels: First steps. *Social networks*, 5(2):109–137, 1983.

[81] Emmanuel Abbe, Afonso S Bandeira, and Georgina Hall. Exact recovery in the stochastic block model. *IEEE Transactions on information theory*, 62(1):471–487, 2015.

[82] Emmanuel Abbe. Community detection and stochastic block models: recent developments. *The Journal of Machine Learning Research*, 18(1):6446–6531, 2017.

[83] Vassilis Kalofolias, Xavier Bresson, Michael Bronstein, and Pierre Vandergheynst. Matrix completion on graphs. *arXiv preprint arXiv:1408.1717*, 2014.

[84] Hao Ma, Dengyong Zhou, Chao Liu, Michael R Lyu, and Irwin King. Recommender systems with social regularization. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 287–296, 2011.

[85] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 135–142, 2010.

[86] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29:3844–3852, 2016.

[87] Mikael Henaff, Joan Bruna, and Yann LeCun. Deep convolutional networks on graph-structured data. *arXiv preprint arXiv:1506.05163*, 2015.

[88] Jungseul Ok, Se-Young Yun, Alexandre Proutiere, and Rami Mochaourab. Collaborative clustering: Sample complexity and efficient algorithms. In *International Conference on Algorithmic Learning Theory*, pages 288–329. PMLR, 2017.

[89] Jing Shi, Bin Wu, and Xiuqin Lin. A latent group model for group recommendation. In *2015 IEEE International conference on mobile services*, pages 233–238. IEEE, 2015.

[90] Ludovico Boratto, Salvatore Carta, and Michele Satta. Groups identification and individual recommendations in group recommendation algorithms. In *PRSAT@ recsys*, pages 27–34, 2010.

[91] Linas Baltrunas, Tadas Makcinskas, and Francesco Ricci. Group recommendations with rank aggregation and collaborative filtering. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 119–126, 2010.

[92] Heung-Nam Kim and Abdulmotaleb El Saddik. A stochastic approach to group recommendations in social media systems. *Information Systems*, 50:76–93, 2015.

[93] Chris Ding, Tao Li, Wei Peng, and Haesun Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135, 2006.

[94] Jialu Liu, Chi Wang, Jing Gao, and Jiawei Han. Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pages 252–260. SIAM, 2013.

[95] Tao Li, Vikas Sindhwani, Chris Ding, and Yi Zhang. Bridging domains with words: Opinion analysis with matrix tri-factorizations. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, pages 293–302. SIAM, 2010.

[96] David Gross. Recovering low-rank matrices from few coefficients in any basis. *IEEE Transactions on Information Theory*, 57(3):1548–1566, 2011.

[97] David Gross, Yi-Kai Liu, Steven T Flammia, Stephen Becker, and Jens Eisert. Quantum state tomography via compressed sensing. *Physical review letters*, 105(15):150401, 2010.

[98] Sahand Negahban and Martin J Wainwright. Restricted strong convexity and weighted matrix completion: Optimal bounds with noise. *The Journal of Machine Learning Research*, 13(1):1665–1697, 2012.

[99] Yudong Chen, Srinadh Bhojanapalli, Sujay Sanghavi, and Rachel Ward. Completing any low-rank matrix, provably. *The Journal of Machine Learning Research*, 16(1):2999–3034, 2015.

[100] Ruslan Salakhutdinov and Nathan Srebro. Collaborative filtering in a non-uniform world: Learning with the weighted trace norm. *arXiv preprint arXiv:1002.2780*, 2010.

[101] Ohad Shamir and Shai Shalev-Shwartz. Matrix completion with the trace norm: Learning, bounding, and transducing. *The Journal of Machine Learning Research*, 15(1):3401–3423, 2014.

[102] Harrison C Carlson and Allan P Colburn. Vapor-liquid equilibria of nonideal solutions. *Industrial & Engineering Chemistry*, 34(5):581–589, 1942.

[103] U Onken, J Rarey-Nies, and J Gmehling. The dortmund data bank: A computerized system for retrieval, correlation, and prediction of thermodynamic properties of mixtures. *International Journal of Thermophysics*, 10(3):739–747, 1989.

[104] Peter J Linstrom and William G Mallard. The nist chemistry webbook: A chemical data resource on the internet. *Journal of Chemical & Engineering Data*, 46(5):1059–1063, 2001.

[105] Venkat Venkatasubramanian. The promise of artificial intelligence in chemical engineering: Is it here, finally. *AIChE J*, 65(2):466–478, 2019.

[106] Rampi Ramprasad, Rohit Batra, Ghanshyam Pilania, Arun Mannodi-Kanakkithodi, and Chiho Kim. Machine learning in materials informatics: recent applications and prospects. *npj Computational Materials*, 3(1):1–13, 2017.

[107] Keith T Butler, Daniel W Davies, Hugh Cartwright, Olexandr Isayev, and Aron Walsh. Machine learning for molecular and materials science. *Nature*, 559(7715):547–555, 2018.

[108] Aage Fredenslund. *Vapor-liquid equilibria using UNIFAC: a group-contribution method*. Elsevier, 2012.

[109] Andreas Klamt. Conductor-like screening model for real solvents: a new approach to the quantitative calculation of solvation phenomena. *The Journal of Physical Chemistry*, 99(7):2224–2235, 1995.

[110] Maarten R Dobbelaere, Pieter P Plehiers, Ruben Van de Vijver, Christian V Stevens, and Kevin M Van Geem. Machine learning in chemical engineering: Strengths, weaknesses, opportunities, and threats. *Engineering*, 2021.

[111] Shipra Agrawal and Navin Goyal. Further optimal regret bounds for thompson sampling. In *Artificial intelligence and statistics*, pages 99–107. PMLR, 2013.

[112] CF Jeff Wu. On the convergence properties of the em algorithm. *The Annals of statistics*, pages 95–103, 1983.

[113] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. The nonstochastic multiarmed bandit problem. *SIAM journal on computing*, 32(1):48–77, 2002.

[114] Sha Li, Xiaofeng Gao, Weiming Bao, and Guihai Chen. Fm-hawkes: A hawkes process based approach for modeling online activity correlations. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1119–1128, 2017.

[115] Tiago Santos, Simon Walk, Roman Kern, Markus Strohmaier, and Denis Helic. Self- and cross-excitation in stack exchange question & answer communities. In *The World Wide Web Conference*, pages 1634–1645, 2019.

[116] Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 58(5):527–535, 1952.

[117] Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on thompson sampling. *arXiv preprint arXiv:1707.02038*, 2017.

[118] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[119] Olivier Chapelle and Lihong Li. An empirical evaluation of thompson sampling. *Advances in neural information processing systems*, 24:2249–2257, 2011.

[120] Javier Sanz-Cruzado, Pablo Castells, and Esther López. A simple multi-armed nearest-neighbor bandit for interactive recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 358–362, 2019.

[121] Deepak Agarwal, Bo Long, Jonathan Traupman, Doris Xin, and Liang Zhang. Laser: A scalable response prediction platform for online advertising. In *Proceedings of the 7th ACM international conference on Web search and data mining*, pages 173–182, 2014.

[122] Thore Graepel, Joaquin Quinonero Candela, Thomas Borchert, and Ralf Herbrich. Web-scale bayesian click-through rate prediction for sponsored search advertising in microsoft's bing search engine. In *ICML*, 2010.

[123] Deepak Agarwal. Computational advertising: the linkedin way. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, pages 1585–1586, 2013.

[124] Jaya Kawale, Hung H Bui, Branislav Kveton, Long Tran-Thanh, and Sanjay Chawla. Efficient thompson sampling for online matrix-factorization recommendation. In *Advances in neural information processing systems*, pages 1297–1305, 2015.

[125] Edouard Fouché, Junpei Komiyama, and Klemens Böhm. Scaling multi-armed bandit algorithms. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1449–1459, 2019.

[126] Marian-Andrei Rizoiu, Lexing Xie, Scott Sanner, Manuel Cebrian, Honglin Yu, and Pascal Van Hentenryck. Expecting to be hip: Hawkes intensity processes for social media popularity. In *Proceedings of the 26th International Conference on World Wide Web*, pages 735–744, 2017.

[127] Swapnil Mishra. Bridging models for popularity prediction on social media. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, pages 810–811, 2019.

[128] Junjie Yao, Bin Cui, Yuxin Huang, and Xin Jin. Temporal and social context based burst detection from folksonomies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 24, 2010.

[129] Qingchao Kong, Wenji Mao, Guandan Chen, and Daniel Zeng. Exploring trends and patterns of popularity stage evolution in social media. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 50(10):3817–3827, 2018.

[130] James P Gleeson, Davide Cellai, Jukka-Pekka Onnela, Mason A Porter, and Felix Reed-Tsochas. A simple generative model of collective online behavior. *Proceedings of the National Academy of Sciences*, 111(29):10411–10415, 2014.

[131] Peng Bao. Modeling and predicting popularity dynamics via an influence-based self-excited hawkes process. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1897–1900, 2016.

[132] Cody Buntain and Jimmy Lin. Burst detection in social media streams for tracking interest profiles in real time. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 777–780, 2016.

[133] Qingyuan Zhao, Murat A Erdogdu, Hera Y He, Anand Rajaraman, and Jure Leskovec. Seismic: A self-exciting point process model for predicting tweet popularity. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 1513–1522, 2015.

[134] Alceu Ferraz Costa, Yuto Yamaguchi, Agma Juci Machado Traina, Caetano Traina Jr, and Christos Faloutsos. Rsc: Mining and modeling temporal activity in social media. In *Proceedings of the 21th ACM SIGKDD international conference on knowledge discovery and data mining*, pages 269–278, 2015.

[135] Orhun Aydin, Mark V Janikas, Renato Assunçao, and Ting-Hwan Lee. Skater-con: Unsupervised regionalization via stochastic tree partitioning within a consensus framework using random spanning trees. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on AI for Geographic Knowledge Discovery*, pages 33–42, 2018.

[136] Juan Carlos Duque, Raúl Ramos, and Jordi Suriñach. Supervised regionalization methods: A survey. *International Regional Science Review*, 30(3):195–220, 2007.

[137] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

[138] Rodrigo Alves, Antoine Ledent, Renato Martins Assuncao, Pedro Vaz de Melo, and Marius Kloft. Are you here to stay? disentangling the loyal audience from the curious on social media. Submitted to: *The 30th ACM International Conference on Information and Knowledge Management.*

[139] Kwang-Sung Jun, Lihong Li, Yuzhe Ma, and Xiaojin Zhu. Adversarial attacks on stochastic bandits. *arXiv preprint arXiv:1810.12188*, 2018.

[140] Irched Chafaa, E Veronica Belmega, and Mérouane Debbah. Adversarial multi-armed bandit for mmwave beam alignment with one-bit feedback. In *Proceedings of the 12th EAI International Conference on Performance Evaluation Methodologies and Tools*, pages 23–30, 2019.

[141] Mohammad Hajiesmaili, Mohammad Sadegh Talebi, John Lui, Wing Shing Wong, et al. Adversarial bandits with corruptions: Regret lower bound and no-regret algorithm. *Advances in Neural Information Processing Systems*, 33, 2020.

[142] G Alistair Watson. Characterization of the subdifferential of some matrix norms. *Linear algebra and its applications*, 170(0):33–45, 1992.

[143] Yu Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

CURRICULUM VITAE

Short biography: Rodrigo Alves has been a research assistant (during his Ph.D. studies) at the Machine Learning Group under the supervision of Prof. Marius Kloft at TU Kaiserslautern, Germany. Previously he was a Lecturer at CEFET-MG, Brazil. He holds a Bachelor's degree in Information Systems – where he received Best Student award — and a Master's in Computer Science from the Department of Computer Science at the Federal University of Minas Gerais. Alves is also a vocational educational teacher certificated by the Häme University of Applied Sciences, Finland. During his career, he has collaborated with various research groups (in different countries) and management teams. Alves is interested in machine learning and applications, especially related to data mining and how artificial intelligence correlates to other areas of computer science. He is also interested in student-centred learning methodology, particularly in relation in how to improve computer science teaching.

Vita:

Nov 16, 1987	Born—Belo Horizonte – MG, Brazil
Dec, 2012	B.S., Federal University of Minas Gerais, Belo Horizonte – MG, Brazil
May, 2015	M.S., Federal University of Minas Gerais, Belo Horizonte – MG, Brazil

Fields of Study

Major Field: Computer Science

Studies in Machine Learning: Recommender Systems